

Save 10%
on Exam
Voucher

See Inside



Practice
Tests



Flash
Cards



Review
Exercises



Study
Planner

Cert Guide

Advance your IT career with hands-on learning

CompTIA®

Linux+

XK0-005



ROSS BRUNSON

FREE SAMPLE CHAPTER |



CompTIA® Linux+ XK0-005 Cert Guide

Companion Website and Pearson Test Prep Access Code

Access interactive study tools on this book's companion website, including practice test software, review exercises, Key Term flash card application, a study planner, and more!

To access the companion website, simply follow these steps:

1. Go to www.pearsonitcertification.com/register.
2. Enter the **print book ISBN: 9780137866885**.
3. Answer the security question to validate your purchase.
4. Go to your account page.
5. Click on the **Registered Products** tab.
6. Under the book listing, click on the **Access Bonus Content** link.

When you register your book, your Pearson Test Prep practice test access code will automatically be populated with the book listing under the Registered Products tab. You will need this code to access the practice test that comes with this book. You can redeem the code at **PearsonTestPrep.com**. Simply choose Pearson IT Certification as your product group and log in to the site with the same credentials you used to register your book. Click the **Activate New Product** button and enter the access code. More detailed instructions on how to redeem your access code for both the online and desktop versions can be found on the companion website.

If you have any issues accessing the companion website or obtaining your Pearson Test Prep practice test access code, you can contact our support team by going to pearsonitp.ehelp.org.

This page intentionally left blank

CompTIA® Linux+ XK0-005 Cert Guide

Ross Brunson



Pearson

Hoboken, New Jersey

CompTIA® Linux+ XK0-005 Cert Guide

Copyright © 2024 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: 978-0-13-786688-5

ISBN-10: 0-13-786688-7

Library of Congress Control Number: 2023913510

\$PrintCode

Trademarks

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

Warning and Disclaimer

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

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.

**Vice President,
IT Professional**
Mark Taub

**Director, ITP Product
Line Management**
Brett Bartow

Executive Editor
Nancy Davis

Development Editor
Ellie Bru

Managing Editor
Sandra Schroeder

Senior Project Editor
Mandie Frank

Copy Editor
Bill McManus

Indexer
Ken Johnson

Proofreader
Jennifer Hinchliffe

Technical Editor
Casey Boyles

Publishing Coordinator
Cindy Teeters

Designer
Chuti Prasertsith

Compositor
codeMantra

Pearson's Commitment to Diversity, Equity, and Inclusion

Pearson is dedicated to creating bias-free content that reflects the diversity of all learners. We embrace the many dimensions of diversity, including but not limited to race, ethnicity, gender, socioeconomic status, ability, age, sexual orientation, and religious or political beliefs.

Education is a powerful force for equity and change in our world. It has the potential to deliver opportunities that improve lives and enable economic mobility. As we work with authors to create content for every product and service, we acknowledge our responsibility to demonstrate inclusivity and incorporate diverse scholarship so that everyone can achieve their potential through learning. As the world's leading learning company, we have a duty to help drive change and live up to our purpose to help more people create a better life for themselves and to create a better world.

Our ambition is to purposefully contribute to a world where

- Everyone has an equitable and lifelong opportunity to succeed through learning
- Our educational products and services are inclusive and represent the rich diversity of learners
- Our educational content accurately reflects the histories and experiences of the learners we serve
- Our educational content prompts deeper discussions with learners and motivates them to expand their own learning (and worldview)

While we work hard to present unbiased content, we want to hear from you about any concerns or needs with this Pearson product so that we can investigate and address them.

Please contact us with concerns about any potential bias at <https://www.pearson.com/report-bias.html>.

This page intentionally left blank

Contents at a Glance

Introduction xliii

Part I: System Management (Obj. 1.1 - 1.7)

CHAPTER 1	Understanding Linux Fundamentals	3
CHAPTER 2	Managing Files and Directories	43
CHAPTER 3	Configuring and Managing Storage	101
CHAPTER 4	Managing Processes and Services	139
CHAPTER 5	Using Network Tools and Configuration Files	185
CHAPTER 6	Building and Installing Software	225
CHAPTER 7	Managing Software Configurations	281

Part II: Security (Obj. 2.1 - 2.5)

CHAPTER 8	Understanding Linux Security Best Practices	321
CHAPTER 9	Implementing Identity Management	353
CHAPTER 10	Implementing and Configuring Firewalls	379
CHAPTER 11	Using Remote Connectivity for System Management	405
CHAPTER 12	Understanding and Applying Access Controls	427

Part III: Scripting, Containers, and Automation (Obj. 3.1 - 3.5)

CHAPTER 13	Automating Tasks via Shell Scripting	463
CHAPTER 14	Perform Basic Container Operations	523
CHAPTER 15	Performing Basic Version Control Using Git	539
CHAPTER 16	Understanding Infrastructure as Code	573
CHAPTER 17	Understanding Containers, Cloud, and Orchestration	597

Part IV: Troubleshooting (Obj. 4.1 - 4.5)

CHAPTER 18	Analyzing and Troubleshooting Storage Issues	619
CHAPTER 19	Analyzing and Troubleshooting Network Resource Issues	667
CHAPTER 20	Analyzing and Troubleshooting CPU and Memory Issues	701

