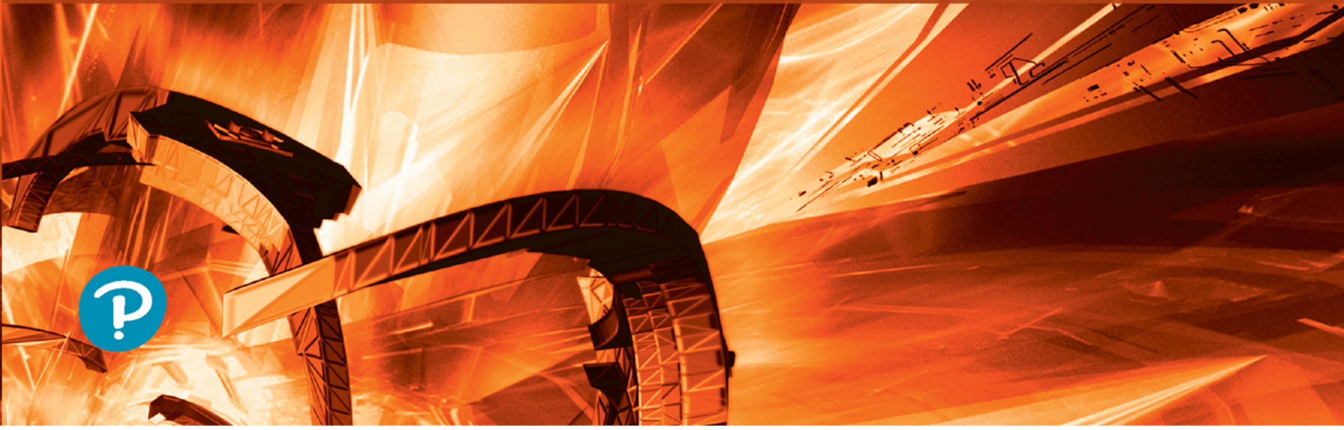
Matthew Helmke

2021 Edition
Covers 20.04, 20.10, and 21.04

Ubuntu Linux

UNLEASHED



FREE SAMPLE CHAPTER

SHARE WITH OTHERS

Matthew Helmke

with Andrew Hudson
and Paul Hudson

Ubuntu Linux

UNLEASHED

2021 Edition

SAMS

Ubuntu Linux Unleashed 2021 Edition

Copyright © 2021 by Pearson Education, Inc.

All rights reserved. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. For information regarding permissions, request forms, and the appropriate contacts within the Pearson Education Global Rights & Permissions Department, please visit www.pearson.com/permissions/. 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: 978-0-13-677885-1

ISBN-10: 0-13-677885-2

Library of Congress Control Number: 2020937219

ScoutAutomatedPrintCode

Trademarks

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Pearson Education 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.

Microsoft and/or its respective suppliers make no representations about the suitability of the information contained in the documents and related graphics published as part of the services for any purpose. All such documents and related graphics are provided "as is" without warranty of any kind. Microsoft and/or its respective suppliers hereby disclaim all warranties and conditions with regard to this information, including all warranties and conditions of merchantability, whether express, implied or statutory, fitness for a particular purpose, title and non-infringement. In no event shall Microsoft and/or its respective suppliers be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of information available from the services. The documents and related graphics contained herein could include technical inaccuracies or typographical errors. Changes are periodically added to the information herein. Microsoft and/or its respective suppliers may make improvements and/or changes in the product(s) and/or the program(s) described herein at any time. Partial screen shots may be viewed in full within the software version specified.

Microsoft® and Windows® are registered trademarks of the Microsoft Corporation in the U.S.A. and other countries. Screenshots and icons reprinted with permission from the Microsoft Corporation. This book is not sponsored or endorsed by or affiliated with the Microsoft Corporation.

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.

Special Sales

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact intlcs@pearson.com.

Editor-in-Chief
Mark Taub

Acquisitions Editor
Debra Williams

Managing Editor
Sandra Schroeder

Senior Project Editor
Lori Lyons

Copy Editor
Paula Lowell

Production Manager
Aswini Kumar/
codeMantra

Indexer
Ken Johnson

Proofreader
Charlotte Kughen

Cover Designer
Chuti Prasertsith

Compositor
codeMantra

Contents at a Glance

Introduction ... xxxi

Part I Getting Started

1 Installing Ubuntu and Post-Installation Configuration ... 1

2 Background Information and Resources ... 23

Part II Desktop Ubuntu

3 Foundations of the Linux GUI ... 31

4 Ubuntu Desktop Options ... 41

5 On the Internet ... 51

6 Productivity Applications ... 55

7 Multimedia Applications ... 63

8 Games ... 79

Part III System Administration

9 Managing Software ... 91

10 Command-Line Beginner's Class ... 105

11 Command-Line Master Class, Part 1 ... 141

12 Command-Line Master Class, Part 2 ... 167

13 Managing Users ... 201

14 Automating Tasks and Shell Scripting ... 225

15 The Boot Process ... 271

16 System-Monitoring Tools ... 281

17 Backing Up ... 301

18 Networking ... 325

19 Remote Access with SSH and VNC ... 371

20 Securing Your Machines ... 379

21 Performance Tuning ... 393

22 Kernel and Module Management ... 401

Part IV Ubuntu as a Server

23 Sharing Files and Printers ... 421

24 Common Web Server Stacks ... 437

25 Apache Web Server Management ... 443

26 Nginx Web Server Management 467
27 Other HTTP Servers 479
28 Administering Relational Database Services 483
29 NoSQL Databases 509
30 Virtualization on Ubuntu 521
31 Containers and Ubuntu 531
32 Ubuntu and Cloud Computing 535
33 Managing Sets of Servers 543
34 Handling Email 547
35 Proxying, Reverse Proxying, and Virtual Private Networks (VPNs) 563
36 Lightweight Directory Access Protocol (LDAP) 579
37 Name Serving with the Domain Name System (DNS) 587

Part V Programming Linux
38 Using Programming Tools 597
39 Using Popular Programming Languages 621
40 Helping with Ubuntu Development 635
41 Helping with Ubuntu Testing and QA 645

Index 649

NOTE

Bonus Chapters 42–44 can be accessed online at informit.com/title/9780136778851.

Part VI Bonus Online Chapters
42 Using Perl Web:1
43 Using Python Web:21
44 Using PHP Web:39

Table of Contents

Introduction **xxxi**

Part I Getting Started

1 Installing Ubuntu and Post-Installation Configuration **1**

Before You Begin the Installation 1
- Researching Your Hardware Specifications 2
- Installation Options 2
- 32-Bit Versus 64-Bit Ubuntu 4
- Planning Partition Strategies 5
- The Boot Loader 5
- Installing from DVD or USB Drive 6

Step-by-Step Installation 6
- Installing 7
- First Update 11

Shutting Down 11

Finding Programs and Files 12

Software Updater 12

The `sudo` Command 14

Configuring Software Repositories 15

System Settings 17
- Detecting and Configuring a Printer 18
- Configuring Power Management in Ubuntu 18
- Setting the Time and Date 19

Configuring Wireless Networks 20

Troubleshooting Post-Installation Configuration Problems 21

References 22

2 Background Information and Resources **23**

What Is Linux? 23

Why Use Linux? 25

What Is Ubuntu? 27

Ubuntu for Business 27

Ubuntu in Your Home 28

Getting the Most from Linux and Ubuntu Documentation 28
- Linux 29
- Ubuntu 30

Part II Desktop Ubuntu

3 Foundations of the Linux GUI 31

Foundations and the X Server 31
 Basic X Concepts 32
 Using X 33
 Elements of the `xorg.conf` File 34
Starting X 39
 Using a Display Manager 39
 Changing Window Managers 39
References 40

4 Ubuntu Desktop Options 41

Desktop Environment 41
Using GNOME: A Primer 42
KDE and Kubuntu 45
Xfce and Xubuntu 46
LXDE and Lubuntu 47
MATE and Ubuntu MATE 48
Ubuntu Budgie 49
Ubuntu Kylin 50
References 50

5 On the Internet 51

Getting Started with Firefox 52
Checking Out Google Chrome and Chromium 53
References 54

6 Productivity Applications 55

Introducing LibreOffice 56
Other Useful Productivity Software 58
 Working with PDFs 58
 Writing Scripts 59
 Working with XML and DocBook 59
 Working with LaTeX 60
 Creating Mind Maps 61
Productivity Applications Written for Microsoft Windows 61
References 61

7 Multimedia Applications 63

Sound and Music 63
 Sound Cards 64
 Sound Formats 65
 Listening to Music 65

Graphics Manipulation 66
The GNU Image Manipulation Program 66
Using Scanners in Ubuntu 67
Working with Graphics Formats 67
Capturing Screen Images 69
Other Graphics Manipulation Options 70
Using Digital Cameras with Ubuntu 70
Handheld Digital Cameras 70
Using Shotwell Photo Manager 71
Burning CDs and DVDs in Ubuntu 71
Creating CDs and DVDs with Brasero 71
Creating CDs from the Command Line 72
Creating DVDs from the Command Line 73
Viewing Video 75
Video Formats 75
Viewing Video in Linux 76
Recording and Editing Audio 76
Editing Video 77
References 77

8 Games 79

Ubuntu Gaming 79
Installing Proprietary Video Drivers 80
Online Game Sources 81
Steam 81
GOG.com 82
Humble 82
itch.io 82
LGDB 82
Game Jolt 82
Installing Games from the Ubuntu Repositories 82
Warsow 82
Scorched 3D 83
Frozen Bubble 84
SuperTux 84
Battle for Wesnoth 85
Frets on Fire 85
FlightGear 87
Speed Dreams 87
Games for Kids 88
Commercial Games 88
Playing Windows Games 88
References 89

Part III System Administration

9 Managing Software 91

Ubuntu Software 91
Using Synaptic for Software Management 92
Staying Up to Date 94
Working on the Command Line 95
  Day-to-Day APT Usage 95
  Finding Software 98
  Using `apt-get` Instead of `apt` 99
Compiling Software from Source 100
  Compiling from a Tarball 100
  Compiling from Source from the Ubuntu Repositories 101
Configuration Management 102
  dotdee 102
  Ubuntu Core 103
Using the Snappy Package Manager 103
References 104

10 Command-Line Beginner's Class 105

What Is the Command Line? 106
Accessing the Command Line 107
  Text-Based Console Login 107
  Logging Out 108
  Logging In and Out from a Remote Computer 108
User Accounts 109
Reading Documentation 111
  Using Man Pages 111
  Using `apropros` 112
  Using `whereis` 112
Understanding the Linux File System Hierarchy 112
  Essential Commands in `/bin` and `/sbin` 114
  Configuration Files in `/etc` 114
  User Directories: `/home` 115
  Using the Contents of the `/proc` Directory to Interact with or Obtain Information from the Kernel 115
  Working with Shared Data in the `/usr` Directory 117
  Temporary File Storage in the `/tmp` Directory 117
  Accessing Variable Data Files in the `/var` Directory 117
Navigating the Linux File System 117
  Listing the Contents of a Directory with `ls` 118
  Changing Directories with `cd` 120
  Finding Your Current Directory with `pwd` 120

Working with Permissions ... 120
Assigning Permissions ... 121
Directory Permissions ... 122
Altering File Permissions with `chmod` ... 123
File Permissions with `umask` ... 124
File Permissions with `chgrp` ... 125
Changing File Permissions with `chown` ... 125
Understanding Set User ID, Set Group ID, and Sticky Bit Permissions ... 125
Setting Permissions with Access Control Lists ... 127
Working with Files ... 128
Creating a File with `touch` ... 128
Creating a Directory with `mkdir` ... 129
Deleting a Directory with `rmdir` ... 129
Deleting a File or Directory with `rm` ... 130
Moving or Renaming a File with `mv` ... 131
Copying a File with `cp` ... 131
Displaying the Contents of a File with `cat` ... 132
Displaying the Contents of a File with `less` ... 132
Using Wildcards and Regular Expressions ... 133
Working as Root ... 133
Understanding and Fixing `sudo` ... 134
Creating Users ... 136
Deleting Users ... 137
Shutting Down the System ... 137
Rebooting the System ... 138
Commonly Used Commands and Programs ... 139
References ... 139

11 Command-Line Master Class, Part 1 141

Why Use the Command Line? ... 142
Using Basic Commands ... 143
Printing the Contents of a File with `cat` ... 144
Changing Directories with `cd` ... 145
Changing File Access Permissions with `chmod` ... 147
Copying Files with `cp` ... 147
Printing Disk Usage with `du` ... 148
Using `echo` ... 148
Finding Files by Searching with `find` ... 149
Searches for a String in Input with `grep` ... 151
Paging Through Output with `less` ... 152
Creating Links Between Files with `ln` ... 154

Finding Files from an Index with `locate` 156
Listing Files in the Current Directory with `ls` 156
Listing System Information with `lsblk`, `lshw`, `lsmod`, `lspci`, and `neofetch` 158
Reading Manual Pages with `man` 159
Making Directories with `mkdir` 160
Moving Files with `mv` 161
Renaming Files with `rename` 161
Deleting Files and Directories with `rm` 161
Sorting the Contents of a File with `sort` 162
Printing the Last Lines of a File with `tail` 163
Printing the Location of a Command with `which` 164
Downloading Files with `wget` 164
References 165

12 Command-Line Master Class, Part 2 167

Redirecting Output and Input 167
`stdin`, `stdout`, `stderr`, and Redirection 169
Comparing Files 170
Finding Differences in Files with `diff` 170
Finding Similarities in Files with `comm` 170
Limiting Resource Use and Job Control 171
Listing Processes with `ps` 171
Listing Jobs with `jobs` 173
Running One or More Tasks in the Background 173
Moving Jobs to the Background or Foreground with `bg` and `fg` 174
Printing Resource Usage with `top` 175
Setting Process Priority with `nice` 177
Combining Commands 178
Pipes 178
Combining Commands with Boolean Operators 180
Running Separate Commands in Sequence 180
Process Substitution 181
Executing Jobs in Parallel 181
Using Environment Variables 182
Using Common Text Editors 185
Working with `nano` 186
Working with `vi` 187
Working with `emacs` 188
Working with `sed` and `awk` 189
Working with Compressed Files 191
Using Multiple Terminals with `byobu` 192
Doing a Polite System Reset Using REISUB 194

Fixing an Ubuntu System That Will Not Boot 195
Checking BIOS 195
Checking GRUB 195
Reinstalling GRUB 195
Using Recovery Mode 196
Reinstalling Ubuntu 196
Tips and Tricks 196
Running the Previous Command 196
Running Any Previous Command 197
Running a Previous Command That Started with Specific Letters 197
Running the Same Thing You Just Ran with a Different First Word 197
Viewing Your History and More 197
Doing Two or More Things 198
Using Shortcuts 198
Confining a Script to a Directory 198
Using Coreutils 199
Reading the Contents of the Kernel Ring Buffer with `dmesg` 200
References 200

13 Managing Users 201

User Accounts 201
The Super User/Root User 202
User IDs and Group IDs 204
File Permissions 204
Managing Groups 205
Group Listing 205
Group Management Tools 206
Managing Users 207
User Management Tools 208
Adding New Users 209
Monitoring User Activity on the System 211
Managing Passwords 212
System Password Policy 212
The Password File 212
Shadow Passwords 214
Managing Password Security for Users 216
Changing Passwords in a Batch 216
Granting System Administrator Privileges to Regular Users 217
Temporarily Changing User Identity with the `su` Command 217
Granting Root Privileges on Occasion: The `sudo` Command 219

Disk Quotas 222
Implementing Quotas 222
Manually Configuring Quotas 223
Related Ubuntu Commands 223
References 224

14 Automating Tasks and Shell Scripting 225

What Is a Shell? 225
Scheduling Tasks 226
Using `at` and `batch` to Schedule Tasks for Later 227
Using `cron` to Run Jobs Repeatedly 229
Using `rtcwake` to Wake Your Computer from Sleep Automatically 231
Basic Shell Control 233
The Shell Command Line 233
Shell Pattern-Matching Support 235
Redirecting Input and Output 236
Piping Data 237
Background Processing 237
Writing and Executing a Shell Script 237
Running the New Shell Program 239
Storing Shell Scripts for System-wide Access 240
Interpreting Shell Scripts Through Specific Shells 240
Using Variables in Shell Scripts 242
Assigning a Value to a Variable 242
Accessing Variable Values 243
Positional Parameters 243
A Simple Example of a Positional Parameter 243
Using Positional Parameters to Access and Retrieve Variables from the Command Line 244
Using a Simple Script to Automate Tasks 244
Built-in Variables 246
Special Characters 247
Comparison of Expressions in `pdksh` and `bash` 250
Comparing Expressions with `tcsh` 255
The `for` Statement 259
The `while` Statement 261
The `until` Statement 263
The `repeat` Statement (`tcsh`) 263
The `select` Statement (`pdksh`) 264
The `shift` Statement 264
The `if` Statement 265

The `expr` Statement 266
The `case` Statement 267
The `break` and `exit` Statements 269
Using Functions in Shell Scripts 269
References 270

15 The Boot Process 271
Running Services at Boot 271
Beginning the Boot Loading Process 272
Loading the Linux Kernel 274
Starting and Stopping Services with `systemd` 275
Controlling Services at Boot with Administrative Tools 278
Troubleshooting Runlevel Problems 278
Boot-Repair 278
References 279

16 System-Monitoring Tools 281
Console-Based Monitoring 281
Using the `kill` Command to Control Processes 283
Using Priority Scheduling and Control 285
Displaying Free and Used Memory with `free` 286
Disk Space 286
Disk Quotas 287
Checking Log Files 287
Rotating Log Files 289
Graphical Process- and System-Management Tools 292
System Monitor 292
Conky 292
Other Graphical Process- and System-Monitoring Tools 297
KDE Process- and System-Monitoring Tools 298
Enterprise Server Monitoring 298
References 299

17 Backing Up 301
Choosing a Backup Strategy 301
Why Data Loss Occurs 302
Assessing Your Backup Needs and Resources 303
Evaluating Backup Strategies 304
Making the Choice 308
Choosing Backup Hardware and Media 308
External Hard Drive 308
Network Storage 308

Tape Drive Backups 309
Cloud Storage 309
Using Backup Software 309
`tar`: The Most Basic Backup Tool 310
The GNOME File Roller 312
The KDE `ark` Archiving Tool 312
Déjà Dup 313
Back In Time 314
Unison 315
Amanda 315
Alternative Backup Software 316
Copying Files 316
Copying Files Using `tar` 317
Compressing, Encrypting, and Sending `tar` Streams 318
Copying Files Using `cp` 318
Using `rsync` 319
Version Control for Configuration Files 320
System Rescue 323
The Ubuntu Rescue Disc 323
Restoring the GRUB2 Boot Loader 323
Saving Files from a Nonbooting Hard Drive 324
References 324

18 Networking 325

Laying the Foundation: The `localhost` Interface 326
Checking for the Availability of the Loopback Interface 326
Configuring the Loopback Interface Manually 327
Checking Connections with `ping`, `traceroute`, and `mtr` 328
Networking with TCP/IP 330
TCP/IP Addressing 331
Using IP Masquerading in Ubuntu 332
Ports 333
IPv6 Basics 334
Network Organization 337
Subnetting 337
Subnet Masks 337
Broadcast, Unicast, and Multicast Addressing 338
Hardware Devices for Networking 338
Network Interface Cards 338
Network Cable 340
Hubs and Switches 342
Routers and Bridges 343
Initializing New Network Hardware 343

Using Network Configuration Tools ... 345
    Command-Line Network Interface Configuration ... 346
    Network Configuration Files ... 350
    Using Graphical Configuration Tools ... 355
Dynamic Host Configuration Protocol ... 355
    How DHCP Works ... 356
    Activating DHCP at Installation and Boot Time ... 357
    DHCP Software Installation and Configuration ... 358
    Using DHCP to Configure Network Hosts ... 359
    Other Uses for DHCP ... 361
Wireless Networking ... 361
    Support for Wireless Networking in Ubuntu ... 361
    Choosing from Among Available Wireless Protocols ... 363
Beyond the Network and onto the Internet ... 363
Common Configuration Information ... 364
    Configuring Digital Subscriber Line Access ... 365
    Understanding PPP over Ethernet ... 366
    Configuring a PPPoE Connection Manually ... 366
    Configuring Dial-up Internet Access ... 367
    Troubleshooting Connection Problems ... 368
References ... 369

19 Remote Access with SSH and VNC 371

Setting Up an SSH Server ... 371
SSH Tools ... 372
    Using `scp` to Copy Individual Files Between Machines ... 372
    Using `sftp` to Copy Many Files Between Machines ... 373
    Using `ssh-keygen` to Enable Key-Based Logins ... 373
Virtual Network Computing ... 375
Guacamole ... 377
References ... 377

20 Securing Your Machines 379

Understanding Computer Attacks ... 379
Assessing Your Vulnerability ... 381
Protecting Your Machine ... 382
    Securing a Wireless Network ... 382
    Passwords and Physical Security ... 383
    Configuring and Using Tripwire ... 384
    Securing Devices ... 385
Viruses ... 385
Configuring Your Firewall ... 386

AppArmor ... 388
Forming a Disaster Recovery Plan ... 390
References ... 391

21 Performance Tuning 393

Storage Disk ... 394
 Linux File Systems ... 394
 The `hdparm` Command ... 395
 File System Tuning ... 396
 The `tune2fs` Command ... 396
 The `e2fsck` Command ... 397
 The `badblocks` Command ... 397
 Disabling File Access Time ... 397
Kernel ... 398
Tuned ... 399
References ... 400

22 Kernel and Module Management 401

The Linux Kernel ... 402
 The Linux Source Tree ... 403
 Types of Kernels ... 405
Managing Modules ... 406
When to Recompile ... 408
Kernel Versions ... 409
Obtaining the Kernel Sources ... 409
Patching the Kernel ... 410
Compiling the Kernel ... 412
 Using `xconfig` to Configure the Kernel ... 414
 Creating an Initial RAM Disk Image ... 418
When Something Goes Wrong ... 418
 Errors During Compile ... 418
 Runtime Errors, Boot Loader Problems, and Kernel Oops ... 419
References ... 419