CHAPTER 21 Analyzing and Troubleshooting User and File Permissions 725

CHAPTER 22 Analyzing and Troubleshooting Common Problems Using Systemd 743

CHAPTER 23 Final Preparation 765

APPENDIX A Answers to the “Do I Know This Already?” Quizzes and Review Questions 773

Glossary 790

Index 809

Online Elements

APPENDIX B Study Planner

Table of Contents

Introduction xliii

Part I: System Management (Obj. 1.1 - 1.7)

Chapter 1 Understanding Linux Fundamentals 3

“Do I Know This Already?” Quiz 3

Foundation Topics 6

Filesystem Hierarchy Standard 6

What Goes Where in the FHS 6

The Root of the Filesystem 6

Where to Put Programs 7

Basic Boot Process 8

What Is the Boot Process? 8

System Boot Options 9

initrd and initramfs 9

Booting with UEFI 10

Booting via PXE 11

Booting via NFS 12

Booting via ISO 13

Boot Loaders and Files 14

GRUB Legacy 15

GRUB2 15

Changes Made for GRUB2 15

GRUB2 Command Names 16

Installing GRUB2 17

Using the GRUB2 Command Line 18

Demystifying Kernel Images 19

Configuring GRUB2 20

Common Commands at Boot Time 21

When Kernels Panic 22

Identifying a Kernel Panic 22

Getting More Information 22

Kernel Panic Causes 23

Device Types in /dev	23
Installing Software from Source	24
Components of a Source Code Install	26
The Makefile	26
Example of a Compilation of Source Code	27
Storage Concepts	28
Partitions	28
<i>MBR</i>	29
<i>GPT</i>	29
Data Storage Methods	29
<i>Block Storage</i>	30
<i>File Storage</i>	30
<i>Object Storage</i>	31
FUSE (Filesystem in Userspace)	31
<i>Why Use a FUSE?</i>	31
<i>Handling FUSE User Requests</i>	32
<i>Possible FUSE Applications/Uses</i>	32
RAID	32
<i>RAID Levels</i>	33
<i>Mirroring</i>	34
<i>Striping</i>	34
<i>Parity</i>	35
Listing Your Hardware Info	35
The Proc Filesystem	36
Notable ls* Commands	36
Friends of procs	37
The dmidecode Command	37
<i>Displaying DMI Table Information</i>	37
Summary	38
Exam Preparation Tasks	39
Review All Key Topics	39
Define Key Terms	40
Review Questions	40

Chapter 2	Managing Files and Directories	43
	“Do I Know This Already?” Quiz	43
	Foundation Topics	46
	File and Directory Operations	46
	Tips for Working with Linux Files	46
	Basic Navigation	47
	Advanced Navigation	48
	Listing Files and Directories	49
	Touching Files	50
	Copying Files and Directories	51
	Moving Objects	54
	Creating and Removing Directories	56
	Removing Objects	57
	File Metadata and File Types	57
	Determining File Types	57
	The file Command	57
	Displaying File Metadata	58
	The stat Command	58
	Soft and Hard Links	60
	Linking Files	60
	<i>Symbolic Links</i>	61
	<i>Hard Links</i>	62
	File Compression, Archiving, and Backup	63
	Using tar	64
	Using Compression Utilities	66
	<i>Using tar with Compression Utilities</i>	67
	<i>Taking Pity on the Unarchiver</i>	70
	<i>Useful Creation Options</i>	71
	Listing Archive Files	71
	Using cpio	72
	Using the dd Command	73
	Compression Tools	75
	Backing Up Is Hard to Do	76
	<i>Backup Types</i>	76
	Other Backup Types	77

Copying Objects Between Systems	78	
Using the scp Command	78	
Everything and the Kitchen rsync	79	
The nc Command	81	
File Editing	82	
The Message Line	83	
Editing in vim	83	
<i>Opening a File for Editing</i>	84	
<i>Navigating Within a File</i>	85	
<i>Force Multipliers</i>	86	
<i>Undo Operations</i>	86	
<i>Saving Files</i>	87	
<i>Quitting vim</i>	87	
<i>Changing or Replacing Text</i>	88	
<i>Deleting Text and Lines</i>	88	
Searching in vim	89	
Searching and Replacing	89	
nano, nano	90	
Why nano and Not Pico?	90	
nano's Interface	91	
That's a Wrap	92	
Feeling a Bit (awk)ward	92	
The printf Function	94	
Summary	95	
Exam Preparation Tasks	96	
Review All Key Topics	96	
Define Key Terms	97	
Review Questions	97	
Chapter 3	Configuring and Managing Storage	101
“Do I Know This Already?” Quiz		101
Foundation Topics		104
Determining Storage Hardware		104
Viewing SCSI Device Information		104
Viewing Partition and Filesystem Device File Information		105
<i>The lsblk Command</i>		105

<i>The blkid Command</i>	106
<i>The (Non-Linux) fdisk Command</i>	108
Performing Disk Partitioning	108
fdisk	108
Partitioning with parted	112
partprobe	113
Inspecting and Managing Software RAID	114
Creating and Inspecting a Software RAID	114
Inspecting a Software RAID Device	114
Using Logical Volume Manager (LVM)	114
Overview of LVM	115
Understanding Multipath	116
Managing LVM	117
Mounting Local and Remote Devices	118
The Filesystem Table	119
Manually Mounting Filesystems	121
Unmounting Filesystems	121
Using systemd to Mount	122
Linux Unified Key Setup (LUKS)	123
Monitoring Storage Space and Disk Usage	123
Using iostat	123
Using du	124
Using df	125
Filesystem Management	126
Filesystem Checker	126
EXT2/3/4 Tools Overview	127
Tuning ext Filesystems	128
Btrfs Tools	128
XFS Commands	130
Storage Area Networks/Network-Attached Storage	131
SAN vs. NAS	131
Multipathing with multipathd	132
Typical Network Filesystems Used by SAN/NAS	133
Samba vs. SMB and CIFS	133
Summary	134
Exam Preparation Tasks	135