Part IV Ubuntu as a Server

23 Sharing Files and Printers 421

Using Network File System ... 422
 Installing and Starting or Stopping NFS ... 422
 NFS Server Configuration ... 422
 NFS Client Configuration ... 423

Putting Samba to Work ... 424
  Manually Configuring Samba with /etc/samba/smb.conf ... 426
  Testing Samba with the testparm Command ... 429
  Starting, Stopping, and Restarting the smbd Daemon ... 429
  Mounting Samba Shares ... 430
Network and Remote Printing with Ubuntu ... 431
  Creating Network Printers ... 431
  Using the CUPS GUI ... 433
  Avoiding Printer Support Problems ... 434
References ... 435

24 Common Web Server Stacks 437

LAMP ... 437
LEMP ... 439
MEAN ... 440
References ... 441

25 Apache Web Server Management 443

About the Apache Web Server ... 443
Installing the Apache Server ... 444
  Starting and Stopping Apache ... 444
Runtime Server Configuration Settings ... 446
  Runtime Configuration Directives ... 446
  Editing apache2.conf ... 447
  Apache Multiprocessing Modules ... 449
  Using .htaccess Configuration Files ... 450
File System Authentication and Access Control ... 452
  Restricting Access with Require ... 452
  Authentication ... 453
  Final Words on Access Control ... 455
Apache Modules ... 455
  mod_access ... 456
  mod_alias ... 456
  mod_asis ... 456
  mod_auth ... 457
  mod_auth_anon ... 457
  mod_auth_dbm ... 457
  mod_auth_digest ... 457
  mod_autoindex ... 458
  mod_cgi ... 458
  mod_dir and mod_env ... 458
  mod_expires ... 458

mod_headers 458
mod_include 459
mod_info and mod_log_config 459
mod_mime and mod_mime_magic 459
mod_negotiation 459
mod_proxy 459
mod_rewrite 459
mod_setenvif 460
mod_speling 460
mod_status 460
mod_ssl 460
mod_unique_id 460
mod_userdir 460
mod_usertrack 460
mod_vhost_alias 460
Virtual Hosting 461
Address-Based Virtual Hosts 461
Name-Based Virtual Hosts 461
Logging 463
HTTPS 464
References 466

26 Nginx Web Server Management 467

About the Nginx Web Server 467
Installing the Nginx Server 469
Installing from the Ubuntu Repositories 469
Building the Source Yourself 469
Configuring the Nginx Server 470
Virtual Hosting 473
Setting Up PHP 474
Adding and Configuring Modules 475
HTTPS 476
Reference 477

27 Other HTTP Servers 479

lighttpd 479
Yaws 480
Cherokee 480
Jetty 481
thttpd 481
Apache Tomcat 482

WildFly 482
Caddy 482
References 482

28 Administering Relational Database Services 483

A Brief Review of Database Basics 484
    How Relational Databases Work 486
    Understanding SQL Basics 487
    Creating Tables 488
    Inserting Data into Tables 489
    Retrieving Data from a Database 490
Choosing a Database: MySQL Versus PostgreSQL 492
    Speed 492
    Data Locking 492
    ACID Compliance in Transaction Processing to Protect Data Integrity 493
    SQL Subqueries 494
    Procedural Languages and Triggers 494
Configuring MySQL 494
    Setting a Password for the MySQL Root User 495
    Creating a Database in MySQL 496
Configuring PostgreSQL 498
    Initializing the Data Directory in PostgreSQL 498
    Creating a Database in PostgreSQL 499
    Creating Database Users in PostgreSQL 499
    Deleting Database Users in PostgreSQL 500
    Granting and Revoking Privileges in PostgreSQL 500
Database Clients 501
    SSH Access to a Database 501
    Local GUI Client Access to a Database 503
    Web Access to a Database 503
    The MySQL Command-Line Client 504
    The PostgreSQL Command-Line Client 505
    Graphical Clients 506
References 507

29 NoSQL Databases 509

Key/Value Stores 512
    Berkeley DB 512
    Cassandra 513
    etcd 513
    Memcached and MemcacheDB 513

Redis ... 514
Riak ... 514
Scylla ... 514
Document Stores ... 514
CouchDB ... 515
MongoDB ... 516
BaseX ... 517
Wide Column Stores ... 517
BigTable ... 517
HBase ... 518
Graph Stores ... 518
Neo4j ... 518
OrientDB ... 519
HyperGraphDB ... 519
FlockDB ... 519
References ... 519

30 Virtualization on Ubuntu 521

KVM ... 523
VirtualBox ... 527
VMware ... 528
Xen ... 528
References ... 529

31 Containers and Ubuntu 531

LXC and LXD ... 532
Docker ... 533
Kubernetes ... 534
References ... 534

32 Ubuntu and Cloud Computing 535

Why a Cloud? ... 536
Software as a Service (SaaS) ... 537
Platform as a Service (PaaS) ... 537
Infrastructure as a Service (IaaS) ... 537
Metal as a Service (MaaS) ... 537
Things to Consider Before You Make Any Decisions ... 538
Ubuntu on the Public Cloud ... 538
OpenStack ... 538
Amazon Web Services ... 539
Google Cloud ... 539

Microsoft Azure ... 539
Hybrid Clouds ... 539
Canonical-Specific Cloud Offerings ... 540
Juju ... 540
Mojo: Continuous Delivery for Juju ... 541
Landscape ... 541
References ... 541

33 Managing Sets of Servers 543

Puppet ... 543
Chef ... 544
Ansible ... 544
SaltStack ... 544
CFEngine ... 545
Juju ... 545
Landscape ... 545
References ... 545

34 Handling Email 547

How Email Is Sent and Received ... 547
The Mail Transport Agent ... 548
Choosing an MTA ... 550
The Mail Delivery Agent ... 550
The Mail User Agent ... 551
Basic Postfix Configuration and Operation ... 552
Configuring Masquerading ... 554
Using Smart Hosts ... 554
Setting Message Delivery Intervals ... 555
Mail Relaying ... 555
Forwarding Email with Aliases ... 556
Using Fetchmail to Retrieve Mail ... 557
Installing Fetchmail ... 557
Configuring Fetchmail ... 557
Choosing a Mail Delivery Agent ... 560
Procmail ... 561
Spamassassin ... 561
Squirrelmail ... 561
Virus Scanners ... 562
Autoresponders ... 562
References ... 562

35 Proxying, Reverse Proxying, and Virtual Private Networks (VPNs) 563

What Is a Proxy Server? 563
Installing Squid 564
Configuring Clients 564
Access Control Lists 565
Specifying Client IP Addresses 569
Sample Configurations 570
Virtual Private Networks (VPNs) 572
    Setting Up a VPN Client 573
    Setting Up a VPN Server 575
References 577

36 Lightweight Directory Access Protocol (LDAP) 579

Configuring the Server 580
    Creating Your Schema 580
    Populating Your Directory 582
Configuring Clients 584
LDAP Administration 584
References 585

37 Name Serving with the Domain Name System (DNS) 587

Understanding Domain Names 588
    DNS Servers 589
    DNS Records 590
Setting Up a DNS Server with BIND 593
References 595

Part V Programming Linux

38 Using Programming Tools 597

Programming in C with Linux 598
Using the C Programming Project Management Tools Provided with Ubuntu 599
    Building Programs with `make` 599
    Using the `autoconf` Utility to Configure Code 601
    Debugging Tools 602
Using the GNU C Compiler 603
Programming in Java with Linux 604
Graphical Development Tools 605

IDEs and SDKs ... 605
Using the KDevelop Client ... 606
The Glade Client for Developing in GNOME ... 607
Beginning Mobile Development for Android ... 607
Hardware ... 608
Linux Kernel ... 608
Libraries ... 608
Android Runtime ... 608
Application Framework ... 608
Applications ... 609
Installing Android Studio ... 609
Creating Your First Android Application ... 609
Version Control Systems ... 609
Managing Software Projects with Git ... 610
Managing Software Projects with Bazaar ... 611
Managing Software Projects with Subversion ... 612
Continuous Integration and Continuous Delivery and DevOps Tools ... 613
CI/CD Tools ... 614
Chaos Engineering ... 614
Canonical-created Tools ... 614
Launchpad ... 614
Ubuntu Make ... 615
Creating Snap Packages ... 615
Bikeshed and Other Tools ... 616
References ... 618

39 Using Popular Programming Languages 621

Ada ... 622
Clojure ... 622
COBOL ... 623
D ... 624
Dart ... 624
Elixir ... 625
Elm ... 625
Erlang ... 625
Forth ... 626
Fortran ... 626
Go ... 626
Groovy ... 627
Haskell ... 627

Java 627
JavaScript 628
Kotlin 628
Lisp 629
Lua 629
Mono 629
OCaml 630
Perl 630
PHP 631
Python 631
Raku 631
Ruby 631
Rust 632
Scala 632
Scratch 632
Vala 633
References 633

40 Helping with Ubuntu Development 635

Introduction to Ubuntu Development 636
Setting Up Your Development System 637
  Install Basic Packages and Configure 637
  Creating a Launchpad Account 638
  Setting Up Your Environment to Work with Launchpad 638
Fixing Bugs and Packaging 640
References 643

41 Helping with Ubuntu Testing and QA 645

Community Teams 645
  Ubuntu Testing Team 646
  Ubuntu QA Team 646
Bug Squad 647
References 647

Index 649

NOTE

Bonus Chapters 42–44 can be accessed online at informit.com/title/9780136778851.

Part VI Bonus Online Chapters

42 Using Perl Web:1

Using Perl with Linux Web:1
- Perl Versions Web:2
- A Simple Perl Program Web:2

Perl Variables and Data Structures Web:4
- Perl Variable Types Web:4
- Special Variables Web:5

Perl Operators Web:6
- Comparison Operators Web:6
- Compound Operators Web:7
- Arithmetic Operators Web:7
- Other Operators Web:8
- Special String Constants Web:8

Conditional Statements: `if/else` and `unless` Web:9
- `if` Web:9
- `unless` Web:9

Looping Web:10
- `for` Web:10
- `foreach` Web:10
- `while` Web:11
- `until` Web:11
- `last` and `next` Web:12
- `do ... while` and `do ... until` Web:12

Regular Expressions Web:12

Access to the Shell Web:13

Modules and CPAN Web:14

Code Examples Web:15
- Sending Mail Web:15
- Purging Logs Web:17
- Posting to Usenet Web:18
- One-Liners Web:19
- Command-Line Processing Web:19

References Web:20

43 Using Python Web:21

Python on Linux Web:22

The Basics of Python Web:23
- Numbers Web:23
- More on Strings Web:24
- Lists Web:27

Dictionaries Web:30
Conditionals and Looping Web:30
Functions Web:33
Object Orientation Web:34
Class and Object Variables Web:34
Constructors and Destructors Web:35
Class Inheritance Web:36
The Standard Library and the Python Package Index Web:38
References Web:38

44 Using PHP **Web:39**

Introduction to PHP Web:39
Entering and Exiting PHP Mode Web:40
Variables Web:40
Arrays Web:42
Constants Web:43
References Web:44
Comments Web:44
Escape Sequences Web:45
Variable Substitution Web:46
Operators Web:46
Conditional Statements Web:48
Special Operators Web:50
Switching Web:51
Loops Web:52
Including Other Files Web:54
Basic Functions Web:55
Strings Web:55
Arrays Web:58
Files Web:60
Miscellaneous Web:63
Handling HTML Forms Web:66
Databases Web:67
References Web:69

About the Author

Matthew Helmke has used Ubuntu since 2005. He has written about Linux for several magazines and websites, is a lead author of *The Official Ubuntu Book*, and has coauthored both *A Practical Guide to Linux: Commands, Editors, and Shell Programming* and *The VMware Cookbook*. In his day job, Matthew works for Gremlin (https://www.gremlin.com/) making the Internet more reliable. 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 teaches as an adjunct professor for the University of Arizona. You can find out more about Matthew at https://www.linkedin.com/in/matthewhelmke/ or drop him a line with errata or suggestions at matthew@matthewhelmke.com.

Dedication

To Sage, Sedona, and Philip—the most amazing kids a guy could hope for; to Sandra and Evan, who are wonderful and welcome additions to our lives; to my grandfather for always believing in me and teaching me to believe in myself; and to my friends in the Linux, developer, sysadmin, cloud computing, DevOps, Site Reliability Engineering, and Chaos Engineering communities.

Acknowledgments

I want to thank the many people who helped with past editions, helpful comments and ideas, and technical edits and both formal and informal advice. I owe a huge debt of gratitude to the Ubuntu community, Canonical, and Mark Shuttleworth for inviting me to participate in the community years ago. Finally, thanks to my colleagues at Pearson, especially Debra Williams, for the trust placed in me and the opportunity to collaborate on projects like this one.

Figure Credits

Figures 1-1 to 1-9, 4-7 to 4-9, 8-1, 9-1 to 9-3, 11-1, 15-1, 16-1, 17-2, 19-1, 19-2, 35-1, 35-3, 36-1: Screenshots of Ubuntu © 2020 Canonical Ltd

Figures 4-1 to 4-4: Screenshots of GNOME © The GNOME Project

Figure 4-5: Screenshot of Kubuntu © 2020 Canonical Ltd

Figure 4-6: Screenshot of Xubuntu © 2020 Canonical Ltd

Figure 5-1: Screenshot of Mozilla Firefox ©1998–2020 by individual mozilla.org contributors.

Figure 5-2: Screenshot of Google Chrome ©2020 Google LLC.

Figure 8-2: Screenshot of Warsow © 2020, Chasseur de Bots

Figure 8-3: Screenshot of Scorched 3D © 2020 Scorched3D

Figure 8-4: Screenshot of Frozen Bubble © Guillaume Cottenceau

Figure 8-5: Screenshot of Supertux © Pelya

Figure 8-6: Screenshot of Battle of Wesnoth © 2003–2020 by The Battle for Wesnoth Project

Figure 8-7: Screenshot of Frets on Fire offers flashy graphics and familiar game play © Unreal Voodoo

Figure 8-8: Screenshot of FlightGear features stunning landscapes and technically accurate control panels © Free Software Foundation

Figure 8-9: Screenshot of In Speed Dreams © Speed Dreams

Figure 12-1: Screenshot of Command Prompt © Microsoft 2020

Figures 23-1, 23-2: Screenshots of CUPS © 2020 Apple Inc

Figure 30-1: Screenshot of Oracle VM VirtualBox Manager © 2020 Oracle

Cover image: Widegren/Karlsson/Photodisc/Getty Images

This page intentionally left blank

Introduction

IN THIS INTRODUCTION

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

We are pleased to present the 2021 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 become a perennial favorite for 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.

Ubuntu has leveraged the quality of Debian to create an operating system that ordinary people can use. This 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. *Unleashed* books offer an in-depth look at their subjects, helping both beginner and advanced users move 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 the information in this book mirrors closely the development of the distribution.

A QUICK WORD ABOUT MARKETING

Almost all of the content in this book applies regardless of what Ubuntu release version you are using, as 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. These 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.

Do not let the highly technical reputation of Linux discourage you, however. Many people who have heard of Linux think that it is found only on servers, used to look after websites and email. But that is far from the truth. Distributions like Ubuntu are making huge inroads into 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 are improved, 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, so 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 that 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 open source software covered in 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.)

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 are licensed under other terms. Those packages are subject to a variety of software licenses, and many of them fall under a broad definition known as *open source*. Some of these licenses 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

This book varies in coverage from deep to shallow over a wide range of topics. This is intentional. Some topics are Ubuntu specific and are not covered by any other book, and they deserve deep coverage here. There are some topics that every power user really must master. Other topics are things power users should know about so that they understand some history, know some other options, or simply have what they need to be able to listen and participate in further discussions with other technical people without being completely confused.

Some topics, like using the Linux command line, receive deep and extensive coverage because we believe that information to be vital to anyone who wants to be a power user or become a skilled DevOps guru. That topic gets three full chapters.

Other topics get only brief coverage so that people who are interested get a few guideposts to help them continue if they are interested. For example, Chapter 39, "Using Popular Programming Languages," describes languages such as Ada and Fortran, along with more than 20 other programming languages. In that chapter, dozens of programming languages are covered in just a few pages. These are useful topics to some but not topics we would consider vital.

In addition, some topics are just too broad to be covered in great depth in this book, but they deserve mention because, again, an intermediate to advanced user should have at least a foundational knowledge of them. These are covered briefly and then information is provided to help you find more resources and expand your understanding, as needed.

Those Wanting to Become Intermediate or Advanced Users

Ubuntu Unleashed is intended for intermediate and advanced users or those who want to become intermediate and advanced users. 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 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 importantly, 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*. This concept describes how a person learns and masters a technique, passing through three levels:

- **Following—**The level where the learner looks for one very detailed process that works and sticks to it to accomplish a task.
- **Detaching—**The level where the learner feels comfortable with one method and begins to learn other ways to accomplish the same task.
- **Fluent—**The level 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.

We all start at the first level and progress from there. Few reach the last level, but those who do are incredibly effective and efficient. People aiming for this level are the very ones for whom we intend this book.

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 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, DevOps, and Site Reliability Engineering

System administrators, or sysadmins, are people who keep servers and networks up and running. Their role is sometimes called *operations*. They deal with software installation and configuration and security, and they 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 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. Site Reliability Engineering (SRE) began at a similar time as DevOps and from a distance looks almost the same. The major difference is that the SRE focus is solely on keeping sites and systems up and running, whereas DevOps has evolved a wider philosophy that may touch on people processes outside of the typical scope of SRE. This book is a great foundation for those who want to learn knowledge that will help with both roles and implementations, hopefully presented in a way that balances them nicely.

What This Book Contains

Ubuntu Unleashed is organized into six parts, described here:

- Part I, "Getting Started," 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 by using the development tools available.
- Part VI, "Bonus Chapters," consists of three online chapters on using the Perl, Python, and PHP languages on Ubuntu. See *informit.com/title*/9780136778851.

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 today 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 intermediate and advanced users. 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 a command or code listing example shows input that you as the user would type, 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 &
```

Other formatting techniques include the use of italic for placeholders in computer command syntax, as shown here:

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

Computer terms or concepts are also italicized upon first introduction in text.

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. A note augments a discussion with ancillary details or may point you to an article, a white paper, or another online reference for more information about a specific topic.

TIP

Tips contain special insights or time-saving techniques, 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 something is in a sidebar does not mean that you will not find something new here. Be sure to watch for these elements, which bring in outside content that is an aside to the discussion in the text. In sidebars, you will read about other technologies, Linux-based hardware, and special procedures to make your system more robust and efficient.

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.

This page intentionally left blank

CHAPTER 10 Command-Line Beginner's Class

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

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, console, command prompt, and *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.

Using 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. 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 11, "Command-Line Master Class, Part 1," advances the subject further and expands 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.

What Is the Command Line?

If you have spent any amount of time with experienced Linux users, you have heard them mention the command line. Some, especially those who began 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 were the most natural and obvious way to complete a task.

Embracing either extreme is not necessary. You might develop an affinity for the command line when performing some tasks and might prefer using 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, some good reasons exist 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 such 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 importantly, 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; 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 11, "Command-Line Master Class, Part 1," and Chapter 12, "Command-Line Master Class, Part 2."

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

- **Routine tasks**—Logging in and out, changing passwords, and listing and navigating file directories
- **Basic file management**—Creating files and folders, copying or moving them around the file system, and 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 and system administrators who are new to Linux and are learning to use the command line for the first time.

TIP

If you have used a computer for many years, you have probably 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 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 macOS, which provides access to the terminal.

Accessing the Command Line

You can quickly access the terminal by 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 4, "Ubuntu Desktop Options," as is logging in to a Linux system using a graphical interface. This chapter focuses on text-based logins and the use of Linux.

Text-Based Console Login

If you access the terminal as described previously, by opening the Terminal app, you will not need to log in because you are already logged in to your account.

If you are accessing the terminal remotely, such as from another computer using SSH (see Chapter 19, "Remote Access with SSH and VNC") you will start with a prompt similar to this one:

```
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? Security. This 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 says that the user is logged in as the user `matthew` on the system `seymour` and is currently in the home directory; Linux uses the tilde (~) as shorthand for the home directory, which would usually be something like `/home/matthew`.

Note that throughout this book we use the default shell for the command line, but other shells exist. The default and other shells are discussed in more detail in Chapter 14, "Automating Tasks and Shell Scripting."

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 that this command is `pwd` because it is an abbreviation of *print working directory*. You simply type `pwd` at any point to get the full path of your location. For example, typing `pwd` after following the preceding 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 of where you are.

Logging Out

Use the `exit` or `logout` command or press 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

You can happily log in on your computer, an act known as a *local* login. In addition, you can 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 18, "Networking," to see how to set up network interfaces with Linux to support remote network logins and Chapter 19 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 then 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 this section 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 system-wide changes that could affect other users.

To make system-wide 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 tasks. 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 would erase everything on your hard drive if it did not require appending `--no-preserve-root` to work—which was added specifically to prevent people from accidentally doing this. 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 `sudo` before a command while using a regular user account with super user privileges, you log in to the root account and 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 see a pound sign (`#`) in the command-line prompt in place of the dollar sign (`$`). For example, you see `matthew@seymour:~#` instead of the usual `matthew@seymour:~$` prompt.