Review All Key Topics	135
Define Key Terms	136
Review Questions	136
Chapter 4 Managing Processes and Services	139
“Do I Know This Already?” Quiz	139
Foundation Topics	142
Managing Processes	142
Viewing Processes	142
What’s the Diff?	143
The htop Command	144
The free Command	145
Blocks and Buffers	146
Pages, Slabs, and Caches	146
Interpreting Displayed Information from free	147
Sending Signals to Processes	148
Killing Processes by PID	149
Using pgrep and pkill to Send Signals	150
Killing Processes Using Other Criteria	152
Finding What Is Using a Resource	153
<i>Introducing lsof</i>	153
<i>Using lsof</i>	154
Job Control	155
Managing Process Priorities	157
systemd	159
What’s Different About systemd?	159
Units in systemd	161
<i>Unit File Directory Locations</i>	161
<i>systemd Environment Variables</i>	162
systemd Targets and Runlevels	163
Wants and Requires	163
Booting with systemd	164
Commands to Manage systemd	165
<i>The State of Services/Units</i>	165
<i>Listing Services</i>	166
<i>Enabling and Disabling Services</i>	166

	<i>Service Start, Stop, Restart, and Status</i>	167
	<i>Masking Services</i>	169
Scheduling Services		170
	Configuring crontabs	170
	<i>Using the crontab Command</i>	170
	<i>Matching Times</i>	171
	<i>Spelling Out Month and Day Names</i>	172
	<i>Making Multiple Matches</i>	172
	<i>Step Values</i>	172
	<i>Putting Together the crontab</i>	173
	<i>Issues About PATH</i>	173
	<i>Dealing with Output</i>	174
	<i>Nicknames</i>	175
	Other Files	175
	<i>System crontabs</i>	176
	<i>Convenience crontabs</i>	176
	<i>Restricting Access</i>	177
	<i>Running ad hoc Jobs</i>	178
	<i>The at Command</i>	178
	<i>The batch Command</i>	179
Summary		180
Exam Preparation Tasks		181
Review All Key Topics		181
Define Key Terms		182
Review Questions		182
Chapter 5	Using Network Tools and Configuration Files	185
	“Do I Know This Already?” Quiz	186
	Foundation Topics	188
	Interface Management	188
	iproute2	188
	Interface Management Commands	188
	<i>The ip Command</i>	189
	<i>Examples of the ip Command</i>	190
	<i>The ss Command</i>	191
	NetworkManager	192

<i>The nmtui Command</i>	192
<i>The nmcli Command</i>	192
The net-tools Suite	194
The ifconfig Command	195
Configuring via ifcfg-* files and network-scripts	195
<i>On Debian-Based Systems</i>	196
The hostname Command	197
The arp Command	198
The route Command	199
Name Resolution	199
Configuring Name Resolution	200
Controlling Resolution	202
Setting Hostnames on systemd Systems	204
The host, dig, and nslookup Commands	204
The whois Command	206
Network Monitoring	207
Is the Remote Host Reachable?	207
<i>The ping Command</i>	207
<i>The traceroute Command</i>	208
<i>The mtr Command</i>	210
Is the Data Flowing Properly?	211
Using netstat	211
<i>Swimming with Wireshark</i>	214
<i>The tcpdump Command</i>	216
Remote Networking Tools	217
SSH (Secure Shell)	217
Using curl and wget	218
The nc Command	219
Using rsync	221
Using scp	221
Summary	221
Exam Preparation Tasks	222
Review All Key Topics	222
Define Key Terms	223
Review Questions	223

Chapter 6	Building and Installing Software	225
	“Do I Know This Already?” Quiz	226
	Foundation Topics	228
	Package Management	228
	The Most Common Package Types	228
	Package Managers	229
	Debian Package Management	229
	<i>Managing Local Debian Packages</i>	230
	<i>Installing Packages with dpkg</i>	231
	<i>Removing Packages</i>	231
	<i>Dependency Issues</i>	232
	<i>Querying Packages</i>	233
	<i>Reconfiguring Packages</i>	234
	Using Repositories	235
	<i>What Is a Repository?</i>	236
	<i>Viewing Configured Repositories</i>	236
	<i>Adding Repositories</i>	238
	Installing Remote Packages	239
	<i>Working with the Cache</i>	241
	Upgrading the System	241
	Removing Packages	242
	Graphical Managers	242
	RPM and YUM Package Management	243
	The RPM Database	243
	RPM Package Files	244
	<i>Package Name Conventions</i>	244
	The rpm Command	245
	<i>Validation of Packages</i>	246
	<i>Installation of Packages</i>	246
	<i>Additional Installation Options</i>	247
	<i>Verifying a Package’s Integrity</i>	248
	<i>Freshening vs. Upgrading</i>	249
	Removing Packages	250
	<i>Other Removal Options</i>	251

<i>Querying Packages</i>	252
Package Management with YUM	255
<i>Installing Packages</i>	255
<i>Fetching Updates</i>	257
<i>Finding Packages to Install</i>	258
Configuring YUM	259
Dandified YUM	262
ZYpp	262
<i>Installing Software Packages with zypper</i>	263
<i>Removing a Package with zypper</i>	265
<i>Managing Repositories</i>	265
Sandboxed Applications	268
What Is a Sandboxed App?	268
App Sandbox Applications	269
<i>Flatpak</i>	269
<i>AppImage</i>	269
<i>Snapd</i>	270
System Updates	270
Updating the Kernel	270
Choosing an Update Method	271
Reboot Methods	271
<i>Manual Update</i>	271
<i>Update with Package Manager</i>	272
<i>Linux Kernel Utilities</i>	272
No Reboot Method	273
<i>Live Kernel Patching Overview</i>	274
<i>Issues with Live Patching</i>	274
<i>The Live Patch Process</i>	274
Summary	275
Exam Preparation Tasks	276
Review All Key Topics	276
Define Key Terms	277
Review Questions	277

Chapter 7	Managing Software Configurations	281
	“Do I Know This Already?” Quiz	281
	Foundation Topics	283
	Updating Configuration Files	283
	Restart or Reload?	283
	<i>Restarting a Service</i>	283
	<i>Reloading a Service</i>	283
	<i>Grace Under Pressure</i>	284
	Dealing with RPM Configurations	284
	<i>Rage About Your Machine</i>	285
	<i>Two Methods to Retain the Original Configuration File</i>	285
	<i>Handling .rpmsave and .rpmnew Files</i>	286
	Repository Configuration Files	287
	Repository Configuration File Overview	288
	<i>The APT Configuration File</i>	288
	<i>The YUM Configuration File</i>	288
	<i>The DNF Configuration File</i>	289
	Actual Repository Files	289
	<i>The YUM Repo Files</i>	289
	<i>The APT Repo Files</i>	289
	Configuring Kernel Options	289
	Viewing Kernel Parameters	290
	<i>Doing It the Manual Way</i>	290
	<i>Getting Used to Using sysctl</i>	291
	Ways to Set Kernel Parameters	291
	<i>Using the sysctl.conf File</i>	291
	<i>Using the sysctl Command Directly</i>	291
	<i>Using /etc/sysctl.conf</i>	292
	<i>Using the sysctl Command to Load Parameters</i>	292
	Understanding Kernel Modules	293
	Managing Kernel Modules	294
	Loading and Unloading Modules Manually	296
	The modprobe Command	298
	Configuring Common System Services	300
	Secure Shell (SSH)	301

Network Time Protocol (NTP)	301
<i>NTP Expressed Through Chrony</i>	302
The <code>timedatectl</code> Command	303
System Logging with Syslog	304
Representing Locales	304
Fallback Locales	306
Contents of a Locale	306
The <code>localectl</code> Command	307
How Linux Uses the Locale	307
systemd and syslog	308
<i>syslog</i>	309
<i>The logger Command</i>	312
<i>Configuring syslogd</i>	312
<i>Key File Locations</i>	313
<i>Other syslog Implementations</i>	314
Summary	315
Exam Preparation Tasks	316
Review All Key Topics	316
Define Key Terms	317
Review Questions	317

Part II: Security (Obj. 2.1 - 2.5)

Chapter 8 Understanding Linux Security Best Practices 321

“Do I Know This Already?” Quiz	321
Foundation Topics	323
Public Key Infrastructure	323
Purpose of Certificates	323
Certificate Authentication	323
Self-Signed Certificates	323
Certificate Authorities	323
Private Keys	324
Public Keys	324
Encryption and Hashing	324
Digital Signatures	325
Certificate Use Cases	325

Authentication	326
Multifactor Authentication	326
<i>Tokens</i>	326
<i>OTP</i>	327
<i>Biometrics</i>	327
LDAP	327
Pluggable Authentication Modules (PAMs)	327
<i>Password Policies</i>	328
<i>Password Length</i>	329
LDAP Integration	329
User Lockouts	329
The /etc/pam.d Directory	330
pam_tally2 and faillock	330
System Security Services Daemon	331
Single Sign-On (SSO)	332
Linux Hardening	333
The nmap Command	333
The nc Command	338
Secure Boot and UEFI	340
System Logging Configurations	340
Using umask	340
Disabling/Removing Insecure Services	342
Enforcing Password Strength	343
<i>Setting Password Parameters</i>	343
<i>Aging Your Passwords</i>	344
<i>No Wire Hangers (Group Passwords)</i>	345
Removing Unused Packages	345
Tuning Kernel Parameters	347
Securing Service Accounts	347
Configuring the Host Firewall	348
Summary	348
Exam Preparation Tasks	349
Review All Key Topics	349
Define Key Terms	350
Review Questions	350