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 other 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 every one of the hundreds of commands included with Linux has a man page; however, some do not have man pages or have only simple ones. You can 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 `apropros`

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
```

For the record, `apropos` is the equivalent of using `man -k`.

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 use 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 10.1 shows some of the top-level directories that are part of a standard Linux distro.

Table 10.1 Basic Linux Directories

Directory	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`	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, file systems that are additional to the standard, such as a DVD-ROM or another HDD
`/opt`	Add-on software packages
`/proc`	Kernel information, process control
`/root`	Super user (root) home
`/sbin`	System commands (mostly root only)
`/sys`	Real-time information on devices used by the kernel
`/tmp`	Temporary files
`/usr`	Software not essential for system operation, such as applications
`/var`	Variable files relating to services that run on the system, but whose contents are expected to change regularly during normal operation

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; you can choose to put certain directories on their own distinct partition, which can be useful for things like isolating directories from one another (for example, for server security you might put a directory like `/boot` that doesn't change often on its own partition and make it read-only and unchangeable without specific operations being done by a super user during a maintenance cycle). Desktop users probably don't need to think about such operations, but knowing the directory tree is still quite useful when you want to find the configuration file for a specific program and set some program options system-wide to affect all users.

NOTE

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

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

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

More information about file systems is available in Chapter 21, "Performance Tuning."

Some of the important directories in Table 10.1, 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; it makes the system what it is, and changing or removing this software could cause instability or a complete system failure. Often, the commands in these two directories are *statically* linked, which means 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 so that changes are more difficult and so 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 that lists each hard drive, CD-ROM, 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 22, "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 system startup.
- **`passwd`**—This file holds the list of users for the system, including special-purpose nonhuman users like syslog and CouchDB, along with user account information.
- **`sudoers`**—This file holds 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 you have an account named `matthew`, your home directory would generally be found in `/home/matthew`. This can be changed, as described in Chapter 11.

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 or Obtain Information from the Kernel

The contents of the `/proc` directory are created from memory and exist 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 in `/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. Files in this directory are cleared daily by a `cron` job and every time the system is booted.

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`.

NOTE

Linux also uses `/var` for other important system services, such as the Apache web server's initial home page directory for the system, `/var/www/html`. (See Chapter 25, "Apache Web Server Management," for more information about using Apache.)

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, and 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, the `ls` command lists the files and directories in your 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 you know what you are looking for but cannot remember the exact name. However, using `ls` in this matter 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 (discussed later in this chapter), 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 2015-06-04 08:20 .
drwxr-xr-x  3 root    root      4096 2015-05-16 16:48 ..
-rw-------  1 matthew matthew    421 2015-06-04 10:27 .bash_history
```

```
-rw-r--r--  1 matthew matthew    220 2015-05-16 16:48 .bash_logout
-rw-r--r--  1 matthew matthew   3353 2015-05-16 16:48 .bashrc
drwxr-xr-x 13 matthew matthew   4096 2015-05-21 10:42 .config
drwxr-xr-x  2 matthew matthew   4096 2015-05-16 17:07 Documents
-rw-r--r--  1 matthew matthew   335  2015-05-16 16:48 file.txt
drwxr-xr-x  3 matthew matthew   4096 2015-05-16 17:07 .local
drwxr-xr-x  2 matthew matthew   4096 2015-05-16 17:07 Music
drwxr-xr-x  3 matthew matthew   4096 2015-05-16 18:07 Pictures
-rw-r--r--  1 matthew matthew    675 2015-05-16 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 subdirectories are included; remember that the listing here is abbreviated.) Then, the details are as shown in Figure 10.1.

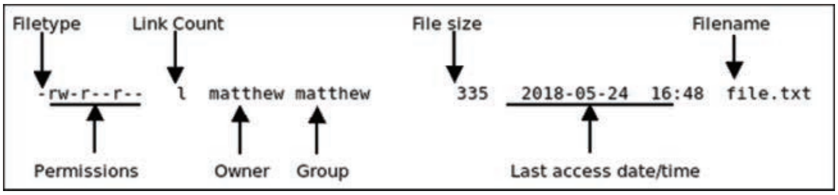


FIGURE 10.1 Decoding the output of a detailed directory listing.

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 14. In the meantime, note that you can read the contents of the text file by 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 as follows:

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

This command 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 12. To return to your home directory, type this command and press Enter:

```
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 determine you where you are within the file system:

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

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 2015-06-30 13:06 file
```

In this example, the `touch` command quickly creates 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`), `l` indicates a symbolic link, and `b` is used for a block device (such as `/dev/sda`).
- **Permissions**—Read, write, and execute permissions may be assigned for the owner, group, and all others on the system. (You learn more about these permissions later in this section.) Permissions are traditionally called the *mode*, which is where the chmod command we will discuss later comes from.
- **Number of hard links to the file**—The number `1` designates that there is only one file, and 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 a pointer to the original file, which might be located elsewhere on the system. Only the root operator can create a hard link of a directory.
- **The owner**—This is the account that owns the file; it is originally the file creator, but you can change this designation by using the `chown` command.
- **The group**—This is the group of users allowed to access the file; it is originally the file creator's main group, but you can change this designation by 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
rwx      rwx      rxw
```

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

- `r` indicates permission for an owner, a member of the owner's group, or others to open and read the file.

- w indicates permission for an owner, a member of the owner's group, or others to open and write to the file.
- x indicates permission for an owner, a 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 when 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 2015-06-30 13:23 directory
```

In this example, the `mkdir` command is used to create a directory. The `ls` command, with 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 see the following:

```
matthew@seymour:~$ ls -l /dev/ttyS0
crw-rw---- 1 root dialout 4, 64 2015-06-30 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 2015-06-30 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`, covered in the next section):

```
-rw-r--r-- 1 matthew matthew 0 2015-06-30 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 2015-06-30 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 2015-06-30 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 2015-06-30 13:33 readme.txt
```

If you take away execution permission for a directory, files will 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 `umask`

When you create a file, it is created with a default set of permissions, `666`. Directories have a default set of permissions, `777`. You can view and modify the default permissions for either with `umask`, which works like a filter. When a file is created by a user account, whether that account is owned by a human like `matthew` or a process like `init`, the file will be created using specific permissions.

The numbers we used earlier when discussing file permissions are also used with `umask`, but with an interesting change. Now, the numbers defined in `umask` are subtracted from the ultimate file permissions. So, if you wanted all new directories to be created with a default permission of `777`, you would type this:

```
matthew@seymour:~$ umask 000
```

Of course, you would never want to have all your directories or files accessible by default because that would be incredibly insecure and unsafe. The default `umask` is `022`, which means that files are created by default with `644` permissions, except in the `/home` directory (for all user directories under it) where the umask is `002` and files are created with `775`. Note: The umask is a file system–wide variable for each user session, so you cannot have a different setting for specific directories, such as one for `/home` and another for `/home/seymour`.

To find the current umask setting, use this:

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

This may list four digits instead of three. If so, don't be confused. The additional digit is the first one; it is explained later in this chapter, in the section "Understanding Set User ID, Set Group ID, and Sticky Bit Permissions."

To change the `umask` setting—for example, if you wanted the default for files to be `640`—use the following:

```
matthew@seymour:~$ umask 037
```

File Permissions with `chgrp`

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

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

Understanding Set User ID, Set Group ID, and Sticky Bit Permissions

The first two of the three listed 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 2015-01-26 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`.

By default, suid and sgid are turned off on files. To set them, add an extra digit to the beginning of a number in a `chmod` command. Suid uses `4`. Sgid uses `2`. You can set both at the same time by using `6` (`4 + 2`). For example, for a file owned by root with current `711` permissions allowing anyone to run it, you can make it run as root with the following:

```
matthew@seymour:~$ chmod 4711 filename
```

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.

Sgid has an additional function when set on directories; in this case, new directory contents are automatically assigned the group owner on the directory.

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 with 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 a 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 a program's execute permission for anyone. As the root operator, you have to decide 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.

An additional setting called the *sticky bit* is available using this same additional first digit. A sticky bit limits who may rename or delete files within a directory. When it is set, files in that directory may be unlinked or renamed only by a super user, the directory owner, or the file owner. Set the sticky bit to on by using a `1`, like this for a directory with `755` permissions:

```
matthew@seymour:~$ chmod 1755 directoryname
```

You can set the sticky bit concurrently with suid and sgid, like this (`4 + 2 + 1`):

```
matthew@seymour:~$ chmod 7755 directoryname
```

Setting Permissions with Access Control Lists

POSIX is a family of standards created to maintain stability and consistency across operating systems for UNIX and UNIX-like systems, such as Linux. One important feature of POSIX is the access control list (ACL; often pronounced “AK-el”). ACLs permit even more fine-grained control over access permissions.

By default, all files have an ACL. To view the ACL for a file, use this:

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

Typical `getfacl` output includes multiple lines, like this for `filename.txt:`

```
# file: filename.txt
# owner: matthew
# group: matthew
user::rw-
group::rw-
other::r--
```

The information listed here is standard and clear, based on what you already know. The real power of ACLs is that you can add to them. You are not restricted to the standard set of user, group, other. You can add multiple users and groups with permissions specific to each.

To add the user `sandra` with read, write, and execute permissions to the ACL for a file named `secrets.txt`, use the following:

```
matthew@seymour:~$ setfacl -m u:sandra:rwx secrets.txt
```

To remove and reset `sandra's` permissions on the file to the file’s defaults, use the following:

```
matthew@seymour:~$ setfacl -x u:sandra: secrets.txt
```

From these two examples, you can see that `-m` is for modify and `-r` is for remove.

ACLs permit similar actions with groups and others as with a user. Instead of the `u:` before the name, use a `g:` for groups and an `o:` for others, like this:

```
matthew@seymour:~$ setfacl -m g:groupname:rwx secrets.txt
matthew@seymour:~$ setfacl -m o:r secrets.txt
```

Notice that with others, there is no username or group name to include in the commands.

A useful feature is masking, which allows you to list only the permissions that are available, as in this example:

```
matthew@seymour:~$ setfacl -m m:rx secrets.txt
```

This limits everyone, regardless of any other settings. So, in this case, a group may have `rwx` settings on the file, but the mask here says to permit only `rx`, so `rx` will be the only settings that are available.

As an exercise, see if you can figure out the meaning of this output from `getfacl` for a file named `coffeecup.conf`:

```
# file: coffeecup.conf
# owner: matthew
# group: yirgacheffe
user::rw-
group::rw-
other::r--
group:qa:rwx
group:uat:rwx
mask::rwx
```

Working with Files

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

Creating a File with `touch`

If you are the user `matthew`, to create an empty file called `myfile` within your current directory, use the following command:

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

To edit this file, you must use a text editor. (Several text editors are discussed in Chapter 14. However, it is sometimes useful to create an empty file as doing so also 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 you are already in your home directory, you can use the following:

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

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

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

Or from anywhere using a path shortcut, you can 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
```

If you are the user `matthew`, 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 you are already in your `/home` directory, you can use the following:

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

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

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

Or from anywhere using a path shortcut, you 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 you 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`

If you are the user `matthew`, 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 you are already in your `/home` directory, you can use the following:

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

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

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

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

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

The directory must be empty to be removed using `rmdir`. However, you can remove a directory with its contents by 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`

If you are the user `matthew`, to delete a file named `filename`, use this command:

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

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

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

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

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

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

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

If you try to use `rm` to remove an empty directory, you 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 by 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 to ensure 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 lets 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 (which works with many commands, not only `rm)`:

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

Everything in `randomdirectory` as well as the directory itself will be deleted, including other subdirectories, without considering whether they are empty or have contents.

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.

Assuming that you are the user `matthew`, to rename a file that is in your current directory, you could use the following:

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

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

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

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

```
matthew@seymour:~$ mv /home/matthew/documents/oldfilename
[ccc]/home/matthew/archive/newfilename
```

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

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

Copying a File with `cp`

Copying works similarly to moving, but it retains the original in the original location. Assuming that you are the user `matthew`, 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 your current directory, you 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 you copy it from `~/documents` to `~/archive`, you could use this:

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

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

```
matthew@seymour:~$ cp /home/matthew/documents/oldfilename
[ccc]/home/matthew/archive/newfilename
```

Or from anywhere using a path shortcut, you could 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, assuming that you are the user `matthew`, 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 12. This command works best with short files because the contents of longer files scroll off 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
```

In the early days of UNIX, a program called `more` gave paged output. 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`."

Using Wildcards and Regular Expressions

Each of the commands in the previous sections 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 11 and in Chapter 14.

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 11, this mention is 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 / --no-preserve-root
```

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 execute single commands as root and then immediately 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 wrapping.

Understanding and Fixing `sudo`

Most Ubuntu users never have problems with `sudo`, but sometimes, people who like to experiment break things, especially while learning. This section helps you 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 usually can tell that a problem has occurred because an error message like this appears 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 by using the following knowledge and procedure.

In order for a user to use `sudo`, the user account must belong to the *sudo* group and must 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` in several situations:

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

These problems generally result from users 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, the account does not have access to root. To fix it, you 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, and have several options, select the menu entry for root: Drop to Root Shell Prompt. You are now at the command line, with full root access to the computer.
4. Because Ubuntu mounts file systems as read-only by default in recovery mode, you need to remount the root file system, `/`, 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 and knows what he or she is doing, that person can easily and quickly gain full control over the machine and all it contains.

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

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

If the problem exists because the permissions for `/etc/sudoers` are wrong, enter this:

```
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 12, that runs a check on the file after editing to be certain 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 as follows:

```
#
# 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:

#includedir /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 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 is created in `/etc/passwd` for the user. The system also creates a directory, labeled with the user's username, in the `/home` directory. For example, if you create a user named `sandra`, the user's home directory is `/home/sandra`.

NOTE

In this chapter, you learn how to manage users from the command line. See Chapter 13, "Managing Users," for more information on user administration, including administration using graphical administration utilities.

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

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

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

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

Changing password for user sandra.
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 13 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` options 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 `-remove-home` 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`:

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

You can also follow `-h` with the numeral zero (`0`), like this, to get the same effect:

```
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 multiuser systems.

It used to be that to create a custom MOTD file, 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 the MOTD file by using `cron`. To modify how the MOTD file 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 (as with 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 the intruder by using computer forensics, but rather plan to merely wipe everything clean and rebuild or reinstall the system from scratch.

Rebooting the System

You should 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`:

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

You can also follow `-r` with the numeral zero (`0`), like this, to get the same effect:

```
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`, `poweroff`, and `reboot` commands, but the `shutdown` command is more flexible.

Commonly Used Commands and Programs

A number of programs and built-in shell commands are commonly used when working at the command line. These commands are organized here by category to help you understand the purpose of each category:

- **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`

If you need to find full information for using the command, you can find that information under the command's man page.

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

& operator, 237
    background processes, running, 173–174
    special shell character, 247
    VirtualBox commands, 527
&& operators, combining commands, 180, 198
* special shell character, 247
\ special shell character, 249–250
` special shell character, 247, 250
[ ] special shell character, 248
[a, z] special shell character, 248
[a-z] special shell character, 248
[az] special shell character, 248
$, 235
$? built-in variable, 247
$ special shell character, 247
$# built-in variable, 246
$* built-in variable, 247
$0 built-in variable, 247
! ! operator, 196–197
!= operator, 491
< special shell character, 247
<< operator, 236
<< special shell character, 248
> special shell character, 247
>> special shell character, 247
| (pipe) special shell character, 247
| | operator, 198, 491
# special shell character, 247
? special shell character, 247

" special shell character, 248–249
' special shell character, 249

Numbers

10BASE-T networking, 339
10G Ethernet networking, 340
32-bit Ubuntu, 64-bit Ubuntu versus, 4–5
50G Ethernet networking, 340
64-bit Ubuntu, 32-bit Ubuntu versus, 4–5
100BASE-T networking, 339
1000BASE-T networking, 339

A

A records, 590
AAAA records, 590
ac command, 211, 223
accessing
  ACL
    setting permissions, 127–128
    Squid, 565–569
  Apache web server, 452, 455
  command line, 107
  databases
    local GUI clients, 503
    PostgreSQL, 501–503
    SSH, 501–502
    web browsers, 503
  file access time, disabling, 397
  networks, chains, 387
  remote access
    copying files between machines, 372–373
    Guacamole, 377
    key-based logins, 373–375
    online references, 377–378
    SSH server, 371–375
    VNC, 375–377
  shell scripts, storing for system-wide access, 240
  variable values, 243
accounting, user usage statistics, 212
ACID compliance, relational databases, 493–494
ACL
  permissions, setting, 127–128
  Squid, 565–569
activity (user), monitoring, 211–212
Ada, 622
addressing
  broadcasting, 338
  IPv4 addressing, 331–332
  IPv6 addressing, 332, 334–336
  multicasting, 338
  NAT, 332
  TCP/IP, 331–332
  unicast addressing, 338
adduser command, 136–137, 209
administration
  LDAP, 584–585
  privileges, 217
Adobe Photoshop, 67
afio, 316
all-in-one (Print/Fax/Scan) devices, 435
AllowOverrides directives (Apache web server), 451–452
ALSA, 64
Amanda, 315–316
AND, 299
AND operators, 491–492
Android mobile development, 607
  Android Runtime, 608
  Android Studio, 609

Application Framework, 608–609
applications, 609
ARM processors, 608
Google Play, 607
Linux kernels, 608
RISC processors, 608
Ansible, 544
Apache Tomcat, 482
Apache web server
access control, 452, 455
configuring
.htaccess configuration files, 450–452, 469
runtime servers, 446–449
threads, 468
development of, 443–444
file system authentication, 452–455
.htaccess configuration files, 450, 469
AllowOverrides directives, 451–452
Options directives, 451
HTTPS, 464–466
installing, 444
logging, 463–464
modules, 455–461
MPM, 449–450
online references, 466
overview of, 444
Require directive, 452
runtime servers
apache2.conf files, 447–449
configuring, 446–449
DirectoryIndex directive, 449
DocumentRoot directive, 449
GID, 448
Group directive, 448
Listen directive, 447
ServerAdmin directive, 448
ServerName directive, 448–449
ServerRoot directive, 447
UID, 448
User directive, 448
UserDir directive, 449
security, 444, 452–455
starting/stopping, 444–446
threads, 468
virtual hosting, 461–462
apache2.conf files, 447–449
API, 512
AppArmor, 388–390
Application Framework, 608–609
applications
Android applications, 609
AppArmor, 388–390
CD/DVD burning applications
Brasero, 71
Linux command line, CD creation, 72–73
Linux command line, DVD creation, 73–75
Ubuntu support, 71
DVD/CD burning applications
Brasero, 71
Linux command line, CD creation, 72–73
Linux command line, DVD creation, 73–75
Ubuntu support, 71
games, 79–81
Battle for Wesnoth, 85–86
commercial games, 88
documentation, 89
emulators, 79
FlightGear, 87
Frets on Fire, 86
Frozen Bubble, 84
Game Jolt, 82
GOG.com, 82
Humble, 82

installing from Ubuntu repositories, 82–87
installing video drivers (proprietary), 80
itch.io, 82
kid-friendly games, 88
LGDB, 82
online game sources, 81–82
Scorched 3D, 83–84
Speed Dreams, 87
SuperTux, 84–85
Warsow, 82
Windows games, 88–89
graphics manipulation applications, 66, 70
Blender, 70, 77
CinePaint, 70
darktable, 70
digiKam, 70
GIMP, 66–67
Hugin, 70
Inkscape, 70
Krita, 70
nautilus-image converter, 69
netpbm tools, 69
Photoshop, 67
POV-Ray, 70
Radiance, 70
Shotwell Photo Manager, 66, 71
Simple Scan, 67
Xara Xtreme, 70
microservice architectures, 531
multimedia applications
ALSA, 64
CD/DVD burning applications, 71–75
graphics manipulation applications, 66–71
music applications, 65–66
online references, 77
OSS, 64
PulseAudio, 64
sound cards, 63–64
music applications, 65
Banshee, 66
Rhythmbox, 66
Sound Juicer, 66
productivity applications
Celtx, 59
CrossOver Office, 61
defined, 56
gedit, 59
Heimer, 61
Kile, 61
LaTeX, 60
LibreOffice, 55, 56–58
LyX, 60
online references, 61
pdfedit, 58
Publican, 59–60
Texmaker, 60
users (typical), 56
Windows, 61
Wine, 61
XML Copy Editor, 60
recording/editing sound, 76
security, 388–390
Startup Applications Preferences, 278
VBA, LibreOffice and Office compatibility, 55
video editing applications, 77
Avidemux, 77
Blender, 77
Cinelerra, 77
DaVinci Resolve, 77
Kdenlive, 77
Lightworks, 77
OpenShot Video Editor, 77
PiTiVi, 77
Shotcut, 77

apropos command, 112
APT, 95
  apt-get versus, 99–100
  day-to-day usage, 95–98
  finding software, 98–99
  VirtualBox, 527
apt-get, APT versus, 99–100
archives, restoring files from, 311–312
Ardour, 76
arguments (positional), 242
ARM processors, 608
arrays (RAID), 307
Art of Unix Programming, The, 141–142
ash, 226
Aslett, Matt, 511
assessing
  needs/resources, backup strategies, 303–304
  vulnerabilities, 381–382
assigning
  permissions, file systems, 121–122
  values to variables, 242
at command, 227–228, 229
atomic changes, 613
attacks, 380–381
  autocracking scripts, 380
  crackers, 380
  external attacks, 379–380
  hackers, 380
  internal attacks, 379–380
  script kiddies, 380
  spoofing attacks, 593
  viruses, 385
  war driving, 383
  worms, 380
Audacity, 76
Audio CD Extractor. *See* Sound Juicer
authentication
  file systems, Apache web server, 452–455
  PAM, 215–216
autoconf command, 601–602
autocracking scripts, 380
automatically waking computers from sleep, 231–233
automating tasks with scripts, 244–246
availability
  localhost interfaces, checking availability of, 326
  memory space, displaying, 286
.avi files, 75
Avidemux, 77
awk text editor, 189–191
AWS, 539
Azure, 539

B

Back In Time, 314
background jobs, moving jobs to, 174–175
background processes, 173–174 237
  running,
backups
  afio, 316
  Amanda, 315–316
  Back In Time, 314
  cloud computing, 305–306, 309
  configuration files, version control, 320–322
  copying files, 316–317
    cp command, 318
    rsync command, 319–320
    tar command, 317–318
  data loss, 302
  Déjà Dup, 313–314
  external hard drives, 308

File Roller, 312
flexbackup, 316
full backups
with incremental backups, 307
on a periodic basis, 306–307
tar command, 310–311
incremental backups, 310–311
kdat, 312–313
KDE ark archiving tool, 312–313
levels of, 306
mirroring data, 307
NAS, 308
network storage, 308
online references, 324
restoring files from archives, 311–312
scheduling tasks, full backups on a periodic basis, 306–307
software, 309–316
strategies, 301
assessing needs/resources, 303–304
choosing a strategy, 308
choosing hardware/media, 308–309
evaluating, 304–306
full backups on a periodic basis, 306–307
full backups with incremental backups, 307
home users, 304–305
incremental backups, 307
inheriting, 306
large enterprises, 305
RAID arrays, 307
simple backup strategies, 306
small enterprises, 305
small offices, 305
system rescue, 323
restoring GRUB2 boot loaders, 323–324
saving files from nonbooting hard disks, 324
Ubuntu Rescue disc, 323
tape drive backups, 309
tar command, 310
copying files, 317–318
incremental backups, 310–311
restoring files from archives, 311–312
Ubuntu installations, 7
Unison, 315
version control, configuration files, 320–322
badblocks command, 397
Banshee music application, 66
BaseX, 517
bash, 226
comparisons of expressions, 250–251
file operators, 253–254
logical operators, 255
numeric comparisons, 252–253
strings, 251–252
local bash shells, configuring, 640
batch command, 228–229
batches, changing passwords in, 216–217
Battle for Wesnoth, 85–86
Bazaar
configuring, 639
software project management, 611–612
Berkeley DB, 512, 579
bg command, 174–175
BigTable, 517
Bikeshed, 616–618
/bin directory, 114
BIND
configuring DNS servers, 593–595
online references, 595
BIOS, 271
boot process, 272–274
Ubuntu bootups, troubleshooting, 195
Blender
graphics manipulation, 70
video editing, 77

.bmp files, 68
BOFH, 205
boolean operators, combining commands, 180
boot loaders, 271
  boot process, 272–274
  GRUB2, installing, 5
  GRUB2 boot loaders, restoring, 323–324
  Ubuntu installations, 2–3, 5–6
boot process
  BIOS, 271, 272–274
  boot loaders, 271, 272–274
  Boot-Repair, 278–279
  DHCP activation, 357
  GPT, 274
  init systems, 272
  loading process, 272–274
  MBR, 274
  NetBoot, 274
  nonbooting hard disks, saving files from, 324
  PXE, 274
  runlevels, 271–272, 275, 278
  services
    controlling, 278
    running, 271–272
    starting/stopping, 275–278
    Startup Applications Preferences, 278
  troubleshooting, 278–279
  Ubuntu, troubleshooting, 195–196
  UEFI, 273–274
Brasero, 71
break statements, 269
bridges, 343, 524–525
broadcasting, 338
browsers (web)
  Chrome, 53–54
  Chromium, 53–54
  database access, 503
  Firefox, 52–53
  online references, 54
brute-forcing, 373–374
Budgie, 49
bugs, fixing
  Bug Squad, 647
  Ubuntu development, 640–643
built-in variables, 242, 246–247
bundles (software), Snappy package manager, 103–104
bunzip2 command, 191
burning CD/DVD
  Brasero, 71
  Linux command line
    CD creation, 72–73
    DVD creation, 73–75
  Ubuntu support, 71
business applications, Ubuntu for, 27–28
byobu, multiple terminal operation, 192–193
bzip2 command, 191

C

C programming, 597–599
  autoconf command, 601–602
  building programs, 599–601
  configuring code, 601–602
  debugging, 602
    checking source code, 602–603
    gdb command, 603
    gprof command, 603
    splint command, 602–603
    symbolic debugging, 603
    tracking function time, 603
  dependency checking, 599–600
  GCC, 603–604
  macros, 600–601

makefile targets, 600–601
makefiles, 599–600
C++ programming, 597–598, 599
autoconf command, 601–602
building programs, 599–601
configuring code, 601–602
debugging, 602
checking source code, 602–603
gdb command, 603
gprof command, 603
splint command, 602–603
symbolic debugging, 603
tracking function time, 603
dependency checking, 599–600
GCC, 603–604
macros, 600–601
makefile targets, 600–601
makefiles, 599–600
C10K problem, 467–468
cabling, 340
fiber-optic cabling, 342
UTP cabling, 341
Caddy, 482
cameras (digital)
Ubuntu support, 70
USB connections, 70
capturing screen images, 69
case statements, 267–268
Cassandra, 513
cat command, 132, 144–145
/cd command, 120
cd command, 145–147
CD tools, 613–614
CD/DVD burning applications
Brasero, 71
Linux command line
CD creation, 72–73
DVD creation, 73–75
Ubuntu support, 71
cdrecord command, 72–73
Cecilia, 76
Celtx, 59
CFEngine, 545
cgroups, 532
chains, 387
change command, 223
changing
desktop environments, 39
directories, 120, 145–147
passwords in batches, 216–217
permissions, files/folders, 125, 147
shells, 226
user identities, 217–219
window managers, 39
Chaos Engineering, 614
Chef, 544
Cherokee, 480–481
chfn command, 224
chgrp command, 125, 204, 224
children directories, 118
children, games for, 88
chmod command, 123–124, 147, 204, 224
chown command, 125, 204, 224
chpasswd command, 224
Chrome, 53–54
Chromium, 53–54
chroot jail, 531–532
chsh command, 209, 224
CI/CD tools, 613–614
Cinelerra, 77
CinePaint, 70
CLI. *See* command line
client/server database model, 484
Clojure, 622–623
cloud computing
AWS, 539
Azure, 539
backups, 305–306, 309

containers, 531
DevOps, 535–536
Google Cloud, 539
hybrid clouds, 539–540
IaaS, 537
Juju, 540–541, 545
Landscape, 541, 545
MaaS, 537
Mojo, 541
online references, 541
OpenStack, 538–539
PaaS, 537
private clouds, 540
public clouds, 538, 539
reasons for using, 536–537
SaaS, 537
Sysadmin, 535–536
Ubuntu, 535
    AWS, 539
    Azure, 539
    considerations for using, 538
    Google Cloud, 539
    hybrid clouds, 539–540
    IaaS, 537
    install instructions, 536
    Juju, 540–541, 545
    Landscape, 541
    MaaS, 537
    Mojo, 541
    OpenStack, 538–539
    PaaS, 537
    private clouds, 540
    public clouds, 538, 539
    reasons for using, 536–537
    SaaS, 537
    Ubuntu Cloud, 536
    VM, 536
VM, 536
CNAME records, 590–591
COBOL, 623–624
code packaging, Ubuntu development, 640–643
combining commands
    && operators, 180, 198
    ! ! operator, 196–197
    | | operator, 198
    boolean operators, 142–143
    piping, 142–143, 178–180
comm command, 170–171
command line (Linux)
    ! ! operator, 196–197
    | | operator, 198
    accessing, 107
    apropos command, 112
    APT, 95
        apt-get versus, 99–100
        day-to-day usage, 95–98
        finding software, 98–99
    background jobs, 174–175
    background processes, running, 173–174
    basic commands, 143–144
    BIOS, troubleshooting Ubuntu bootups, 195
    byobu, multiple terminal operation, 192–193
    CD creation, 72–73
    combining commands
        && operators, 180, 198
        ! ! operator, 196–197
        | | operator, 198
        boolean operators, 180
        piping, 178–180
    command history, viewing, 197
    commands
        combining, 178–180, 196–197, 198
        list of, 143–144
        redirecting input/output of, 167–170
        shortcuts, 198
        standard input/output, 169

commonly used commands/programs, 139
Coreutils, 199
defined, 106–107
directories, 112–114
  /bin directory, 114
  changing, 120, 145–147
  children directories, 118
  confining scripts to directories, 198–199
  copying files, 131, 147
  creating, 129, 160
  creating files, 128–129
  deleting, 129–131, 161–162
  displaying contents of files, 132
  /etc directory, 114
  finding current directory, 120
  finding files, 147
  /home directory, 115
  listing contents of, 118–119
  listing files in directories, 156–158
  moving files, 131, 161
  parent directories, 118
  permissions, 122–127
  printing directory sizes, 148
  printing last lines of files, 163–164
  /proc directory, 115–117
  regular expressions, 133
  renaming files, 131, 161
  /sbin directory, 114
  sorting file contents, 162–163
  /tmp directories, 117
  /usr directory, 117
  /var directory, 117
  wildcard characters, 133
documentation
  manual pages, 111–112
  reading, 111–112
DVD creation, 73–74
  packet writing, 74–75
  session writing, 74
environment variables, 182–185
file systems
  directories, 112–117
  hierarchy of, 112–114
files/folders
  comparisons, 170
  compressed files, 191–192
  creating links, 154–156
  downloading, 164
  finding differences in files, 170
  finding files, 149–151
  finding from indexes, 156
  finding similarities in files, 170–171
  listing in directories, 156–158
  listing system information, 158–159
  printing contents of, 144–145
foreground jobs, 174–175
GRUB, troubleshooting Ubuntu bootups, 195–196
input string searches, 151–152
jobs
  executing in parallel, 181–182
  listing, 173
kernel ring buffers, reading contents of, 200
location of commands, printing, 164
logging out of, 108
logins
  from remote computers, 108–109
  text-based logins, 107–108
manual pages, 159–160
multiple terminal operation, 192–193
MySQL command line client, 504–505
network interface configuration
  ifconfig command, 346–348
  ip command, 348–349

ip route command, 346, 349
netstat command, 346, 350
online references, 139, 165, 200
paging through output, 152–154
piping commands, 178–180
polite system resets, 194–195
PostgreSQL command line client, 505–506
printing, resource usage, 175–177
processes
listing, 171–173
prioritizing, 177–178
substituting, 181
reasons for using, 142–143
REISUB, 194–195
repeating text, 148–149
rerunning previous commands, 196–197
resource usage, printing, 175–177
root accounts, 110–111, 133–134
sequence, running commands in, 180–181
sudo command, troubleshooting, 134–136
systems
rebooting, 138–139
shutting down, 137–138
text editors, 185–191
Ubuntu bootups, troubleshooting, 195
user accounts
creating, 136–137
deleting, 137
super user accounts, 109–111
whereis command, 112
command line (shells), 233–234
background processes, 237
pattern-matching, 235
piping commands, 237
commands
& operator, running background processes, 173–174
&& operators, 180, 198
! ! operator, 196–197
| | operator, 198
database commands, 506
history of commands, viewing, 197
piping, 142–143, 178–180, 237
printing location of, 164
redirecting input/output, 167–169
redirecting input/output with shell command line, 236
rerunning previous commands, 196–197
running, in sequence, 180–181
shortcuts, 198
substituting, 250
wildcard characters, 235
commercial games, 88
community teams, testing Ubuntu, 645–646
comparing files/folders, 170
comparisons of expressions, 250–251
file operators, 253–254, 257–258
logical operators, 255, 259
numeric comparisons, 252–253, 256–257
strings, 251–252, 255–256
compatibility
free/open-source software, MS Office compatibility, 55
LibreOffice, Office compatibility, 55
Office, free/open-source software compatibility, 55
open-source/free software, MS Office compatibility, 55
Ubuntu installations, Mac hardware compatibility, 3
compiling
kernels, 412–414, 418–419
software
from source, 100, 101–102
from tarballs, 100–101
from Ubuntu repositories, 101–102
compressed files, 191–192, 318

computers (remote), command line logins, 108–109
configuration files
  Dotfiles, 322
  version control, 320–322
configuring
  Apache web server
    .htaccess configuration files, 450–452, 469
    runtime servers, 446–449
    threads, 468
  bash shells, 640
  Bazaar, 639
  C/C++ code, 601–602
  configuration management, 102
    dotdee, 102–103
    Snappy package manager, 103–104
    Ubuntu Core, 103
  DHCP software, 358–359
  dial-up Internet access, 367–368
  disk quotas, 223
  DNS servers with BIND, 593–595
  DSL Internet access, 365
  /etc directory configuration files, 114
  firewalls, 386–388
  GPG keys, 637–639
  hosts (networks) with DHCP, 359–361
  Internet connections, common configuration information, 364–365
  kernels, 403, 414–417
  Launchpad, work environments, 638–640
  LDAP servers, 580
  local bash shells, 640
  localhost interfaces, 327
  MySQL, 494–495
  networks
    command line network interfaces, 345–350
    configuration files, 350–355
    DHCP, 355–361
    /etc/host.conf files, 355
    /etc/hosts files, 350–351
    /etc/netplan/*.yaml files, 353–355
    /etc/nsswitch.conf files, 351–352, 355
    /etc/resolv.conf files, 352–353
    graphical configuration tools, 355
    nm-connection-editor, 345–346
  NFS
    clients, 423–424
    servers, 422–423
  pbuilder, 638
  PostgreSQL, 498
  power management, in Ubuntu, 18–19
  PPP Internet access, 364
  PPPoE, 364
  printers, in Ubuntu, 18
  runtime servers, Apache web server, 446–449
  Samba, 425
    /etc/samba/smb.conf files, 426–429
    global behaviors, 427
  security, 384–385
  software repositories, 15–17
  Squid, 570–572
  Squid clients, 564–565
  SSH server, 371–372
  system settings, in Ubuntu, 17–18
  Ubuntu
    date/time settings, 19–20
    development packages, 637–638, 640–643
    power management, 18–19
    printer configurations, 18
    software repositories, 15–17
    system settings, 17–18
    troubleshooting post-configuration problems, 21–22
    wireless networks, 20

VPN
clients, 573–575
servers, 575–577
wireless networks, 20
confining scripts to directories, 198–199
Conky, 292–297
connectivity
Internet, 363–364
checking, 328–330
common configuration information, 364–365
dial-up access, 367–368
DSL access, 365
ISP, 364–365
Linux commands, 369
online references, 369–370
PPP access, 364
PPPoE, 364
troubleshooting, 368–369
Ubuntu commands, 369
VM, 526
console-based monitoring, 281–283
containers, 531
Docker, 533
Kubernetes, 534
LXC, 532
LXD, 533
online references, 534
contents of directories, listing, 118–119
contents of files
displaying, 132
sorting, 162–163
converting
graphics, 68–69
sound files, 65
copying
files between machines (remote access), 372–373
files/folders, 131, 147, 316–317. *See also* mirroring
cp command, 318
rsync command, 319–320
tar command, 317–318
VM, 526
Coreutils, 199
CouchDB, 515–516
cp command, 131, 147, 318
crackers, 380
createdb command, 506
createuser command, 506
cron command, 229–231
CrossOver Office, 61
cups cancel command, 435
cups command, 435
cups disable command, 435
cups enable command, 435
CUPS GUI, network printers, 431–434
cuspreject command, 435
custom kernels, 402

D

D programming language, 624