Chapter 9 Implementing Identity Management 353

“Do I Know This Already?” Quiz	353
Foundation Topics	355
Account Creation and Deletion	355
User Account Fundamentals	355
What Accounts Are What?	355
<i>Regular User Accounts</i>	356
User Entries in <code>/etc/passwd</code>	357
<i>Special Login Files</i>	357
pam_tally2 and faillock	358
Group Accounts	358
<i>Group Entries in /etc/group</i>	360
<i>Group Passwords</i>	360
Adding Users and Groups	361
<i>Adding Users with useradd</i>	361
<i>useradd Defaults</i>	362
<i>skel Templates</i>	362
<i>Adding Groups with groupadd</i>	364
Modifying Users and Groups	364
<i>Modifying User Accounts with usermod</i>	364
<i>Modifying Groups with groupmod</i>	365
Removing Users and Groups	366
<i>Removing Users</i>	366
<i>Removing Groups</i>	367
The Shadow Suite	368
<i>Encrypted Passwords and Shadow Fields</i>	368
<i>/etc/shadow File Permissions</i>	369
Changing Passwords	370
<i>Aging Passwords</i>	370
<i>A Login Shell Session</i>	371
<i>A Non-Login Shell Session</i>	372
<i>User Identity Query Options</i>	372
Summary	374
Exam Preparation Tasks	375

Review All Key Topics	375
Define Key Terms	375
Review Questions	376
Chapter 10 Implementing and Configuring Firewalls	379
“Do I Know This Already?” Quiz	379
Foundation Topics	382
Common Firewall Technologies	382
iptables: Old and Reliable, but Complicated	383
nftables: Newer, Tighter, More Dynamic	383
<i>firewallld: Newer, Flexible, Easier to Use</i>	384
<i>UFW: Uncomplicated Indeed</i>	384
Understanding iptables	385
Overview of Filtering Packets	385
Important Terms	388
Using iptables to Filter Incoming Packets	389
Filtering by Protocol	391
Multiple Criteria	392
Filtering Based on Destination	392
Changing the Default Policy	393
Revisiting the Original Rules	394
Saving the Rules	394
Using iptables to Filter Outgoing Packets	395
Stateful Rules	396
Logging Rules	396
Implementing NAT	397
Additional Firewall Technologies	398
The fail2ban Service	398
DenyHosts	400
IPset	400
Summary	400
Exam Preparation Tasks	401
Review All Key Topics	401
Define Key Terms	401
Review Questions	402

Chapter 11 Using Remote Connectivity for System Management 405

- “Do I Know This Already?” Quiz 405
- Foundation Topics 408
- SSH (Secure Shell) 408
 - SSH Components 408
 - Tunneling 414
 - X11 Forwarding* 414
 - Port Forwarding* 415
- Executing Commands as Another User 416
 - The sudo Command 416
 - The sudoedit Command 417
 - User Privilege Escalation 418
 - The su Command 419
 - PolicyKit 420
 - The pkexec Command 420
- Summary 421
- Exam Preparation Tasks 422
- Review All Key Topics 422
- Define Key Terms 423
- Review Questions 423

Chapter 12 Understanding and Applying Access Controls 427

- “Do I Know This Already?” Quiz 427
- Foundation Topics 429
- File Permissions 429
 - Permission Trio Bits 429
 - Manipulating Permissions 432
 - Octal Mode 432
 - Symbolic Mode 433
- File and Directory Ownership 434
 - Changing File Ownership 435
 - Changing Group Ownership 436
- Understanding and Using umask 437
- Permission Granularity Issues 437
 - Special Bit Permissions 438

Setting the SUID Bit on Files	439
Setting the SGID Bit on Files	440
Setting the SGID Bit on Directories	441
Setting the Sticky Bit	442
Viewing and Changing File Attributes	442
<i>Displaying File Attributes</i>	442
<i>Key File Attributes</i>	443
<i>Setting File Attributes</i>	443
<i>Removing File Attributes</i>	443
Finding Files by Permission	444
Access Control Lists	445
Not Enough Granularity	445
ACLs to the Rescue	445
Viewing ACLs	446
Setting an ACL	446
We All Wear Masks	447
Context-Based Access	448
Security-Enhanced Linux (SELinux)	449
<i>SELinux Mode</i>	450
<i>SELinux Policy</i>	451
<i>SELinux Booleans</i>	452
<i>SELinux Contexts</i>	454
<i>The audit2allow Command</i>	455
AppArmor	456
<i>aa-disable Command</i>	456
<i>aa-complain Command</i>	457
<i>aa-unconfined Command</i>	457
<i>/etc/apparmor.d/ Directory</i>	457
<i>/etc/apparmor.d/tunables Directory</i>	457
Command-Line Utilities	457
Summary	457
Exam Preparation Tasks	458
Review All Key Topics	458
Define Key Terms	458
Review Questions	459

Part III: Scripting, Containers and Automation (Obj. 3.1 - 3.5)

Chapter 13 Automating Tasks via Shell Scripting 463

“Do I Know This Already?” Quiz	463
Foundation Topics	466
Shell Script Elements	466
Globbing	467
Environment Variables and Settings	469
<i>The PATH Variable</i>	471
<i>The SHELL Variable</i>	472
Variable Expansion	472
Running a Script	473
Good Script Design	474
Working with Input/Output Streams	475
Standard In	475
Standard Out	475
Standard Error	476
Find Errors on Demand	476
Here Documents	477
Redirection of Streams	478
Redirecting Standard Input	478
Redirecting Standard Output	478
Redirecting Standard Error	479
Redirection Redux	480
Understanding /dev/tty	480
Pipes	481
Executing Multiple Commands	483
<i>Multiple Command Operators</i>	483
<i>Command Substitution</i>	484
<i>Splitting Streams with the tee Command</i>	485
<i>Processing Output with the xargs Command</i>	485
Shell Script Elements	487
Using the Output of Another Command	487
Conditionals	488
Testing Files	490
An Easier Test Syntax	490

Testing Strings	491
Testing Integers	492
Combining Multiple Tests	493
case Statements	493
switch Statements	495
Loops	496
<i>For Loops</i>	496
<i>Sequences</i>	497
<i>while and until Loops</i>	498
Interacting with Other Programs	498
Returning an Error Code	499
Accepting Arguments	499
Feeling a Bit (awk)ward	500
Translating Files	502
Cutting Columns	502
He sed, She sed	503
Using grep and Friends	505
<i>Getting a grep</i>	505
<i>Examples of Using grep</i>	506
<i>Expanding grep with egrep and fgrep</i>	510
<i>Using Regular Expressions and grep</i>	511
<i>Pasting and Joining</i>	514
Finding Files	515
Summary	517
Exam Preparation Tasks	518
Review All Key Topics	518
Define Key Terms	519
Review Questions	519
Chapter 14 Performing Basic Container Operations	523
“Do I Know This Already?” Quiz	523
Foundation Topics	525
Container Management	525
Installing and Verifying the Container Tools	525
<i>Installing the Container-Tools Package and Dependencies</i>	525
<i>Verifying the Podman and Skopeo Tool Installation</i>	526

Finding and Pulling a Container Image	526
<i>Finding a Suitable Image</i>	526
<i>Pulling an Image</i>	527
Viewing and Inspecting Images	528
<i>Viewing Local Images</i>	528
<i>Inspecting a Local Image</i>	529
Running an Image as a Container	529
Assigning a Container-Friendly Name	529
Detaching from and Attaching to Containers	530
Exiting and Ending Execution of a Container	530
Removing a Container	531
Viewing Container Logs	531
Exposing and Mapping Ports	532
Container Image Operations	533
build Command	533
push Command	533
pull Command	534
list Command	534
rmi Command	534
Summary	534
Exam Preparation Tasks	535
Review All Key Topics	535
Define Key Terms	535
Review Questions	536
Chapter 15 Performing Basic Version Control Using Git	539
“Do I Know This Already?” Quiz	539
Foundation Topics	541
Version Control Concepts	541
The First Generation	541
The Second Generation	542
<i>What Is a Merge?</i>	542
The Third Generation	543
Using Git for Version Control	546
Installing Git	546

Git Concepts and Features	547
<i>Git Stages</i>	548
<i>Choosing Your Git Repository Host</i>	549
<i>Configuring Git</i>	549
<i>Using git tag</i>	552
<i>Getting the Status of Files</i>	553
<i>The .git Directory</i>	555
<i>Telling Git to Ignore a File</i>	555
Handling Branches	556
Executing Diffs	558
<i>Comparing Versions</i>	560
<i>Dealing with Whitespace</i>	560
<i>Comparing Branches</i>	561
Merging Files	562
Summary	568
Exam Preparation Tasks	569
Review All Key Topics	569
Define Key Terms	569
Review Questions	570
Chapter 16 Understanding Infrastructure as Code	573
“Do I Know This Already?” Quiz	573
Foundation Topics	576
File Formats	576
YAML	576
<i>Key Characteristics of YAML</i>	576
<i>YAML Stream Examples</i>	576
JSON	577
<i>Key Characteristics of JSON</i>	578
<i>JSON Examples</i>	578
Infrastructure as Code Concepts	579
Just Making a Config Change	579
Using Source Control	580
Getting Started with IaC	580
Utilities: Infrastructure as Code	580

IaC Utility Choices	581
Ansible	581
Puppet and Chef	583
<i>Puppet</i>	583
<i>Chef</i>	583
SaltStack	584
<i>Salt Architecture</i>	584
<i>Salt Master Options</i>	584
<i>Salt Configuration Locations</i>	585
<i>Salt State Files</i>	585
Terraform	586
<i>Coding</i>	586
<i>Planning</i>	587
<i>Applying</i>	588
Continuous Integration/Continuous Deployment	588
How Can It Be Solved?	588
Continuous Integration	588
Continuous Delivery	589
CI/CD Use Cases	589
Advanced Git Topics	590
merge	590
rebase	590
Pull Requests	591
Summary	591
Exam Preparation Tasks	592
Review All Key Topics	592
Define Key Terms	593
Review Questions	593
Chapter 17 Understanding Containers, Cloud, and Orchestration	597
“Do I Know This Already?” Quiz	597
Foundation Topics	600
Kubernetes Benefits and Application Use Cases	600
What Is Kubernetes, Really?	600
The High-Level Structure of Kubernetes	601

Kubernetes Control Plane Structure	601
Kubernetes Node Structure	602
Container Runtime Interface	602
Specialized Container Types	603
Benefits of Using Kubernetes	603
Single-Node Multicontainer Use Cases	604
Two-Container Node Example A	604
Two-Container Node Example B	605
Three-Container Node Example	605
Container Persistent Storage	605
Docker Volumes	605
<i>Advantages of Using Docker Volumes</i>	606
<i>Disadvantages of Using Docker Volumes</i>	606
Bind Mounts	606
<i>Advantages of Using Bind Mounts</i>	607
<i>Disadvantages of Using Bind Mounts</i>	607
Kubernetes Persistent Volumes	607
<i>The Kubernetes PersistentVolume Subsystem</i>	607
<i>The PV and PVC Lifecycles</i>	608
Container Networks	608
What's an Overlay Network?	608
Docker Swarm	608
Kubernetes Overlay Networks	609
Bridging Networks	609
Swotting a NAT	610
Host Networking	610
Service Mesh	611
What Is a Service Mesh?	611
Example Service Mesh	611
Istio Components	612
Commercial Istio Distributions	612
Non-Istio Service Meshes	612
Bootstrapping	612
Container Registries	613