daemons
AND, 299
Linux, 25
smbd, starting/stopping, 429
darktable, 70
Dart, 624
dash, 226
Dash (GNOME), 43–44
data, SQL tables
inserting data, 489–490
retrieving data, 490–492

data directory initialization, PostgreSQL, 498
data integrity, relational databases, 493–494
data locking, relational databases, 492
data loss, 302
data mirroring, 307
data recovery, 323
    GRUB2 boot loaders, restoring, 323–324
    saving files from nonbooting hard disks, 324
    Ubuntu Rescue disc, 323
databases
    accessing
        local GUI clients, 503
        MySQL, 501–503
        SSH, 501–502
        web browsers, 503
    AND operators, 491–492
    client/server model, 484
    commands, 506
    flat file databases, 484–485
    graphical database clients, 506
    MySQL, 483
        ACID compliance, 493–494
        command line client, 504–505
        configuring, 494–495
        creating databases, 496–497
        data integrity, 493–494
        data locking, 492
        database access, 501–503
        inserting data into tables, 490
        mysql command, 506
        mysqladmin command, 506
        mysqldump command, 506
        MySQLGUI, 506
        online references, 493, 507
        PostgreSQL versus, 492–494
        procedural languages, 494
        retrieving data from tables, 490–492
        root user passwords, 495
        speed, 492
        SQL subqueries, 494
        stored procedures, 494
        triggers, 494
    NoSQL, 485, 509–510
        advantages of, 510, 511
        BaseX, 517
        Berkeley DB, 512, 579
        BigTable, 517
        Cassandra, 513
        CouchDB, 515–516
        disadvantages of, 510–511
        document stores, 514–515
        etcd, 513
        FlockDB, 519
        graph stores, 518
        HBase, 518
        HyperGraphDB, 519
        key/value stores, 512
        Memcached, 513–514
        MemcacheDB, 514
        MongoDB, 516
        Neo4j, 518
        NewSQL and, 511
        online references, 519–520
        OrientDB, 519
        Redis, 514
        Riak, 514
        Scylla, 514
        UnQL, 511
        wide column stores, 517
    OR operators, 491–492
    PostgreSQL
        ACID compliance, 493–494
        command line client, 505–506
        configuring, 498
        creating database users, 499–500
        creating databases, 499

data integrity, 493–494
data locking, 492
database access, 501–503
deleting database users, 500
granting/revoking privileges, 500–501
initializing data directories, 498
MySQL versus, 492–494
online references, 493, 507
pgAdmin, 506
procedural languages, 494
retrieving data from tables, 491
speed, 492
SQL subqueries, 494
stored procedures, 494
triggers, 494
relational databases, 485. *See also* MySQL; PostgreSQL
creating tables, 488–489
DBA, 483–484
inserting data into tables, 489–490
operation of, 486–487
SQL basics, 487–492
Datadog, 298
date/time settings
date command, 19
Ubuntu, 19–20
DaVinci Resolve, 77
DBA, 483–484, 511
Debian Linux, 24
debugging C/C++ programming, 602
checking source code, 602–603
gdb command, 603
gprof command, 603
splint command, 602–603
symbolic debugging, 603
tracking function time, 603
Déjà Dup, 313–314
deleting
directories, 129–131, 161–162
PostgreSQL database users, 500
user accounts, 137
deluser command, 137, 209
dependency checking, 599–600
depmod command, 407
desktop environments, 41–42
Budgie, 49
changing, 39
GNOME, 41
Dash, 43, 44
Mutter, 42
Power icon, 43, 45
Shell, 43–45
Show Applications icon, 43, 44
GNOME 2, 48
KDE, 45
Kubuntu, 45–46
Kylin, 50
Lubuntu, 47–48
LXDE, 47–48
MATE, 48–49
online references, 50
Ubuntu desktops, sharing, 424
Ubuntu MATE, 48–49
widgets, 42
X server
benefits of, 34
components of, 33–34
display manager, 39
display managers, 33
online references, 40
starting, 39
terminal clients, 33–34
window manager, 33–34
xorg.conf files, 33–39

Xfce, 46
Xubuntu, 46–47
development (Ubuntu)
Bazaar, 639
fixing bugs, 640–643
GPG keys, 637–639
introduction to, 636–637
Launchpad, 614–615
creating accounts, 638
work environments, 638–640
local bash shells, configuring, 640
online references, 643
package installation
code packaging, 640–643
packages, 637–638, 640–643
pbuilder, 638
SSH keys, 638, 639
Device section (xorg.conf files), 37–38
devices
all-in-one (Print/Fax/Scan) devices, 435
devices.txt files, 403
i2c-dev devices, 407
security, 385
DevOps
Chaos Engineering, 614
CI/CD tools, 614
Sysadmin versus, 535–536
df command, 286–287
dhclient command, 369
DHCP, 338, 361
activating, 357
dhcpclient, 358
Dynamic DNS, 356–357
networks
configuring, 355–361
host configurations, 359–361
operation of, 356
servers, 358–359
software, installing/configuring, 358–359
dial-up Internet access, configuring, 367–368
diff command, 170
differences in files, finding, 170
digiKam, 70
digital cameras
Ubuntu support, 70
USB connections, 70
directories, 112–114
/bin directory, 114
changing, 120, 145–147
children directories, 118
creating, 129, 160
data directory initialization, PostgreSQL, 498
deleting, 129–131, 161–162
DIT, 580
/etc directory, 114
files
copying, 131, 147
creating, 128–129
displaying contents of, 132
finding, 147
listing in directories, 156–158
moving, 131, 161
printing last lines of files, 163–164
renaming, 131, 161
sorting file contents, 162–163
finding current directory, 120
home directories, 208, 427–428
/home directory, 115
kernel directories, 403–405
LDAP directories, populating, 582–583
listing contents of, 118–119
parent directories, 118
permissions, 122–127
printing directory sizes, 148

/proc directory, 115–117
regular expressions, 133
/sbin directory, 114
scripts, confining to directories, 198–199
/tmp directories, 117
user directories, 115
/usr directory, 117
shared data, 117
X server, 33
/var directory, 117
wildcard characters, 133
DirectoryIndex directive (Apache web server), 449
disaster recovery, planning, 390–391
disk quotas, 287
configuring, 223
implementing, 222–223
managing, 222
display managers, X server, 33, 39
displaying
available memory space, 286
contents of files, 132
free space (hard disks), 286–287
distributed processing, X server, 32–33
distributions
Linux distributions
included software, 24–25
versions of, 24
Ubuntu distributions, versions of, 27
DIT, 580
dmesg command, 21, 200
DNS, 448–449
Dynamic DNS, 356–357
online references, 595
records, 589–590
A records, 590
AAAA records, 590
CNAME records, 590–591
MX records, 591
NS records, 591
SOA records, 592–593
TXT records, 593
requests, 589–590
search orders, 355
servers, 588, 589–590, 593–595
DocBook, Publican, 59–60
Docker, 533
document stores, 514–515
documentation
apropos command, 112
games, 89
Linux, 28–30
manual pages, 111–112
mind maps, 61
reading, 111–112
TeX documents
Kile, 61
LaTeX, 60
LyX, 60
Texmaker, 60
Ubuntu, 28, 30
Wayland, 32
whereis command, 112
WYGIWYW, 60
DocumentRoot directive (Apache web server), 449
domains
DNS
online references, 595
records, 590–593
requests, 589–590
servers, 588, 589–590, 593–595
hostnames, 589
names, 588–589
root zones, 589
TLD, 588–589

dotdee, 102–103, 322
Dotfiles, 322
downloading
  files/folders, 164
  Ubuntu updates during installation, 8
drivers, video drivers (proprietary), installing, 80
dropdb command, 506
dropuser command, 506
DSL Internet access, configuring, 365
du command, 148
dual-boot systems, Ubuntu installations, 6
dumb gateways, 343
dummy interfaces, 327
DVD installations, Ubuntu, 2, 6–11
dvd+rw-tools, 74
DVD/CD burning applications
  Brasero, 71
  Linux command line
    CD creation, 72–73
    DVD creation, 73–75
  preformatted DVD, 74
  Ubuntu support, 71
Dynamic DNS, 356–357
Dynamo, 514

E

e2fsck command, 397
echo command, 148–149
Eclipse, 606
editing
  /etc/modprobe.conf files, 344
  sound, 76
    Ardour, 76
    Audacity, 76
    Cecilia, 76
    LMMS, 76
    Mixxx, 76
    Rosegarden, 76
  text, 185, 186
    awk, 189–191
    emacs, 185, 188–189
    gedit, 186
    kate, 186
    kedit, 186
    nano, 185, 186–187
    sed, 189–191
    vi, 185, 187–188
    vim, 185
  video, 76, 77
    Avidemux, 77
    Blender, 77
    Cinelerra, 77
    DaVinci Resolve, 77
    Kdenlive, 77
    Lightworks, 77
    OpenShot Video Editor, 77
    PiTiVi, 77
    Shotcut, 77
Elixir, 625
Elm, 625
emacs text editor, 185, 188–189
email, 547
  Autoresponders, 562
  virus scanners, 562
embedded spaces, resolving strings with, 248–249
emulators, 79
encrypted files, 318
endless loops, 261–262
enterprise servers, monitoring, 298–299
environment variables, 182–185, 242
erasing, hard disks, Ubuntu installations, 9
Erlang, 625

error messages
- redirecting input/output of commands, 169–170
- sudo command, 134–136

/etc directory, 114
etcd, 513
/etc/host.conf files, 355
/etc/hosts files, 350–351
/etc/modprobe.conf files, 344, 407
/etc/modprobed files, 407
/etc/netplan/*.yaml files, 353–355
/etc/nsswitch.conf files, 351–352, 355
/etc/resolv.conf files, 352–353
/etc/samba/smb.conf files, 426–429
etckeeper, 321–322
ethereal command, 369
Ethernet
- 10G Ethernet networking, 340
- 50G Ethernet networking, 340
- Gigabit Ethernet. *See* 1000BASE-T networking
- PPPoE, 364

evaluating backup strategies, 304–306
event handlers, 468–469
executing
- jobs in parallel, 181–182
- shell scripts, 237–239

exit statements, 269
expr statements, 266
expressions, 235
- comparisons of expressions, 250–251
  - file operators, 253–254, 257–258
  - logical operators, 255, 259
  - numeric comparisons, 252–253, 256–257
  - strings, 251–252, 255–256
- directories, 133

external attacks, 379–380
external hard drives, backups, 308

F

faillog command, 289
fax machines, all-in-one (Print/Fax/Scan) devices, 435
FDDI networks, 340
Fetchmail, 557–560
fg command, 174–175
fiber-optic networks, 340, 342
File Roller, 312
file systems
- authentication, Apache web server, 452
- directories, 112–114
  - /bin directory, 114
  - changing, 120, 145–147
  - children directories, 118
  - copying files, 131, 147
  - creating, 129, 160
  - creating files, 128–129
  - deleting, 129–131, 161–162
  - displaying contents of files, 132
  - /etc directory, 114
  - finding current directory, 120
  - finding files, 147
  - /home directory, 115
  - listing contents of, 118–119
  - listing files in directories, 156–158
  - moving files, 131, 161
  - parent directories, 118
  - printing directory sizes, 148
  - printing last lines of files, 163–164
  - /proc directory, 115–117
  - regular expressions, 133
  - renaming files, 131, 161
  - /sbin directory, 114
  - sorting file contents, 162–163
  - /tmp directories, 117
  - /usr directory, 117
  - /var directory, 117

wildcard characters, 133
hierarchy of, 112–114
permissions, 120–121
altering, 123–124
assigning, 121–122
changing file groups, 125
changing file permissions, 125, 147
setting with ACL, 127–128
sgid permissions, 125–127
sticky bit permissions, 126–127
suid permissions, 125–127
viewing/modifying default permissions, 124–125
Files section (xorg.conf files), 35–36
files/folders
apache2.conf files, 447–449
comparisons, 170
compressed files, 191–192, 318
configuration files, version control, 320–322
copying. *See also* mirroring, 131–132, 147, 316–317
between machines (remote access), 372–373
cp command, 318
rsync command, 319–320
tar command, 317–318
creating in directories, 128–129
disabling access time, 397
displaying contents of, 132
Dotfiles, 322
downloading, 164
encrypted files, 318
/etc/host.conf files, 355
/etc/hosts files, 350–351
/etc/modprobe.conf files, 407
/etc/modprobed files, 407
/etc/netplan/*.yaml files, 353–355
/etc/nsswitch.conf files, 351–352, 355
/etc/resolv.conf files, 352–353
file operator comparisons, 253–254, 257–258
File Roller, 312
finding, 149–151
differences in files, 170
from indexes, 156
similarities in files, 170–171
in Ubuntu, 12
inodes, 155
LDIF files, 583
links, creating, 154–156
listing in directories, 156–158
log files
checking, 287–289
rotating, 289–291
makefiles, 599–600
mirroring. *See also* copying, 164
moving, 131, 161
network configuration files, 350–355
new user home directories, 208
NFS, 422
clients, 423–424
installing, 422
servers, 422–423
starting/stopping, 422
password files, 212–214
permissions, 204
altering, 123–124
changing file groups, 125
changing permissions, 125, 147
viewing/modifying default permissions, 124–125
printing
contents of, 144–145
file sizes, 148
last lines of files, 163–164
renaming, 131, 161
restoring files from archives, 311–312

Samba, 424
configuring, 425, 427–429
/etc/samba/smb.conf files, 426–429
installing, 425–426
saving files from nonbooting hard disks, 324
sharing
NFS, 422–424
online references, 435
Samba, 424–431
sorting contents of, 162–163
temporary file storage, /tmp directories, 117
time warps, 212
variable data files, /var directory, 117
filesystems
performance tuning, 394–395
synchronizing, 396
find command, 126
finding
directories, 120
files/folders, 149–151
differences in files, 170
from indexes, 156
similarities in files, 170–171
in Ubuntu, 12
groups, 206
software, with APT, 98–99
finger information fields, 213
Firefox, 52–53
firewalls
configuring, 386–388
iptables, 388
nftables, 388
UFW, 386–387
.flac files, 65
flat file databases, 484–485
flexbackup, 316
FlightGear, 87
FlockDB, 519
.flv files, 75
folders. *See* files/folders
for statements, 259–261
foreground jobs, 174–175
formatting, DVD, 74
Forth, 626
Fortran, 626
FQDN, 448–449, 580
free command, 286
free/open-source software, Office compatibility, 55
free space, displaying, 286–287
hard disks, 286–287
memory space, 286
Frets on Fire, 86
Frozen Bubble, 84
fstab command, 114
ftp command, 372
full backups
with incremental backups, 307
on a periodic basis, 306–307
tar command, 310–311
functions, shell scripts, 269–270

G

Game Jolt, 82
games, 79, 80
Battle for Wesnoth, 85–86
commercial games, 88
documentation, 89
emulators, 79
FlightGear, 87
Frets on Fire, 86
Frozen Bubble, 84

Game Jolt, 82
GOG.com, 82
Humble, 82
installing
from Ubuntu repositories, 82–87
video drivers (proprietary), 80
itch.io, 82
kid-friendly games, 88
LGDB, 82
online game sources, 81–82
online references, 89
Scorched 3D, 83–84
Speed Dreams, 87
Steam, 81
SuperTux, 84–85
video drivers (proprietary), installing, 80
Warsow, 82
Windows games, 88–89
gateways
dumb gateways, 343
smart gateways, 343
GCC, 597, 603–604
gdb command, 603
gedit, 59
gedit text editor, 186
genprof, 389
GHC, 627
GID, 204, 448
.gif files, 68
Gigabit Ethernet. *See* 1000BASE-T networking
GIMP, 66–67
Git, software project management, 610–611
Glade, 607
GNOME, 41
Dash, 43, 44
File Roller, 312
Glade, 607
Mutter, 42
performance tuning, 3933
Power icon, 43, 45
Shell, 43–45
Show Applications icon, 43, 44
GNOME 2, 48
gnome-nettool, 297
GNU
Coreutils, 199
GCC, 597, 603–604
GPL, development of, 23–24
Go programming language, 626–627
GOG.com, 82
Google Chrome, 53–54
Google Chromium, 53–54
Google Cloud, 539
Google Play, 607
GParted, 9–10
gpasswd command, 207
GPG keys
configuring, 637–638
uploading, 638–639
GPL, development of, 23–24
gprof command, 603
GPT, boot process, 274
granting/revoking privileges, PostgreSQL, 500–501
graph stores, 518
graphical database clients, 506
graphical development tools, 605
Eclipse, 606
Glade, 607
IDE, 605–606
KDevelop Client, 606–607
NetBeans, 606
Oracle JDeveloper, 606
SDK, 606
Visual Studio Code, 606

graphical network configuration tools, 355
graphical process tools, 292
  Conky, 292–297
  gnome-nettool, 297
  vncviewer, 297
  Wireshark, 298
graphics, 67–68
  .bmp files, 68
  converting, 68–69
  .gif files, 68
  image captures, 69
  .jpg files, 68
  .pbm files, 69
  .pcx files, 68
  .pgm files, 69
  .png files, 68
  .pnm files, 69
  .ppm files, 69
  .svg files, 68
  .tif files, 68
graphics manipulation applications, 66, 70
  Blender, 70, 77
  CinePaint, 70
  darktable, 70
  digiKam, 70
  GIMP, 66–67
  Hugin, 70
  Inkscape, 70
  Krita, 70
  nautilus-image converter, 69
  netpbm tools, 69
  Photoshop, 67
  POV-Ray, 70
  Radiance, 70
  Shotwell Photo Manager, 66, 71
  Simple Scan, 67
  Xara Xtreme, 70
grep command, 151–152, 234
Groovy, 627
Group directive (Apache web server), 448
group management, 205
  finding groups, 206
  listing groups, 205–206
  tools, 206–207
groupadd command, 207
groupdel command, 207
groupmod command, 207
groups command, 224
grpck command, 207
GRUB
  recovery mode, 196
  reinstalling, 195–196
  Ubuntu bootups, troubleshooting, 195–196
GRUB2
  boot loaders, restoring, 323–324
  installing, 5
Guacamole, 377
GUI, 31, 41–42
  Budgie, 49
  GNOME, 41
    Dash, 43, 44
    Mutter, 42
    Power icon, 43, 45
    Shell, 43–45
    Show Applications icon, 43, 44
  GNOME 2, 48
  KDE, 45
  Kubuntu, 45–46
  Kylin, 50
  local GUI clients, database access, 503
  Lubuntu, 47–48
  LXDE, 47–48
  MATE, 48–49
  MySQLGUI, 506

pgAdmin, 506
Ubuntu MATE, 48–49
Wayland, 32
widgets, 42
X server, 32, 33
benefits of, 33–34
components of, 33–34
display manager, 39
display managers, 33
distributed processing, 32–33
online references, 40
starting, 39
terminal clients, 33–34
/usr directory, 33
window manager, 33–34
xorg.conf files, 33–39
Xfce, 46
Xubuntu, 46–47
gunzip command, 191
gzip command, 191

H

hackers, 380
hard disks
disk quotas, 287
erasing, Ubuntu installations, 9
free space, displaying, 286–287
nonbooting disks, saving files from, 324
printing, disk usage, 148
Ubuntu installations, erasing hard disks, 9
hard drives (external), backups, 308
hard links, 155–156
hardware
Mac hardware, Ubuntu compatibility, 3
network hardware, initializing, 343–344
/etc/modprobe.conf files, 343–344
kernel modules, 344–345
modprobe command, 344–345
Ubuntu specifications
Mac hardware, 3
researching, 2–16
Haskell, 627
HBase, 518
hdparm command, 395–396
Heimer, 61
help, Ubuntu online references, 22
history of commands, viewing, 197
hitsujiTMO, 272
home applications, Ubuntu for, 28
home directories, 208
/home directory, 115
home users, backup strategies, 304–305
Horowitz, Eliot, 516
hosting (virtual), Apache web server, 461–462
hostnames, 589
hosts (networks)
adding to /etc/hosts files, 350–351
configuring with DHCP, 359–361
Nginx web server, 473–474
HOWTO documents, Linux, 29
HPC, 598
.htaccess configuration files, Apache web server, 450, 469
AllowOverrides directives, 451–452
Options directives, 451
HTTP servers
Caddy, 482
Cherokee, 480–481
Jetty, 481
lighttpd, 479–480
online references, 482
thttpd, 481–482

Tomcat, 482
Wildfly, 482
Yaws, 480
HTTPS
Apache web server, 464–466
Nginx web server, 476–477
hubs, 342
Hugin, 70
Humble, 82
hwclock command, 19–20
hybrid clouds, 539–540
HyperGraphDB, 519

I

i2c-dev devices, 407
IaaS, 537
ID
GID, 204, 448
PID, 164, 175–176, 237, 274, 566
sgid permissions, 125–127
suid permissions, 125–127
UID, 204, 448
IDE, 605–606
ide.txt files, 404
if statements, 265–266
ifconfig command, 346–348, 369, 385
image captures, 69
image file formats, 67–68
.bmp files, 68
.gif files, 68
.jpg files, 68
.pbm files, 69
.pcx files, 68
.pgm files, 69
.png files, 68
.pnm files, 69
.ppm files, 69
.svg files, 68
.tif files, 68
image scanners, 67
incremental backups
full backups with incremental backups, 307
tar command, 310–311
indexes, finding files from, 156
init systems, 272
initializing
data directories, PostgreSQL, 498
network hardware, 343–344
/etc/modprobe.conf files, 344
kernel modules, 344–345
modprobe command, 344–345
initrd.txt files, 404
Inkscape, 70
inodes, 155
InputDevice section (xorg.conf files), 36–37
input/output of commands
redirecting, 167–170
redirecting with shell command line, 236
standard input/output, 167, 169
input strings, searches, 151–152
inserting data into SQL tables, 489–490
insmod command, 406
installing
Android Studio, 609
Apache web server, 444
DHCP activation at installation, 357
DHCP software, 358–359
games, from Ubuntu repositories, 82–87
GRUB, 195–196
GRUB2, 5
NFS, 422
Nginx web server, 469–470

Samba, 425–426
Squid, 564
Ubuntu, 196
32-bit Ubuntu, 4–5
64-bit Ubuntu, 4–5
backups, 7
boot loaders, 5–6
cloud computing, 536
downloading updates during installation, 8
dual-boot systems, 6
DVD installations, 2, 6–11
erasing hard disks during installation, 9
flavors of Ubuntu, 2–3
language selection, 7–8
Mac hardware compatibility, 3
partition strategies, 5
partitions, 5, 9
password creation, 10
preinstallation process, 1–2
researching hardware specifications, 2–16
storage drives, 9
UEFI, 6
USB thumb drive installations, 3, 6–11
Ubuntu development packages
code packaging, 640–643
packages, 637–638, 640–643
video drivers (proprietary), 80
virt-install, 525–526
instances, 533
internal attacks, 379–380
Internet, 51–52
Chrome, 53–54
Chromium, 53–54
connections, 363–364
checking, 328–330
common configuration information, 364–365
dial-up access, 367–368
DSL access, 365
ISP, 364–365
Linux commands, 369
online references, 369–370
PPP access, 364
PPPoE, 364
troubleshooting, 368–369
Ubuntu commands, 369
dummy interfaces, 327
Firefox, 52–53
introduction to, 51
ISP, 364–365
localhost interfaces, 326
checking availability of, 326
configuring manually, 327
I/O transfer speeds, 396
IP addressing
IPv4 addressing, 331–332
IPv6 addressing, 332, 334–336
Squid, specifying client IP addresses, 569–570
ip command, 348–349
IP masquerading, 332–333
ip route command, 346, 349
iptables, 388
ISP, 364–365
itch.io, 82
iwconfig command, 369

J

Java programming, 604, 627–628
JVM, 604
OpenJDK, 604–605
JavaScript, 628
JDeveloper (Oracle), 606

Jetty, 481
jobs
  background jobs, 174–175
  foreground jobs, 174–175
  listing, 173
  moving, to background/foreground, 174–175
  parallel, executing in, 181–182
  running repeatedly, 229–231
jobs command, 173
.jpg files, 68
JSON, 628
Juju, 540–541, 545
JVM, 604

K

kate text editor, 186
kdat, 312–313
KDE, 45
  kdat, 312–313
  KDE ark archiving tool, 312–313
  monitoring tools, 298
  performance tuning, 3933
Kdenlive, 77
KDevelop Client, 606–607
kdf, 298
kedit text editor, 186
kernel ring buffers, reading contents of, 200