- What Is a Container Registry? 614
- What About Bigger Teams? 614
- What Are My Container Registry Options? 614
- Summary 614
- Exam Preparation Tasks 615
- Review All Key Topics 615
- Define Key Terms 616
- Review Questions 616

Part IV: Troubleshooting (Obj. 4.1 - 4.5)

Chapter 18 Analyzing and Troubleshooting Storage Issues 619

- “Do I Know This Already?” Quiz 619
- Foundation Topics 623
- High Latency Issues 623
 - High Latency Overview 623
 - Causes and Symptoms of High Latency 623
 - Diagnosing and Fixing High Latency 624
 - Diagnosing with the top Command* 624
 - Diagnosing with the vmstat Command* 625
 - Fixing the Problem* 626
- Low Throughput Issues 627
 - Low Throughput Overview 627
 - Causes and Symptoms of Low Throughput 627
 - Diagnosing and Fixing Low Throughput 628
 - Diagnosing with the df Command* 628
 - Diagnosing with the iostat Command* 629
- Input/Output Operations per Second Scenarios 631
 - IOPS Overview 632
 - Scenario 1: Transferring Ten 500MB Files* 632
 - Scenario 2: Transferring 5,000 1MB Files* 632
 - Why IOPS Are Effectively Irrelevant* 632
- Capacity Issues 633
 - Causes and Symptoms of Capacity Issues 633
 - Diagnosing and Fixing Capacity Issues 634
 - Diagnosing with the df Command* 634

<i>Diagnosing with the du Command</i>	636
<i>Diagnosing with the find Command</i>	637
Filesystem Issues	638
Filesystem Corruption Overview	638
Causes and Symptoms of Filesystem Corruption	639
Diagnosing and Fixing Filesystem Corruption	639
<i>Diagnosing and Fixing Filesystem Corruption with the fsck Command</i>	640
<i>Summary</i>	642
Filesystem Mismatch Overview	642
Causes and Symptoms of File Mismatch	642
Diagnosing and Fixing File Mismatch	642
<i>Diagnosing a Mismatch Issue</i>	642
<i>Fixing a Mismatch Issue</i>	643
I/O Scheduler Issues	643
I/O Scheduler Overview	644
Types of I/O Schedulers	644
Viewing and Setting I/O Schedulers	645
<i>Viewing the Current I/O Scheduler</i>	645
<i>Setting the Current I/O Scheduler</i>	646
<i>Making the I/O Scheduler Change Persistent</i>	646
Device Issues	647
NVMe Issues Overview	647
Causes and Symptoms of NVMe Issues	647
Diagnosing and Fixing NVMe Issues	648
<i>Using the nvme Command-Line Tool</i>	648
Solid-State Drive Issues Overview	649
Causes and Symptoms of SSD Issues	649
Diagnosing and Fixing SSD Issues	650
SSD Trim	651
<i>Garbage Collection</i>	651
<i>The TRIM Helper</i>	652
RAID Issues	652
Causes and Symptoms of RAID Failures	653
Diagnosing and Fixing RAID Failures	653
<i>Overview of Tools</i>	653

<i>Monitoring and Alerting with smartd</i>	654
<i>Checking Device Health with smartctl</i>	654
<i>Monitoring RAID Array Health</i>	655
<i>Monitoring RAID Array Health with mdadm</i>	656
LVM Issues	656
LVM Troubleshooting Overview	656
Causes and Symptoms of LVM Issues	657
Diagnosing and Fixing LVM Issues	657
Mount Option Issues	659
Mounting Options Overview	659
Causes and Symptoms of Mount Options Issues	659
Diagnosing and Fixing Mount Option Issues	660
<i>Understanding Mount Option Gotchas</i>	660
<i>What Are All These UUIDs in My fstab?</i>	660
<i>What Is This errors= Option?</i>	661
Exam Preparation Tasks	663
Review All Key Topics	663
Define Key Terms	664
Review Questions	664
Chapter 19 Analyzing and Troubleshooting Network Resource Issues	667
“Do I Know This Already?” Quiz	667
Foundation Topics	670
Network Configuration Issues	670
Causes and Symptoms of Network Configuration Issues	670
Diagnosing and Fixing Network Configuration Issues	670
<i>Diagnosing Subnets with the ip Command</i>	670
<i>Diagnosing Routes Using the ping Command</i>	672
<i>Diagnosing Routes Using the ip route Command</i>	672
<i>Diagnosing Routes Using the traceroute Command</i>	673
Firewall Issues	674
Firewall Refresher	674
What Could Possibly Go Wrong?	675
Causes and Symptoms of Firewall Issues	675
Diagnosing and Fixing Firewall Issues	676