kernels, 401, 402
  Android mobile development, 608
  compiling, 418–419
  configuring, 403, 414–417
  custom kernels, 402
  devices.txt files, 403
  directories, 403–405
  first kernel, 402
  ide.txt files, 404
  initrd.txt files, 404
  kernel oops, 419
  kernel-parameters.txt files, 404
  KVM, 523–526
  loading, 274
  loading process, 344–345
  make utility, 404
  modules, 406–408
  numbering schema, 24
  obtaining sources, 409–410
  online references, 419–420
  patching, 410–411
  performance tuning, 398–399
  /proc directory, 115–117
  RAM disk images, creating, 418
  recompiling, 408–409
  security, 379
  source tree, 403–405
  sysrq.txt files, 404
  troubleshooting, 418–419
  types of, 405–406
  Ubuntu kernels, 401–402
  versions of, 409
key-based logins, 373–375
key/value stores, 512
kid-friendly games, 88
Kile, 61
kill command, 283–284
killall command, 284
Kotlin, 628–629
Krita, 70
ksh, 226
ksysguard, 298
Kubernetes, 534
Kubuntu, 45–46
KVM, 523–526
Kylin, 50

L

LAMP, 437–439
LAN, network printers, 431–432
Landscape, 541, 545
languages
  procedural languages, relational databases, 494
  programming languages, 621–622
    Ada, 622
    Clojure, 622–623
    COBOL, 623–624
    D, 624
    Dart, 624
    Elixir, 625
    Elm, 625
    Erlang, 625
    Forth, 626
    Fortran, 626
    Go, 626–627
    Groovy, 627
    Haskell, 627
    Java, 627–628
    JavaScript, 628
    Kotlin, 628–629
    Lisp, 629
    Lua, 629
    Mono, 629–630
    OCaml, 630
    online references, 633–634
    Perl, 630
    PHP, 631
    Python, 631
    Raku, 631
    Ruby, 631–632
    Rust, 632
    Scala, 632
    Scratch, 632–633
    Vala, 633
  selecting, Ubuntu installations, 7–8
large enterprises, backup strategies, 305
last command, 212
last lines of files, printing, 163–164
lastlog command, 289
later, scheduling tasks for, 227–229
LaTeX, 60
Launchpad, 614–615
  accounts, creating, 638
  GPG keys, 638–639
  SSH keys, 639
  work environments, configuring, 638–640
LDAP, 579–580
  administration, 584–585
  configuring
    clients, 584
    servers, 580
  directories, populating, 582–583
  ldap-utils package, 584–585
  LDIF files, 583
  online references, 585
  schemas, creating, 580–581
  Thunderbird, LDAP client configuration, 584
LDIF files, 583
LEMP, 439
less command, 132, 152–154
LGDB, 82
libraries (software), Android mobile development, 608
LibreOffice
  brief history of, 57–58
  components of, 56–57
  Office compatibility, 55
  VBA, 55
lighttpd, 479–480

Lightworks, 77

links

- files/folders, creating links, 154–156
- hard links, 155–156
- symlinks, 154–156

Linux

- C programming, 598–599
- command line
  - && operators, 180, 198
  - ! ! operator, 196–197
  - | | operator, 198
  - accessing, 107
  - apropos command, 112
  - background jobs, 174–175
  - basic commands, 143–144
  - BIOS, 195
  - boolean operators, 180
  - byobu, 192–193
  - CD creation, 72–73
  - changing directories, 145–147
  - command shortcuts, 198
  - commands list, 143–144
  - commonly used commands/programs, 139
  - compressed files, 191–192
  - confining scripts to directories, 198–199
  - copying files, 147
  - Coreutils, 199
  - creating links, 154–156
  - defined, 106–107
  - directories, 112–117
  - downloading files/folders, 164
  - DVD creation, 73–75
  - environment variables, 182–185
  - executing jobs in parallel, 181–182
  - file comparisons, 170
  - file systems, 112–114
  - finding differences in files, 170
  - finding files, 149–151
  - finding files from indexes, 156
  - finding similarities in files, 170–171
  - foreground jobs, 174–175
  - GRUB, 195–196
  - input string searches, 151–152
  - kernel ring buffers, 200
  - listing files in directories, 156–158
  - listing jobs, 173
  - listing processes, 171–173
  - listing system information, 158–159
  - logging out of, 108
  - manual pages, 111–112, 159–160
  - multiple terminal operation, 192–193
  - online references, 139, 165, 200
  - piping commands, 178–180
  - polite system resets, 194–195
  - printing directory sizes, 148
  - printing disk usage, 148
  - printing file contents, 144–145
  - printing resource usage, 175–177
  - prioritizing processes, 177–178
  - reading documentation, 111–112
  - reasons for using, 142–143
  - rebooting systems, 138–139
  - redirecting input/output of commands, 167–170
  - REISUB, 194–195
  - remote computer logins, 108–109
  - repeating text, 148–149
  - rerunning previous commands, 196–197
  - root accounts, 110–111, 133–134
  - running background processes, 173–174
  - running commands in sequence, 180–181
  - shutting down systems, 137–138

standard input/output of commands, 169
substituting processes, 181
super user accounts, 109–111, 133–134
text editors, 185–191
text-based logins, 107–108
troubleshooting sudo command, 134–136
troubleshooting Ubuntu bootups, 195
user accounts, 109–111, 136–137
viewing command history, 197
whereis command, 112
daemons, 25
Debian Linux, 24
development of, 23–25
directories, 112–114
/bin directory, 114
changing, 120, 145–147
children directories, 118
copying files, 131, 147
creating, 129, 160
creating files, 128–129
deleting, 129–131, 161–162
displaying contents of files, 132
/etc directory, 114
finding current directory, 120
finding files, 147
/home directory, 115
listing contents of, 118–119
listing files in directories, 156–158
moving files, 131, 161
parent directories, 118
permissions, 122–127
printing directory sizes, 148
printing last lines of files, 163–164
/proc directory, 115–117
regular expressions, 133
renaming files, 131, 161
/sbin directory, 114
sorting file contents, 162–163
/tmp directories, 117
/usr directory, 117
/var directory, 117
wildcard characters, 133
distributions
included software, 24–25
versions of, 24
documentation, 28–30
file systems
directories, 112–120
hierarchy of, 112–114
permissions, 120–128
filesystems
performance tuning, 394–395
synchronizing, 396
GNU GPL, development of, 23–24
HOWTO documents, 29
kernels, 401, 402
Android mobile development, 608
compiling, 418–419
configuring, 403, 414–417
creating RAM disk images, 418
custom kernels, 402
devices.txt files, 403
directories, 403–405
first kernel, 402
ide.txt files, 404
initrd.txt files, 404
kernel oops, 419
kernel-parameters.txt files, 404
loading, 274
make utility, 404
modules, 406–408
obtaining sources, 409–410
online references, 419–420
patching, 410–411
performance tuning, 398–399

        recompiling, 408–409
        security, 379
        source tree, 403–405
        sysrq.txt files, 404
        troubleshooting, 418–419
        types of, 405–406
        versions of, 409
    location of commands, printing, 164
    manual pages, 29–30
    network connectivity commands, 369
    numbering schema, 24
    online references, 23
    paging through output, 152–154
    reasons for using, 25–26
    rebooting systems, 138–139
    security, 385
    SELinux, 388
    shells, 25
    shutting down systems, 137–138
    user accounts
        creating, 136–137
        deleting, 137
    video, viewing, 76
    viruses, 385
    web resources, 23
    X server, 32, 33
        benefits of, 33–34
        components of, 33–34
        display manager, 39
        display managers, 33
        distributed processing, 32–33
        online references, 40
        starting, 39
        terminal clients, 33–34
        /usr directory, 33
        window manager, 33–34
        xorg.conf files, 33–39
Lisp, 629
Listen directive (Apache web server), 447
listing
    contents of directories, 118–119
    files/folders in directories, 156–158
    groups, 205–206
    jobs, 173
    processes, 171–173
    system information, 158–159
    units (services), 276
LMMS, 76
ln command, 154–156
loading process
    kernels, 274, 344–345
    modprobe command, 344–345
    modules, 408
local bash shells, configuring, 640
local GUI clients, database access, 503
localhost interfaces, 326
    checking availability of, 326
    configuring manually, 327
locate command, 156
location of commands, printing, 164
locking
    data, relational databases, 492
    users out of accounts, 209
log files
    checking, 287–289
    rotating, 289–291
logging, Apache web server, 463–464
logging out of command line, 108
logical operators, comparisons, 255, 259
logins
    command line
        from remote computers, 108–109
        text-based logins, 107–108
    key-based logins, 373–375
logname command, 224

loopback interfaces. *See* localhost interfaces
loops (endless), 261–262
loss of data, 302
lp command, 435
lpc command, 435
lpq command, 435
lprm command, 435
lpstat command, 435
ls command, 118–119, 156–158
lsblk command, 158–159
lshw command, 158–159
lsmod command, 158–159, 406–408
lspci command, 158–159
Lua, 629
Lubuntu, 47–48
lusers, 205
LXC, 532
LXD, 533
LXDE, 47–48
LyX, 60

M

MaaS, 537
Mac hardware, Ubuntu installations, 3
MAC systems, 388
macros, 600–601
Mail Delivery Agent (MDA), 560-562
Mail Transport Agent (MTA), 548
Maildir Versus Mbox, 550
make utility, 404
makefiles, 599–600
managing
  Apache web server
    .htaccess configuration files, 450–452, 469
    access control, 452, 455
    configuring, 446–449, 450–452
    development of, 443–444
    file system authentication, 45–455
    HTTPS, 464–466
    installing, 444
    logging, 463–464
    modules, 455–461
    MPM, 449–450
    online references, 466
    overview of, 444
    runtime servers, 446–449
    security, 444, 452–455
    starting/stopping, 444–446
    threads, 468
    virtual hosting, 461–462
  configurations, 102
    dotdee, 102–103
    Snappy package manager, 103–104
    Ubuntu Core, 103
  disk quotas, 222
  groups, 205
    finding groups, 206
    listing groups, 205–206
    tools, 206–207
  modules, 406–408
  Nginx web server, 467–469
    building sources, 469–470
    configuring, 470–473
    HTTPS, 476–477
    installing, 469–470
    modules, 475
    online references, 477
    PHP, 474–475
    uninstalling, 470
    virtual hosting, 473–474
  passwords, 212, 216
    changing in batches, 216–217
    password files, 212–214
    shadow passwords, 214–215
    system password policies, 212

sets of servers, 543
  Ansible, 544
  CFEngine, 545
  Chef, 544
  Juju, 545
  Landscape, 545
  online references, 545
  Puppet, 543–544
  SaltStack, 544–545
software
  APT, 95–100
  apt-get, 99–100
  compiling software, 100–102
  configuration management, 102–104
  online references, 104
  Software Updater, 94–95
  Synaptic, 92–94
  Ubuntu Software, 91
software project management
  Bazaar, 611–612
  Git, 610–611
  Subversion, 612–613
system management tools, 292
  Conky, 292–297
  gnome-nettool, 297
  System Monitor, 292
  vncviewer, 297
  Wireshark, 298
system resource management
  APT, 95–100
  apt-get, 99–100
  compiling software, 100–102
  configuration management, 102–104
  online references, 104
  Software Updater, 94–95
  Synaptic, 92–94
  Ubuntu Software, 91
user accounts, 201–202, 204, 207
  adding accounts, 209–211
  administration privileges, 217
  BOFH, 205
  changing identities, 217–219
  disk quotas, 222–223
  file permissions, 204
  GID, 204
  group management, 205–207
  locking users out of accounts, 209
  lusers, 205
  monitoring user activity, 211–212
  online references, 224
  passwords, 202, 212–217
  root privileges, 219–222
  root user accounts, 202–203
  stereotypes, 205
  super user accounts, 202, 203
  system user accounts, 203–204
  tools, 208–209
  Ubuntu commands, 223–224
  UID, 204
  usernames, 211
manual pages, 111–112
  Linux, 29–30
  reading, 159–160
MATE, 48–49
MBR, boot process, 274
MEAN, 440–441
Memcached, 513–514
MemcacheDB, 514
memory, displaying available space, 286
Merriman, Dwight, 509
message block printing, 432–433
microservice architectures, 531
Microsoft Azure, 539
mind maps, 61

mirroring data, 307
mirroring. *See also* copying, 164
Mixxx, 76
mkdir command, 129, 160
mkisofs command, 72
"moat mentality", 333
mobile development, Android, 607
  Android Runtime, 608
  Android Studio, 609
  Application Framework, 608–609
  applications, 609
  ARM processors, 608
  Google Play, 607
  Linux kernels, 608
  RISC processors, 608
modifying default file permissions, 124–125
modinfo command, 407
modprobe command, 344–345, 406–407
modprobe.d/ command, 114
Module section (xorg.conf files), 36
modules, 406–408
  Apache web server, 455–461
  MPM, Apache web server, 449–450
  Nginx web server, 475
Mojo, 541
MongoDB, 516
Monitor section (xorg.conf files), 37
monitoring
  console-based monitoring, 281–283
  enterprise servers, 298–299
  KDE monitoring tools, 298
  systems
    checking log files, 287–289
    Conky, 292–297
    console-based monitoring, 281–283
    disk quotas, 287
    displaying free hard disk space, 286
    displaying used memory, 286
    enterprise servers, 298–299
    graphical process tools, 292–298
    KDE monitoring tools, 298
    killing processes, 283–284
    online references, 299
    priority scheduling, 285
    rotating log files, 289–291
    system management tools, 292–298
    System Monitor, 292
  user activity, 211–212
Mono, 629–630
motd files, 138
mounting, Samba shares, 430–431
.mov files, 75
moving
  files/folders, 131, 161
  jobs
    background jobs, 174–175
    to background/foreground, 174–175
    foreground jobs, 174–175
.mp3 files, 65
.mp4 files, 65
.mpeg files, 75
MPM, Apache web server, 449–450
MS Office
  CrossOver Office, 61
  free/open-source software compatibility, 55
  LibreOffice compatibility, 55
  VBA, 55
MS-DOS, 107
mtr command, Internet connections, checking, 329–330
multicasting, 338
multimedia applications
  ALSA, 64
  CD/DVD burning applications

Brasero, 71
Linux command line, CD creation, 72–73
Linux command line, DVD creation, 73–75
Ubuntu support, 71
graphics manipulation applications, 66, 70
Blender, 70, 77
CinePaint, 70
darktable, 70
digiKam, 70
GIMP, 66–67
Hugin, 70
Inkscape, 70
Krita, 70
nautilus-image converter, 69
netpbm tools, 69
Photoshop, 67
POV-Ray, 70
Radiance, 70
Shotwell Photo Manager, 66, 71
Simple Scan, 67
Xara Xtreme, 70
music applications, 65
Banshee, 66
Rhythmbox, 66
Sound Juicer, 66
online references, 77
OSS, 64
PulseAudio, 64
recording/editing sound, 76
sound cards, 63–64
video editing applications, 77
Avidemux, 77
Blender, 77
Cinelerra, 77
DaVinci Resolve, 77
Kdenlive, 77
Lightworks, 77
OpenShot Video Editor, 77
PiTiVi, 77
Shotcut, 77
multiple jobs, executing in parallel, 181–182
multiple terminal operation, byobu, 192–193
music applications, 65
Banshee, 66
Rhythmbox, 66
Sound Juicer, 66
Mutter, 42
mv command, 131, 161
MX records, 591
MySQL, 483
ACID compliance, 493–494
command line client, 504–505
configuring, 494–495
creating databases, 496–497
data integrity, 493–494
data locking, 492–493
database access, 501
local GUI clients, 503
SSH, 501–502
web browsers, 503
inserting data into tables, 490
mysql command, 506
mysqladmin command, 506
mysqldump command, 506
MySQLGUI, 506
online references, 493, 507
PostgreSQL versus, 492–494
procedural languages, 494
retrieving data from tables, 490–492
root user passwords, 495
speed, 492
SQL subqueries, 494
stored procedures, 494
triggers, 494

mysql command, 506
mysqladmin command, 506
mysqldump command, 506
MySQLGUI, 506

N

Nagios, 298
name servers
  DNS
    online references, 595
    records, 590–593
    requests, 589–590
    servers, 588, 589–590, 593–595
  setting, 352–355
naming services, 351–352
nano text editor, 185, 186–187
NAS, 308
NAT, 332
nautilus-image converter, 69
needs assessments, backup strategies, 303–304
Neo4j, 518
neofetch command, 158–159
NetBeans, 606
NetBoot, boot process, 274
NetCat, 330
netmasks. *See* subnet masks
netpbm tools, 68–69
netstat command, 346, 350
networks, 337
  10BASE-T networking, 339
  10G Ethernet networking, 340
  50G Ethernet networking, 340
  100BASE-T networking, 339
  1000BASE-T networking, 339
  accessing, chains, 387
  bridges, 343, 524–525
  broadcasting, 338
  cabling, 340
    fiber-optic cabling, 342
    UTP cabling, 341
  chains, 387
  command line network interface configuration
    ifconfig command, 346–348
    ip command, 348–349
    ip route command, 346, 349
    netstat command, 346, 350
  configuring
    command line network interfaces, 345–350
    configuration files, 350–355
    DHCP, 355–361
    /etc/host.conf files, 355
    /etc/hosts files, 350–351
    /etc/netplan/*.yaml files, 353–355
    /etc/nsswitch.conf files, 351–352, 355
    /etc/resolv.conf files, 352–353
    graphical configuration tools, 355
    nm-connection-editor, 345–346
  connections, troubleshooting, 342
  DHCP, 361
    activating, 357
    configuring hosts, 359–361
    installing/configuring software, 358–359
  DNS search orders, 355
  dummy interfaces, 327
  FDDI networks, 340
  fiber-optic networks, 340, 342
  gateways
    dumb gateways, 343
    smart gateways, 343
  hosts
    adding to /etc/hosts files, 350–351
    configuring with DHCP, 359–361

hubs, 342
initializing hardware, 343–344
  /etc/modprobe.conf files, 344
  kernel modules, 344–345
  modprobe command, 344–345
Internet connections, 363–364
  checking, 328–330
  common configuration information, 364–365
  dial-up access, 367–368
  DSL access, 365
  ISP, 364–365
  Linux commands, 369
  PPP access, 364
  PPPoE, 364
  troubleshooting, 342
  Ubuntu commands, 369
IP masquerading, 332–333
IPv4 addressing, 331–332
IPv6 addressing, 332, 334–336
LAN, network printers, 431–432
localhost interfaces, 326
  checking availability of, 326
  configuring manually, 327
"moat mentality", 333
multicasting, 338
name servers, 352–355
naming services, 351–352
NAT, 332
NIC, 338–340
online references, 369–370
printers, 431
  CUPS GUI, 431–434
  LAN, 431–432
  online references, 435
  server message block printing, 432–433
  troubleshooting, 434–435
routers, 343
storage, backups, 308
subnet masks, 337–338
subnetting, 337–338
switches, 342
TCP/IP, 330
  addressing, 331–332
  IP masquerading, 332–333
  ports, 333–334
Token Ring networking, 339
troubleshooting, connections, 342
Ubuntu, 330
unicast addressing, 338
VPN, 563, 572–573
  configuring clients, 573–575
  configuring servers, 575–577
  online references, 577
wireless networks, 361, 382–383
  configuring, 20
  interfaces, 340
  selecting protocols, 363
  Ubuntu support, 361–363
New Relic, 299
new shell programs, running, 239–240
NewSQL, 511
newusers command, 224
NFS, 422
  configuring
    clients, 423–424
    servers, 422–423
  installing, 422
  starting/stopping, 422
nftables, 388
Nginx web server, 467–469
  building sources, 469–470
  configuring, 470–473
  HTTPS, 476–477

installing, 469–470
modules, 475
online references, 477
PHP, 474–475
uninstalling, 470
virtual hosting, 473–474
NIC, 338–339
10BASE-T networking, 339
10G Ethernet networking, 340
50G Ethernet networking, 340
100BASE-T networking, 339
1000BASE-T networking, 339
fiber-optic networks, 340
promiscuous mode, 385
security, 385
Token Ring networking, 339
wireless network interfaces, 340
nice command, 177–178
Nmap, 330, 382
nm-connection-editor, 345–346, 369
nonbooting hard disks, saving files from, 324
NoSQL databases, 485, 509–510
advantages of, 510, 511
BaseX, 517
Berkeley DB, 512, 579
BigTable, 517
Cassandra, 513
CouchDB, 515–516
disadvantages of, 510–511
document stores, 514–515
etcd, 513
FlockDB, 519
graph stores, 518
HBase, 518
HyperGraphDB, 519
key/value stores, 512
Memcached, 513–514
MemcacheDB, 514
MongoDB, 516
Neo4j, 518
NewSQL and, 511
online references, 519–520
OrientDB, 519
Redis, 514
Riak, 514
Scylla, 514
UnQL, 511
wide column stores, 517
NS records, 591
numbering schema, Linux kernels, 24
numeric comparisons, 252–253, 256–257

O

OCaml, 630
Office
CrossOver Office, 61
free/open-source software compatibility, 55
LibreOffice compatibility, 55
VBA, 55
.ogg files, 65, 66
.ogv/.ogg files, 75
online references
Apache web server, 466
backups, 324
BIND, 595
boot process, 278–279
cloud computing, 541
command line, 139, 165, 200
containers, 534
desktop environments, 50
DNS, 595
file sharing, 435
games, 89
HTTP servers, 482

Internet connections, 369–370
kernels, 419–420
LDAP, 585
Linux, 23, 28
multimedia applications, 77
MySQL, 493, 507
network printers, 435
networks, 369–370
Nginx web server, 477
NoSQL databases, 519–520
performance tuning, 396–400
PostgreSQL, 493, 507
programming, 618–619
programming languages, 633–634
QA, 647
remote access, 377–378
security, 391–392
sets of servers, 545
shell scripts, 270
software management, 104
sound, 65
Squid, 577
system monitoring, 299
system resource management, 104
testing, 647
Ubuntu, 22, 28, 30
Ubuntu development, 643
user accounts, 224
virtualization, 529
VPN, 577
Wayland, 32
web browsers, 54
web server stacks, 441
X server, 40
OpenJDK, 604–605
OpenLDAP. *See* LDAP
OpenShot Video Editor, 77
open-source/free software, Office compatibility, 55
OpenSSH. *See* SSH
OpenStack, 538–539
OR operators, 491–492
optimizing performance, 393–394
badblocks command, 397
e2fsck command, 397
file access time, disabling, 397
filesystems, 394–395
GNOME, 3933
hdparm command, 395–396
I/O transfer speeds, 396
KDE, 3933
kernels, 398–399
online references, 396–400
storage disks, 394
tune2fs command, 396–397
Tuned Project, 396–400
Options directives (Apache web server), 451
Oracle JDeveloper, 606
OrientDB, 519
OSS, 64
output
commands
redirecting, 167–170
redirecting with shell command line, 236
standard input/output, 167, 169
paging through, 152–154

P

PaaS, 537
packages
PPA, 615
snap packages, 615–616

Ubuntu development
code packaging, 640–643
configuring, 637–638
installing, 637–638
packet writing, Linux command line, DVD creation, 74–75
paging through output, 152–154
PAM, 215–216
parallel, executing jobs in, 181–182
parameters (positional), 243
accessing/retrieving variables, 244
example of, 243–244
parent directories, 118
partitions
planning, Ubuntu installations, 5
Ubuntu installations, 5
passwd command, 114, 209, 224
passwords, 212, 216, 383–384
brute-forcing, 373–374
changing in batches, 216–217
MySQL root users, 495
password files, 212–214
shadow passwords, 214–215
system password policies, 212
Ubuntu, 10
user accounts, 202
patching kernels, 410–411
pattern-matching, 235
.pbm files, 69
pbuilder, 638
.pcx files, 68
pdcsh, 226
PDF, 58
pdfedit, 58
pdksh, comparisons of expressions, 250–251
file operators, 253–254
logical operators, 255
numeric comparisons, 252–253
strings, 251–252
performance tuning, 393–394
badblocks command, 397
e2fsck command, 397
file access time, disabling, 397
filesystems, 394–395
GNOME, 3933
hdparm command, 395–396
HPC, 598
I/O transfer speeds, 396
KDE, 3933
kernels, 398–399
online references, 396–400
storage disks, 394
tune2fs command, 396–397
Tuned Project, 396–400
periodic basis, full backups on, 306–307
peripherals
digital cameras, 70
scanners, 67
Perl, 630
permissions
directories, 122–127
file systems, 120–128
altering permissions, 123–124
assigning permissions, 121–122
changing file groups, 125
changing file permissions, 125, 147
viewing/modifying default permissions, 124–125
files/folders, 204
altering permissions, 123–124
changing file groups, 125
changing permissions, 125, 147
viewing/modifying default permissions, 124–125
setting with ACL, 127–128
sgid permissions, 125–127
sticky bit permissions, 126–127
suid permissions, 125–127

pgAdmin, 506
.pgm files, 69
Photoshop, 67
PHP, 474–475, 631
physical security, 383–384
PID, 164, 175–176, 237, 274, 566
ping command, checking Internet connections, 328–329
ping6 command, checking Internet connections, 328–329
pipelines (CI/CD), 613
piping commands, 142–143, 178–180, 237
PiTiVi, 77
planning
  backup strategies
    assessing needs/resources, 303–304
    choosing a strategy, 308
    choosing hardware/media, 308–309
    evaluating, 304–306
    full backups on a periodic basis, 306–307
    full backups with incremental backups, 307
    home users, 304–305
    incremental backups, 307
    inheriting, 306
    large enterprises, 305
    mirroring data, 307
    RAID arrays, 307
    simple backup strategies, 306
    small enterprises, 305
    small offices, 305
  disaster recovery plans, 390–391
  partition strategies, Ubuntu installations, 5
  Ubuntu installations, 1–2
    flavors of Ubuntu, 2–3
    partition strategies, 5
    researching hardware specifications, 2–16
.png files, 68
.pnm files, 69
polite system resets, 194–195
ports
  TCP/IP networking, 333–334
  Ubuntu network ports, 421
  uplink ports, 342
positional arguments, 242
positional parameters, 243
  accessing/retrieving variables, 244
  example of, 243–244
Postfix, 552–556
PostgreSQL
  ACID compliance, 493–494
  command line client, 505–506
  configuring, 498
  creating
    database users, 499–500
    databases, 499
  data integrity, 493–494
  data locking, 492
  database access, 501–503
    local GUI clients, 503
    SSH, 501–502
    web browsers, 503
  deleting database users, 500
  granting/revoking privileges, 500–501
  initializing data directories, 498
  MySQL versus, 492–494
  online references, 493, 507
  pgAdmin, 506
  procedural languages, 494
  retrieving data from tables, 491
  speed, 492
  SQL subqueries, 494
  stored procedures, 494
  triggers, 494
POV-Ray, 70

Power icon (GNOME), 43, 45
power management, configuring, in Ubuntu, 18–19
PPA, 615
.ppm files, 69
PPP Internet access, configuring, 364
PPPoE, 366–367
preformatted DVD, 74
preinstallation processes, Ubuntu, 1–2
  flavors of Ubuntu, 2–3
  hardware specifications, 2–16
previous commands, rerunning, 196–197
printing
  all-in-one (Print/Fax/Scan) devices, 435
  commands, location of, 164
  directory sizes, 148
  disk usage, 148
  file contents, 144–145
  file sizes, 148
  last lines of files, 163–164
  network printers, 431
    CUPS GUI, 431–434
    LAN, 431–432
    online references, 435
    server message block printing, 432–433
    troubleshooting, 434–435
  resource usage, 175–177
  server message block printing, 432–433
  sharing printers, Samba, 428–429
    Ubuntu printer configurations, 18
prioritizing
  processes, 177–178
  scheduling priorities, 285
private clouds, 540
privileges
  administration privileges, 217
  granting/revoking with PostgreSQL, 500–501
  root privileges, 219–222
  user accounts, 109–111
/proc directory, 115–117
procedural languages, relational databases, 494
processes
  killing, 283–284
  listing, 171–173
  prioritizing, 177–178
  substituting, 181
processors, Android mobile development, 608
productivity applications
  Celtx, 59
  CrossOver Office, 61
  defined, 56
  gedit, 59
  Heimer, 61
  Kile, 61
  LaTeX, 60
  LibreOffice
    brief history of, 57–58
    components of, 56–57
    Office compatibility, 55
    VBA, 55
  LyX, 60
  online references, 61
  pdfedit, 58
  Publican, 59–60
  Texmaker, 60
  users (typical), 56
  Windows, 61
  Wine, 61
  XML Copy Editor, 60
productivity suites, LibreOffice, Office compatibility, 55
programming
  Android mobile development, 607
    Android Runtime, 608
    Android Studio, 609

Application Framework, 608–609
applications, 609
ARM processors, 608
Google Play, 607
Linux kernels, 608
RISC processors, 608
Bikeshed, 616–618
C, 597–599
autoconf command, 601–602
building programs, 599–601
checking source code, 602–603
configuring code, 601–602
debugging tools, 602–603
dependency checking, 599–600
GCC, 603–604
gdb command, 603
gprof command, 603
macros, 600–601
makefile targets, 600–601
makefiles, 599–600
splint command, 602–603
symbolic debugging, 603
tracking function time, 603
C++597–598, 599
autoconf command, 601–602
building programs, 599–601
checking source code, 602–603
configuring code, 601–602
debugging tools, 602–603
dependency checking, 599–600
GCC, 603–604
gdb command, 603
gprof command, 603
macros, 600–601
makefile targets, 600–601
makefiles, 599–600
splint command, 602–603
symbolic debugging, 603
tracking function time, 603
Chaos Engineering, 614
CI/CD tools, 613–614
GCC, 597
graphical development tools, 605
Eclipse, 606
Glade, 607
IDE, 605–606
KDevelop Client, 606–607
NetBeans, 606
Oracle JDeveloper, 606
SDK, 606
Visual Studio Code, 606
Java programming, 604
JVM, 604
OpenJDK, 604–605
languages, 621–622
Ada, 622
Clojure, 622–623
COBOL, 623–624
D, 624
Dart, 624
Elixir, 625
Elm, 625
Erlang, 625
Forth, 626
Fortran, 626
Go, 626–627
Groovy, 627
Haskell, 627
Java, 627–628
JavaScript, 628
Kotlin, 628–629
Lisp, 629
Lua, 629
Mono, 629–630

OCaml, 630
online references, 633–634
Perl, 630
PHP, 631
Python, 631
Raku, 631
Ruby, 631–632
Rust, 632
Scala, 632
Scratch, 632–633
Vala, 633
online references, 618–619
snap packages, 615–616
software project management
Bazaar, 611–612
Git, 610–611
Subversion, 612–613
Ubuntu Make, 615
version control systems, 609–610
promiscuous mode (NIC), 385
proxy servers
defined, 563–564
Squid, 563–564
ACL, 565–569
configuring, 570–572
configuring clients, 564–565
installing, 564
online references, 577
specifying client IP addresses, 569–570
ps command, 171–173
psql command, 506
public clouds, 538, 539
Publican, 59–60
PulseAudio, 64
Puppet, 543–544
/pwd command, 120
PXE, boot process, 274
Python, 631

Q

QA
Bug Squad, 647
online references, 647
Ubuntu QA Teams, 646–647
Qmail, 549–550
.qt files, 75
quotas (disk), 287
configuring, 223
implementing, 222–223
managing, 222

R

Radiance, 70
RAID arrays, 307
Raku, 631
RAM disk images (kernels), creating, 418
RARP, 338
.raw files, 65
Raymond, Eric, 141–142
rcp command, 372
RDP, VirtualBox, 528
rebooting systems, 138–139
recompiling kernels, 408–409
recording/editing sound, 76
Ardour, 76
Audacity, 76
Cecilia, 76
LMMS, 76
Mixxx, 76
Rosegarden, 76
recovery
data, 323
GRUB2 boot loaders, restoring, 323–324

saving files from nonbooting hard disks, 324
Ubuntu Rescue disc, 323
disaster recovery plans, 390–391
recovery mode (GRUB), 196
reddit.com, 272
redirecting input/output of commands, 167–170, 236
Redis, 514
references (online)
Apache web server, 466
BIND, 595
boot process, 278–279
cloud computing, 541
command line, 139, 165, 200
containers, 534
desktop environments, 50
DNS, 595
file sharing, 435
games, 89
HTTP servers, 482
Internet connections, 369–370
kernels, 419–420
LDAP, 585
Linux, 23, 28
multimedia applications, 77
MySQL, 493, 507
network printers, 435
networks, 369–370
Nginx web server, 477
NoSQL databases, 519–520
online references, 324
performance tuning, 396–400
PostgreSQL, 493, 507
programming, 618–619
programming languages, 633–634
QA, 647
remote access, 377–378
security, 391–392
sets of servers, 545
shell scripts, 270
software management, 104
sound, 65
Squid, 577
system monitoring, 299
system resource management, 104
testing, 647
Ubuntu, 22, 28, 30
Ubuntu development, 643
user accounts, 224
virtualization, 529
VPN, 577
Wayland, 32
web browsers, 54, 61
web server stacks, 441
X server, 40
regular expressions, 235
comparisons of expressions, 250–251
file operators, 253–254, 257–258
logical operators, 255, 259
numeric comparisons, 252–253, 256–257
strings, 251–252, 255–256
directories, 133
reinstalling
GRUB, 195–196
Ubuntu, 196
REISUB, polite system resets, 194–195
relational databases, 485
DBA, 483–484
graphical database clients, 506
MySQL, 483
ACID compliance, 493–494
command line client, 504–505
configuring, 494–495

creating databases, 496–497
data integrity, 493–494
data locking, 492
database access, 501–503
inserting data into tables, 490
mysql command, 506
mysqladmin command, 506
mysqldump command, 506
MySQLGUI, 506
online references, 493, 507
PostgreSQL versus, 492–494
procedural languages, 494
retrieving data from tables, 490–492
root user passwords, 495
speed, 492
SQL subqueries, 494
stored procedures, 494
triggers, 494
operation of, 486–487
PostgreSQL
ACID compliance, 493–494
command line client, 505–506
configuring, 498
creating database users, 499–500
creating databases, 499
data integrity, 493–494
data locking, 492
database access, 501–503
deleting database users, 500
granting/revoking privileges, 500–501
initializing data directories, 498
MySQL versus, 492–494
online references, 493, 507
pgAdmin, 506
procedural languages, 494
retrieving data from tables, 491
speed, 492
SQL subqueries, 494
stored procedures, 494
triggers, 494
SQL, 487–492
creating tables, 488–489
inserting data into tables, 489–490
remote access
copying files between machines, 372–373
Guacamole, 377
key-based logins, 373–375
online references, 377–378
SSH server, 371–375
VNC, 375–377
remote computers, command line, logins, 108–109
rename command, 161
renaming files/folders, 161
repeat statements, 263
repeatedly running jobs, 229–231
repeating text, 148–149
repositories
software repositories, configuring, 15–17
Ubuntu repositories, compiling software, 101–102
Require directive (Apache web server), 452
rerunning previous commands, 196–197
resetting systems, polite system resets, 194–195
resources
assessing, backup strategies, 303–304
usage, printing, 175–177
restoring
files from archives, 311–312
GRUB2 boot loaders, 323–324
retrieving data from SQL tables, 490–492
revoking/granting privileges, PostgreSQL, 500–501

Rhythmbox music application, 66
Riak, 514
ripping audio, 71
RISC processors, 608
rm command, 130–131, 161–162
rmdir command, 129–130
rmmod command, 406
root accounts, 110–111, 133–134
root privileges, 219–222
root prompt, dangers of working in, 15
root user accounts, 202–203
root zones, 589
Rosegarden, 76
rotating log files, 289–291
route command, 369
routers, 343
rsync command, 319–320
rtcwake command, 231–233
Ruby, 631–632
runlevels, 271–272, 275, 278
Runtime (Android), 608
runtime servers, Apache web server
  apache2.conf files, 447–449
  configuring, 446–449
  DirectoryIndex directive, 449
  DocumentRoot directive, 449
  GID, 448
  Group directive, 448
  Listen directive, 447
  ServerAdmin directive, 448
  ServerName directive, 448–449
  ServerRoot directive, 447
  UID, 448
  User directive, 448
  UserDir directive, 449
Rust, 632

S

SaaS, 537
SaltStack, 544–545
Samba, 424
  configuring, 425, 427
  /etc/samba/smb.conf files, 426–429
  installing, 425–426
  mounting, 430–431
  sharing
    home directories, 427–428
    printers, 428–429
  smbclient command, 430
  smbd daemon, starting/stopping, 429–430
  smbstatus command, 430
  testing, 429
saving, files from nonbooting hard disks, 324
/sbin directory, 114
Scala, 632
scanners, 67, 435
scheduling
  backups, full backups on a periodic basis, 306–307
  priorities, 285
  tasks, 226
    automating tasks with scripts, 244–246
    for later, 227–229
    running jobs repeatedly, 229–231
    waking computers from sleep, 231–233
Scorched 3D, 83–84
scp command, 372–373
Scratch, 632–633
screen images, capturing, 69
Screen section (xorg.conf files), 38–39
script kiddies, 380
scripting
  autocracking scripts, 380
  automating tasks, 244–246

break statements, 269
case statements, 267–268
Celtx, 59
comparisons of expressions, 250–251
file operators, 253–254, 257–258
logical operators, 255, 259
numeric comparisons, 252–253, 256–257
strings, 251–252, 255–256
confining to directories, 198–199
endless loops, 261–262
executing scripts, 237–239
exit statements, 269
expr statements, 266
for statements, 259–261, 263
functions, 269–270
if statements, 265–266
interpreting scripts through specific shells, 240–241
new shell programs, 239–240
online references, 270
positional arguments, 242
positional parameters, 243
accessing/retrieving variables, 244
example of, 243–244
repeat statements, 263
select statements, 264
shift statements, 264
special shell characters
` special shell character, 250
", 248–249
' special shell character, 249
backslash, 249–250
list of, 247–248
storing scripts for system-wide access, 240
strings with embedded spaces, resolving, 248–249
variables, 242
accessing values, 243
accessing/retrieving with positional parameters, 244
assigning values, 242
built-in variables, 246–247
unexpanded variables, 249
while statements, 261, 262
writing scripts, 237–239
Scylla, 514
SDK, 606
searches
DNS search orders, 355
input strings, 151–152
security
access control, Apache web server, 452–455
Apache web server, 444, 452
AppArmor, 388–390
applications, 388–390
attacks, 380–381
autocracking scripts, 380
crackers, 380
external attacks, 379–380
hackers, 380
internal attacks, 379–380
script kiddies, 380
viruses, 385
war driving, 383
worms, 380
authentication
Apache web server, 452–455
PAM, 215–216
brute-forcing, 373–374
checklist, 381
configuring, 384–385
devices, 385
disaster recovery plans, 390–391
encrypted files, 318

firewalls, 386–388
iptables, 388
nftables, 388
UFW, 386–387
genprof, 389
kernels, 379
nftables, 388
NIC, 385
online references, 391–392
passwords, 212, 216, 383–384
brute-forcing, 373–374
changing in batches, 216–217
MySQL root users, 495
password files, 212–214
shadow passwords, 214–215
system password policies, 212
Ubuntu installations, 10
user accounts, 202
physical security, 383–384
proxy servers. *See also* Squid, 563–564
spoofing attacks, 593
Tripwire, 384–385
UFW, 386–387
viruses, 385
VPN, 563, 572–573
configuring clients, 573–575
configuring servers, 575–577
online references, 577
vulnerability assessments, 381–382
wireless networks, 382–383
sed text editor, 189–191
select statements, 264
SELinux, 388
semistructured data, 515
sequence, running commands in, 180–181
ServerAdmin directive (Apache web server), 448
ServerLayout section (xorg.conf files), 35
ServerName directive (Apache web server), 448–449
ServerRoot directive (Apache web server), 447
servers
Apache web server
access control, 452, 455
configuring, 446–449, 450–452
development of, 443–444
file system authentication, 452–455
.htaccess configuration files, 450–452, 469
HTTPS, 464–466
installing, 444
logging, 463–464
modules, 455–461
MPM, 449–450
online references, 466
overview of, 444
runtime servers, 446–449
security, 444, 452–455
starting/stopping, 444–446
threads, 468
virtual hosting, 461–462
DHCP servers, 358–359
DNS servers, 588, 589–590, 593–595
enterprise servers, monitoring, 298–299
FQDN, 580
HTTP servers
Caddy, 482
Cherokee, 480–481
Jetty, 481
lighttpd, 479–480
online references, 482
thttpd, 481–482
Tomcat, 482
Wildfly, 482
Yaws, 480

LDAP servers, configuring, 580
message block printing, 432–433
name servers, 352–355
NFS servers, 422–423
Nginx web server, 467–469
building sources, 469–470
configuring, 470–473
HTTPS, 476–477
installing, 469–470
modules, 475
online references, 477
PHP, 474–475
uninstalling, 470
virtual hosting, 473–474
proxy servers, defined. *See also* Squid, 563–564
runtime servers, Apache web server, 446–449
sets of servers, managing, 543
Ansible, 544
CFEngine, 545
Chef, 544
Juju, 545
Landscape, 545
online references, 545
Puppet, 543–544
SaltStack, 544–545
Squid, 563–564
ACL, 565–569
configuring, 570–572
configuring clients, 564–565
installing, 564
online references, 577
specifying client IP addresses, 569–570
SSH server
configuring, 371–372
copying files between machines, 372–373
ftp command, 372
key-based logins, 373–375
scp command, 372–373
sftp command, 373
ssh-keygen command, 373–375
VPN servers, configuring, 575–577
web servers
C10K problem, 467–468
event handlers, 468–469
threads, 468
X server, 32, 33
/usr directory, 33
benefits of, 33–34
components of, 33–34
display manager, 39
display managers, 33
distributed processing, 32–33
online references, 40
starting, 39
terminal clients, 33–34
window manager, 33–34
xorg.conf files, 33–39
server stacks, 437
LAMP, 437–439
LEMP, 439
MEAN, 437
online references, 441
services
boot process
controlling, 278
running services, 271–272
starting/stopping services, 275–278
Startup Applications Preferences, 278
/etc/services files, 351
microservice architectures, 531
naming services, 351–352
units, listing, 276

session writing, DVD creation, 74
set group ID permissions (suid), 125–127
set user ID permissions (suid), 125–127
sets of servers, managing, 543
  Ansible, 544
  CFEngine, 545
  Chef, 544
  Juju, 545
  Landscape, 545
  online references, 545
  Puppet, 543–544
  SaltStack, 544–545
sftp command, 373, 383
sgid permissions, 125–127
shadow passwords, 214–215
shared data, /usr directory, 117
sharing
  files/folders
    NFS, 422–424
    online references, 435
    Samba, 424–431
  home directories with Samba, 427–428
  printers, Samba, 428–429
  Ubuntu desktops, 424
shells. *See also* command line, 233
  ash, 226
  bash, 226, 250–255
  changing, 226
  command line, 233–234
    background processes, 237
    pattern-matching, 235
    piping commands, 237
  dash, 226
  defined, 225–226
  expressions, 235
  GNOME Shell, 43–45
  ksh, 226
  Linux, 25
  local bash shells, configuring, 640
  new shell programs, 239–240
  pdcsh, 226
  pdksh, comparisons of expressions, 250–255
  scripting
    ` special shell character, 250
    ", 248–249
    ' special shell character, 249
    automating tasks, 244–246
    backslash, 249–250
    break statements, 269
    case statements, 267–268
    comparisons of expressions, 250–259
    endless loops, 261–262
    executing scripts, 237–239
    exit statements, 269
    expr statements, 266
    functions, 269–270
    if statements, 265–266
    interpreting through specific shells, 240–241
    new shell programs, 239–240
    online references, 270
    positional arguments, 242
    positional parameters, 243–244
    repeat statements, 263
    resolving strings with embedded spaces, 248–249
    select statements, 264
    shift statements, 264
    special shell characters, 247–248
    for statements, 259–261
    storing scripts for system-wide access, 240
    until statements, 263
    variables, 242–243, 244, 246–247, 249

while statements, 261, 262
writing scripts, 237–239
tcsh, 226, 255–259
types of, 225–226
zsh, 226
shift statements, 264
shortcuts, commands, 198
Shotcut, 77
Shotwell Photo Manager, 66, 71
Show Applications icon (GNOME), 43, 44
shred command, 155
shutdown command, 137–139
shutting down
systems, 137–138
Ubuntu, 11–12
SignalFX, 299
similarities in files, finding, 170–171
simple backup strategies, 306
Simple Mail Transfer Protocol (SMTP), 547
Simple Scan, 67
sleep, waking computers from, 231–233
small enterprises, backup strategies, 305
small offices, backup strategies, 305
smart gateways, 343
smbclient command, 430
smbd daemon, starting/stopping, 429
smbstatus command, 430
snap packages, 615–616
SnapCraft, 616
Snappy package manager, 103–104
Snappy Playpen, 616
snaps, Snappy package manager, 103–104
SOA records, 592–593
soft links. *See* symlinks
software
backup software, 309–316
bundles, Snappy package manager, 103–104
compiling
from source, 100, 101–102
from tarballs, 100–101
from Ubuntu repositories, 101–102
DHCP software, installing/configuring, 358–359
finding, with APT, 98–99
free/open-source software, MS Office compatibility, 55
libraries, Android mobile development, 608
Linux distributions, included software, 24–25
management tools
APT, 95–100
apt-get, 99–100
compiling software, 100–102
configuration management, 102–104
online references, 104
Software Updater, 94–95
Synaptic, 92–94
Ubuntu Software, 91
open-source/free software, MS Office compatibility, 55
project management
Bazaar, 611–612
Git, 610–611
Subversion, 612–613
repositories, configuring, 15–17
Snappy package manager, 103–104
updating, 12–14
Software Updater, 12–14, 94–95
sort command, 162–163
sorting, file contents, 162–163
sound, 65
ALSA, 64
CD/DVD burning applications
Brasero, 71
Linux command line, CD creation, 72–73

Linux command line, DVD creation, 73–75
Ubuntu support, 71
converting files, 65
.flac files, 65
.mp3 files, 65
.mp4 files, 65
music applications, 65
Banshee, 66
Rhythmbox, 66
Sound Juicer, 66
.ogg files, 65, 66
online references, 65
OSS, 64
PulseAudio, 64
recording/editing, 76
Ardour, 76
Audacity, 76
Cecilia, 76
LMMS, 76
Mixxx, 76
Rosegarden, 76
recording/editing applications, sound, 76
ripping audio, 71
sound cards, 63–64
sox command, 65
UNIX and, 63–64
.wav files, 65
Sound Juicer, 66
source, compiling software from, 100, 101–102
source tree (kernels), 403–405
sources (kernels), obtaining, 409–410
sox command, 65
spaces
embedded spaces, resolving strings with, 248–249
as special shell characters, 248
Spamassassin, 561
special characters
& operator, 173–174, 237, 247, 527
&& operators, combining commands, 180, 198
* special shell character, 247
\ special shell character, 249–250
` special shell character, 247, 250
[ ] special shell character, 248
[a, z] special shell character, 248
[a-z] special shell character, 248
[az] special shell character, 248
$, 235
$? built-in variable, 247
$ special shell character, 247
$# built-in variable, 246
$* built-in variable, 247
$0 built-in variable, 247
! ! operator, 196–197
!= operator, 491
< special shell character, 247
<< operator, 236
<< special shell character, 248
> special shell character, 247
>> special shell character, 247
| (pipe) special shell character, 247
| | operator, 198, 491
# special shell character, 247
? special shell character, 247
" special shell character, 248–249
' special shell character, 249
speed, relational databases, 492
Speed Dreams, 87
splint command, 602–603
Splunk, 299
spoofing attacks, 593
SQL, 487–488
!= operator, 491
MySQL, 483

ACID compliance, 493–494
command line client, 504–505
configuring, 494–495
creating databases, 496–497
data integrity, 493–494
data locking, 492
database access, 501–503
inserting data into tables, 490
mysql command, 506
mysqladmin command, 506
mysqldump command, 506
MySQLGUI, 506
online references, 493, 507
PostgreSQL versus, 492–494
procedural languages, 494
retrieving data from tables, 490–492
root user passwords, 495
speed, 492
SQL subqueries, 494
stored procedures, 494
triggers, 494
NoSQL databases, 485
PostgreSQL
ACID compliance, 493–494
command line client, 505–506
configuring, 498
creating database users, 499–500
creating databases, 499
data integrity, 493–494
data locking, 492
database access, 501–503
deleting database users, 500
granting/revoking privileges, 500–501
initializing data directories, 498
MySQL versus, 492–494
online references, 493, 507
pgAdmin, 506
procedural languages, 494
retrieving data from tables, 491
speed, 492
SQL subqueries, 494
stored procedures, 494
triggers, 494
subqueries, 494
tables
creating, 488–489
inserting data, 489–490
Squid, 563–564
ACL, 565–569
client IP addresses, 569–570
configuring, 564–565, 570–572
installing, 564
online references, 577
Squirrelmail, 561
SSH, database access, 501–502
ssh command, 369, 383
SSH keys, 638, 639
SSH server
configuring, SSH server, 371–372
copying files between machines, 372–373
ftp command, 372
key-based logins, 373–375
scp command, 372–373
sftp command, 373
ssh-keygen command, 373–375
ssh-keygen command, 373–375
stacks, 437
LAMP, 437–439
LEMP, 439
MEAN, 437
online references, 441
standard input/output of commands, 169
starting/stopping
Apache web server, 444–446
NFS, 422

services, 275–278
smbd daemon, 429–430
VM, 526
Startup Applications Preferences, 278
statements
break statements, 269
case statements, 267–268
exit statements, 269
expr statements, 266
for statements, 259–261
if statements, 265–266
repeat statements, 263
select statements, 264
shift statements, 264
for statements, 259–261
until statements, 263
while statements, 261, 262
statistics, user usage, 212
stderr command, 169–170
stdin command, 169–170
stdout command, 169–170
Steam, 81
stereotypes, users, 205
sticky bit permissions, 126–127
stopping/starting
Apache web server, 444–446
NFS, 422
services, 275–278
smbd daemon, 429–430
VM, 526
storage
backups
cloud storage, 309
external hard drives, 308
NAS, 308
network storage, 308
tape drive backups, 309
disks, performance tuning, 394
drives, Ubuntu installations, 9
temporary file storage, /tmp directories, 117
stored procedures
MySQL, 494
relational databases, 494
strings
command substitution, 250
comparisons, 251–252, 255–256
with embedded spaces, resolving, 248–249
input strings, searches, 151–152
su command, 217–219, 224
subnet masks, 337–338
subnetting, 337–338
subqueries (SQL), relational databases, 494
substitute user accounts, 217–218
substituting
commands, 250
processes, 181
Subversion, software project management, 612–613
sudo command, 14–15, 110–111, 134–136, 219–222, 224
sudoers command, 114
suid permissions, 125–127
super user accounts, 109–111, 133–134, 202–203, 217–218
SuperTux, 84–85
support, Ubuntu online references, 22
.svg files, 68
switches, 342
symbolic debugging, 603
symbols. *See also* wildcard characters
& operator, 237
background processes, running, 173–174
special shell character, 247
VirtualBox commands, 527

&& operators, combining commands, 180, 198
* special shell character, 247
\ special shell character, 249–250
` special shell character, 247, 250
[ ] special shell character, 248
[a, z] special shell character, 248
[a-z] special shell character, 248
[az] special shell character, 248
$, 235
$? built-in variable, 247
$ special shell character, 247
$# built-in variable, 246
$* built-in variable, 247
$0 built-in variable, 247
! ! operator, 196–197
!= operator, 491
< special shell character, 247
<< operator, 236
<< special shell character, 248
> special shell character, 247
>> special shell character, 247
| (pipe) special shell character, 247
| | operator, 198, 491
# special shell character, 247
? special shell character, 247
" special shell character, 248–249
' special shell character, 249
symlinks, 154–156
Synaptic, 92–94
synchronizing, filesystems, 396
Sysadmin, DevOps versus, 535–536
sysctl command, 398–399
sysrq.txt files, 404
System Monitor, 292
system resource management
APT, 95
apt-get versus, 99–100
day-to-day usage, 95–98
finding software, 98–99
apt-get, APT versus, 99–100
compiling software
from source, 100, 101–102
from tarballs, 100–101
from Ubuntu repositories, 101–102
configuration management, 102
dotdee, 102–103
Snappy package manager, 103–104
Ubuntu Core, 103
online references, 104
Software Updater, 94–95
Synaptic, 92–94
Ubuntu Software, 91
system settings, Ubuntu settings, configuring, 17–18
system user accounts, 203–204
systemd command, 275, 277–278, 532
service-related commands, 276–277
state-related commands, 276
targets, 275–276
systems
administration privileges, 217
listing information, 158–159
management tools, 292
Conky, 292–297
gnome-nettool, 297
System Monitor, 292
vncviewer, 297
Wireshark, 298
monitoring
checking log files, 287–289
Conky, 292–297
console-based monitoring, 281–283
disk quotas, 287
displaying free hard disk space, 286–287
displaying used memory, 286

enterprise servers, 298–299
graphical process tools, 292–298
KDE monitoring tools, 298
killing processes, 283–284
online references, 299
priority scheduling, 285
rotating log files, 289–291
system management tools, 292–298
System Monitor, 292
user activity, 211–212
password policies, 212
polite resets, 194–195
rebooting, 138–139
rescue, 323
restoring GRUB2 boot loaders, 323–324
saving files from nonbooting hard disks, 324
Ubuntu Rescue disc, 323
shutting down, 137–138
storing shell scripts for system-wide access, 240
SystemV runlevels, 275–276

T

tables (SQL)
creating, 488–489
inserting data, 489–490
tail command, 163–164
tape drive backups, 309
tar command, 192, 310
compressed files, 318
copying files, 317–318
encrypted files, 318
full backups, 310–311
incremental backups, 310–311
restoring files from archives, 311–312
tarballs, compiling software from, 100–101
tasks, scheduling, 226
automating tasks with scripts, 244–246
for later, 227–229
running jobs repeatedly, 229–231
waking computers from sleep, 231–233
tcpdump command, 385
TCP/IP networking, 330
addressing, 331–332
IP masquerading, 332–333
ports, 333–334
tcsh, comparisons of expressions, 226, 255
file operators, 257–258
logical operators, 259
numeric comparisons, 256–257
strings, 255–256
temporary file storage, /tmp directories, 117
terminals. *See also* command line
clients, X server, 33–34
multiple terminal operation, byobu, 192–193
testing
online references, 647
Samba, 429
Ubuntu
community teams, 645–646
Ubuntu Testing Teams, 646
testparm command, 429
TeX documents
Kile, 61
LaTeX, 60
LyX, 60
Texmaker, 60
Texmaker, 60
text
editors, 185, 186
awk, 189–191
emacs, 185, 188–189

gedit, 186
kate, 186
kedit, 186
nano, 185, 186–187
sed, 189–191
vi, 185, 187–188
vim, 185
repeating, 148–149
text-based logins, command line, 107–108
threads, 468
thttpd, 481–482
thumb drives, Ubuntu installations, 3, 6–11
Thunderbird, LDAP client configuration, 584
.tif files, 68
time warps, 212
time/date settings
date command, 19
hwclock command, 19–20
Ubuntu, 19–20
TLD, 588–589
/tmp directories, 117
Token Ring networking, 339
Tomcat, 482
top command, 175–177
Totem Movie Player, 76
touch command, 128–129
traceroute command, checking Internet connections, 329
traceroute6 command, checking Internet connections, 329
triggers, relational databases, 494
Tripwire, 384–385
troubleshooting
boot process, 278–279
bugs
Bug Squad, 647
Ubuntu development, 640–643
Internet connections, 368–369
kernels, 418–419
network printers, 434–435
networks, connections, 342
runlevels, 278
sudo command, 134–136
Ubuntu
bootups, 195–196
post-configuration problems, 21–22
tune2fs command, 396–397
Tuned Project, 396–400
TXT records, 593
type name command, 160

U

Ubuntu, 27
32-bit Ubuntu, 4–5
64-bit Ubuntu, 4–5
Android mobile development, 607
Android Runtime, 608
Android Studio, 609
Application Framework, 608–609
applications, 609
ARM processors, 608
Google Play, 607
Linux kernels, 608
RISC processors, 608
bash, 226
Bikeshed, 616–618
bootups, troubleshooting, 195–196
Budgie, 49
business applications, 27–28
C programming
autoconf command, 601–602
building programs, 599–601
checking source code, 602–603

configuring code, 601–602
debugging tools, 602–603
dependency checking, 599–600
GCC, 603–604
gdb command, 603
gprof command, 603
macros, 600–601
makefile targets, 600–601
makefiles, 599–600
splint command, 602–603
symbolic debugging, 603
tracking function time, 603
C++ programming
autoconf command, 601–602
building programs, 599–601
checking source code, 602–603
configuring code, 601–602
debugging tools, 602–603
dependency checking, 599–600
GCC, 603–604
gdb command, 603
gprof command, 603
macros, 600–601
makefile targets, 600–601
makefiles, 599–600
splint command, 602–603
symbolic debugging, 603
tracking function time, 603
CD/DVD burning applications, 71
Chaos Engineering, 614
CI/CD tools, 613–614
cloud computing, 535
AWS, 539
Azure, 539
considerations for using, 538
Google Cloud, 539
hybrid clouds, 539–540
IaaS, 537
install instructions, 536
Juju, 540–541, 545
Landscape, 541
MaaS, 537
Mojo, 541
OpenStack, 538–539
PaaS, 537
private clouds, 540
public clouds, 538, 539
reasons for using, 536–537
SaaS, 537
Ubuntu Cloud, 536
VM, 536
configuring
date/time settings, 19–20
power management, 18–19
system settings, 17–18
troubleshooting post-configuration problems, 21–22
wireless networks, 20
database commands, 506
date/time settings, 19–20
desktop environments, changing, 39
desktops, sharing, 424
development
Bazaar, 639
code packaging, 640–643
fixing bugs, 640–643
GPG keys, 637–639
introduction to, 636–637
Launchpad, 614–615, 638–640
local bash shells, 640
online references, 643
packages, 637–638, 640–643
pbuilder, 638
SSH keys, 638, 639
digital cameras, 70
distributions, versions of, 27

documentation, 28, 30
dual-boot systems, 6
DVD/CD burning applications, 71
finding files/folders, 12
graphical development tools, 605
  Eclipse, 606
  Glade, 607
  IDE, 605–606
  KDevelop Client, 606–607
  NetBeans, 606
  Oracle JDeveloper, 606
  SDK, 606
  Visual Studio Code, 606
home applications, 28
installing, 196
  32-bit Ubuntu, 4–5
  64-bit Ubuntu, 4–5
  backups, 7
  boot loaders, 5–6
  downloading updates during installation, 8
  dual-boot systems, 6
  DVD installations, 2, 6–11
  erasing hard disks during installation, 9
  flavors of Ubuntu, 2–3
  language selection, 7–8
  Mac hardware compatibility, 3
  partitions, 5, 9
  password creation, 10
  preinstallation process, 1–2
  researching hardware specifications, 2–16
  storage drives, 9
  UEFI, 6
  USB thumb drive installations, 3, 6–11
Java programming, 604
  JVM, 604
  OpenJDK, 604–605
kernels, 401–402
Kylin, 50
MATE, 48–49
networks, 330
  connectivity commands, 369
  ports, 421
  printers, 431–436
Nginx web server installations, 469
online references, 22
partitions, installing, 5, 9
passwords, 10
power management, configuring, 18–19
preinstallation process, 1–2
  flavors of Ubuntu, 2–3
  researching hardware specifications, 2–16
printers, configuring, 18
proxy servers. *See* Squid
QA
  Bug Squad, 647
  Ubuntu QA Teams, 646–647
reinstalling, 196
repositories, compiling software, 101–102
root prompt, dangers of working in, 15
scanners, 67
shutting down, 11–12
snap packages, 615–616
SnapCraft, 616
Snappy Playpen, 616
software project management
  Bazaar, 611–612
  Git, 610–611
  Subversion, 612–613
software repositories, configuring, 15–17
Software Updater, 12–14
storage drives, Ubuntu installations, 9
sudo command, 14–15
system settings, configuring, 17–18

testing
    community teams, 645–646
    Ubuntu Testing Teams, 646
troubleshooting, post-configuration problems, 21–22
Ubuntu Make, 615
Ubuntu Rescue disc, 323
updating
    first updates, 11
    during installation, 8
    software, 12–14
user account management, 223–224
version control systems, 609–610
virtualization, 521–523
    bridged networks, 524–525
    KVM, 523–526
    VirtualBox, 527–528
    VM, 522, 525–526, 536
web resources, 22
window managers, changing, 39
wireless networks, 20, 361–363
X server, 32–33
    benefits of, 33–34
    components of, 33–34
    display manager, 39
    display managers, 33
    distributed processing, 32–33
    online references, 40
    starting, 39
    terminal clients, 33–34
    /usr directory, 33
    window manager, 33–34
    xorg.conf files, 33–39
Ubuntu Budgie, 49
Ubuntu Core, 103
Ubuntu Kylin, 50
Ubuntu MATE, 48–49
Ubuntu QA Teams, 646–647
Ubuntu Software, 91
Ubuntu Testing Teams, 646
ubuntu-restricted-extras, 76
UEFI
    boot process, 273–274
    Ubuntu installations, 6
ufw command, 369
UFW, configuring, 386–387
UID, 204, 448
unexpanded variables, maintaining, 249
unicast addressing, 338
uninstalling Nginx web server, 470
Unison, 315
units (services), listing, 276
UNIX
    backup levels, 306
    CUPS GUI, network printers, 431–434
    sound, 63–64
UnQL, 511
until statements, 263
updating
    software, 12–14
    Software Updater, 94–95
    Ubuntu
        first updates, 11
        installations, downloading updates during installation, 8
        Software Updater, 12–14
uplink ports, 342
uptime command, 286
usage statistics, user accounts, 212
USB
    digital cameras, 70
    thumb drives, Ubuntu installations, 3, 6–11
used memory, displaying, 286
user accounts, 201–202, 204, 207
    adding accounts, 209–211

administration privileges, 217
BOFH, 205
changing identities, 217–219
creating, 136–137
deleting, 137
disk quotas, managing, 222–223
file permissions, 204
GID, 204
group management, 205
finding groups, 206
listing groups, 205–206
tools, 206–207
locking users out of accounts, 209
lusers, 205
monitoring user activity, 211–212
online references, 224
passwords, 202, 212, 216
changing in batches, 216–217
password files, 212–214
shadow passwords, 214–215
system password policies, 212
privileges, 109–111
root privileges, 219–222
root user accounts, 202–203
stereotypes, 205
substitute user accounts, 217–218
super user accounts, 109–111, 133–134, 202–203, 217–218
system user accounts, 203–204
tools, 208–209
Ubuntu commands, 223–224
UID, 204
usernames, 211
User directive (Apache web server), 448
user directories, 115
user variables, 242
useradd command, 208, 224
useradd -D command, 208
useradd -G command, 207
UserDir directive (Apache web server), 449
usermod command, 209, 224
usermod -G command, 207
users, productivity suites, 56
/usr directory
shared data, 117
X server, 33
UTP cabling, 341

V

Vala, 633
values, assigning to variables, 242
/var directory, 117
variable data files, /var directory, 117
variables
built-in variables, 242, 246–247
environment variables, 182–185, 242
shell scripts, 242
accessing values, 243
accessing/retrieving variables, 244
assigning values, 242
resolving strings with embedded spaces, 248–249
unexpanded variables, maintaining, 249
user variables, 242
VBA, LibreOffice and Office compatibility, 55
version control
configuration files, 320–322
systems, 609–610
vi text editor, 185, 187–188
video
.avi files, 75
dvd+rw-tools, 74
DVD/CD burning applications

Brasero, 71
Linux command line, CD creation, 72–73
Linux command line, DVD creation, 73–75
Ubuntu support, 71
editing, 76, 77
Avidemux, 77
Blender, 77
Cinelerra, 77
DaVinci Resolve, 77
Kdenlive, 77
Lightworks, 77
OpenShot Video Editor, 77
PiTiVi, 77
Shotcut, 77
.flv files, 75
.mov files, 75
.mpeg files, 75
.ogv/.ogg files, 75
.qt files, 75
Totem Movie Player, 76
ubuntu-restricted-extras, 76
video drivers (proprietary), installing, 80–81
viewing
in Linux, 76
video formats, 75
VLC, 76
.webm files, 75
viewing
command history, 197
default file permissions, 124–125
video
in Linux, 76
video formats, 75
vim text editor, 185
vimtutor command, 188
virt-clone, 526
virt-install, 525–526
virtual hosting
Apache web server, 461–462
Nginx web server, 473–474
VirtualBox, 527–528
virtualization, 521–523
bridged networks, 524–525
KVM, 523–526
online references, 529
VirtualBox, 527–528
VM, 522, 525–526, 536
connections, 526
copying, 526
starting, 526
stopping, 526
virt-clone, 526
virt-install, 525–526
vmbuilder, 525
VMware, 528
Xen, 528
virus scanners, email, 562
viruses, 385
Visual Studio Code, 606
visudo command, 135–136
VLC, 76
VM, 522, 536
connections, 526
copying, 526
KVM, 523–526
starting, 526
stopping, 526
virt-clone, 526
vmbuilder, 525
vmstat command, 286
VMware, 528
VNC, 375–377
vncviewer, 297

VPN, 563, 572–573
  configuring
    clients, 573–575
    servers, 575–577
  servers, online references, 577
vulnerability assessments, 381–382

W

waking computers from sleep, 231–233
war driving, 383
Warsow, 82
.wav files, 65
Wayland, 32
web browsers
  Chrome, 53–54
  Chromium, 53–54
  database access, 503
  Firefox, 52–53
  online references, 54
.webm files, 75
web resources
  Apache web server, 466
  backups, 324
  BIND, 595
  boot process, 278–279
  cloud computing, 541
  command line, 139, 165, 200
  containers, 534
  desktop environments, 50
  DNS, 595
  file sharing, 435
  games, 89
  HTTP servers, 482
  kernels, 419–420
  LDAP, 585
  Linux, 23, 28
  multimedia applications, 77
  MySQL, 493, 507
  network printers, 435
  networks, 369–370
  Nginx web server, 477
  NoSQL databases, 519–520
  performance tuning, 396–400
  PostgreSQL, 493, 507
  programming, 618–619
  programming languages, 633–634
  QA, 647
  remote access, 377–378
  security, 391–392
  sets of servers, 545
  shell scripts, 270
  software management, 104
  sound, 65
  Squid, 577
  system monitoring, 299
  system resource management, 104
  testing, 647
  Ubuntu, 22, 28, 30
  Ubuntu development, 643
  user accounts, 224
  virtualization, 529
  VPN, 577
  Wayland, 32
  web browsers, 54, 61
  web server stacks, 441
  X server, 40
web server stacks, 437
  LAMP, 437–439
  LEMP, 439
  MEAN, 437
  online references, 441

web servers
  Apache web server
    access control, 452, 455
    configuring, 446–449, 450–452
    development of, 443–444
    file system authentication, 452–455
    .htaccess configuration files, 450–452, 469
    HTTPS, 464–466
    installing, 444
    logging, 463–464
    modules, 455–461
    MPM, 449–450
    online references, 466
    overview of, 444
    runtime servers, 446–449
    security, 444, 452–455
    starting/stopping, 444–446
    threads, 468
    virtual hosting, 461–462
  C10K problem, 467–468
  event handlers, 468–469
  Nginx web server, 467–469
    building sources, 469–470
    configuring, 470–473
    HTTPS, 476–477
    installing, 469–470
    modules, 475
    online references, 477
    PHP, 474–475
    uninstalling, 470
    virtual hosting, 473–474
  threads, 468
wget command, 164
whatis command, 160
whereis command, 112, 160
which command, 164
while statements, 261, 262
wide column stores, 517
widgets, 42
wildcard characters. *See also* symbols
  commands and, 235
  directories, 133
Wildfly, 482
window managers
  changing, 39
  X server, 33–34
Windows
  CrossOver Office, 61
  dual-boot systems, 6
  games, 88–89
  productivity applications, 61
  Wine, 61
Wine, 61
wireless networks, 361
  configuring, 20
  interfaces, 340
  security, 382–383
  selecting protocols, 363
  Ubuntu support, 361–363
Wireshark, 298, 330
worms, 380
writing, shell scripts, 237–239
WYGIWYW, 60
WYSIWYG, 60

X

X server, 32, 33
  benefits of, 33–34
  components of, 33–34
  display manager, 39
  display managers, 33

  distributed processing, 32–33
  online references, 40
  starting, 39
  terminal clients, 33–34
  /usr directory, 33
  window manager, 33–34
  xorg.conf files, 33–34
    components of, 34–35
    Device section, 37–38
    Files section, 35–36
    InputDevice section, 36–37
    Module section, 36
    Monitor section, 37
    Screen section, 38–39
    ServerLayout section, 35
Xara Xtreme, 70
xconfig command, 414–417
xDSL Internet access, 365
Xen, 528
Xfce, 46
XML, 59
  DocBook, Publican, 59–60
  gedit, 59
  Publican, 59–60
  XML Copy Editor, 60
XML Copy Editor, 60
X.Org, 33
xorg.conf files, 33–34
  components of, 34–35
  Device section, 37–38
  Files section, 35–36
  InputDevice section, 36–37
  Module section, 36
  Monitor section, 37
  Screen section, 38–39
  ServerLayout section, 35
Xubuntu, 46–47
xz command, 192

Y

YAML, /etc/netplan/*.yaml files, 353–355
Yaws, 480

Z

Zabbix, 299
zsh, 226