<i>Diagnosing with the ping Command</i>	677
<i>Diagnosing with the telnet Command</i>	677
<i>Making Sure Services Are Running</i>	678
Interface Errors	679
Dropped Packets and Collisions	680
Diagnosing and Fixing Issues Related to Dropped Packets and Collisions	680
<i>Diagnosing with the ethtool Command</i>	681
<i>Diagnosing with the netstat Command</i>	681
<i>Diagnosing with the /sys/class/net/<device>/statistics Information</i>	682
<i>Wrapping Up Dropping Packets</i>	683
Link Status	683
Diagnosing and Fixing Link Status Issues	684
<i>Layer 1: Physical</i>	684
<i>Layer 2: Data Link</i>	685
Bandwidth Limitations	686
Bandwidth and Latency	686
Diagnosing and Fixing Bandwidth Limitations	686
Name Resolution Issues	687
The Trifecta of DNS	687
Diagnosing and Fixing Name Resolution Issues	687
Testing Remote Systems	689
How (Not) to Break the Law	689
Purposes of Testing a Remote System	689
Testing Remote Systems with NMAP	690
<i>Running a Simple System Scan</i>	690
<i>Running a Service Discovery Scan</i>	691
<i>Running a Vulnerability Scan</i>	692
Testing Remote Systems with s_client	693
Verifying SSL Connection to a Remote System	693
Other Useful s_client Commands	694
<i>Commands to Use when Connected to a Host</i>	695
<i>Opening an SSL Connection to a Host</i>	695
<i>Showing SSL Certificates on a Host</i>	695
<i>Testing a Particular TLS Version on a Host</i>	696

Summary	696
Exam Preparation Tasks	697
Review All Key Topics	697
Define Key Terms	698
Review Questions	698
Chapter 20 Analyzing and Troubleshooting CPU and Memory Issues	701
“Do I Know This Already?” Quiz	701
Foundation Topics	704
Runaway and Zombie Processes	704
Runaway Processes	704
<i>What Causes Runaway Processes?</i>	704
<i>Reserving Some CPU for Non-Realtime Processes</i>	704
<i>Identifying a Runaway Process</i>	705
<i>Ending a Runaway Process</i>	705
Zombie Processes	705
<i>What Causes Zombie Processes?</i>	706
<i>Are Zombie Processes Bad?</i>	706
<i>Removing Zombie Processes</i>	706
High CPU Utilization/Load Average/Run Queues	707
High CPU Utilization	707
High Load Average	707
<i>Viewing System Load Details with the uptime Command</i>	708
High Run Queues	708
<i>Viewing System Load Details with the tload Command</i>	710
CPU Times and CPU Process Priorities	711
Measuring CPU Time	711
Important CPU Time Terms	712
CPU Process Priorities	713
Memory Exhaustion and Out of Memory	713
What Is Out of Memory (OOM)?	714
<i>Memory Leaks</i>	716
<i>The Process Killer</i>	716
Swapping	717
Swap, Caching, and Buffers	717

Blocks and Buffers	718
Pages, Slabs, and Caches	718
How Much Swap Is Enough?	719
Hardware	719
Viewing CPU Info	719
Viewing Memory Info	720
Summary	720
Exam Preparation Tasks	721
Review All Key Topics	721
Define Key Terms	721
Review Questions	721
Chapter 21 Analyzing and Troubleshooting User and File Permissions	725
“Do I Know This Already?” Quiz	725
Foundation Topics	728
User Login Issues	728
Inspecting Account Details	728
<i>ID Please</i>	728
<i>Get Entity</i>	729
Case-Sensitivity	730
Has the User Ever Logged In?	730
<i>The last Command</i>	730
<i>The lastlog Command</i>	730
User File Access Issues	731
Group Issues	732
Context Issues	732
Permission Issues	732
ACL Issues	733
Attribute Issues	733
Policy/Non-Policy Issues	734
Password Issues	735
The faillog Command	735
/etc/security.access.conf	736
Privilege Elevation Issues	736
Quota Issues	736

Possible Files or Entirety of Blocks? 737

Converting to Usable Numbers 737

Summary 738

Exam Preparation Tasks 739

Review All Key Topics 739

Define Key Terms 739

Review Questions 739

Chapter 22 Analyzing and Troubleshooting Common Problems Using Systemd 743

“Do I Know This Already?” Quiz 743

Foundation Topics 745

Unit Files 745

Service Unit File Issues 745

Networking Services 746

ExecStart and ExecStop 747

Before and After 747

Type 748

User 749

Requires and Wants 749

Timer Unit File Issues 750

OnCalendar 750

OnBootSec 750

Unit 750

Time Expressions 750

Mount Unit File Issues 750

Naming Conventions 751

What 751

Where 751

Type 751

Options 752

Target Unit File Issues 752

Default 752

Multiuser 754

Network-online 754

Graphical 754

	Common Problems	756
	Name Resolution Failure	756
	Application Crash	756
	Time-Zone Configuration	759
	Boot Issues	760
	Journal Issues	760
	Services Not Starting on Time	760
	Summary	761
	Exam Preparation Tasks	762
	Review All Key Topics	762
	Define Key Terms	762
	Review Questions	763
Chapter 23	Final Preparation	765
	Exam Information	765
	Getting Ready	767
	Tools for Final Preparation	768
	Pearson Test Prep Practice Test Software and Questions on the Website	768
	<i>Accessing the Pearson Test Prep Practice Test Software Online</i>	769
	<i>Accessing the Pearson Test Prep Practice Test Software Offline</i>	769
	Customizing Your Exams	770
	Updating Your Exams	771
	<i>Premium Edition</i>	771
	Chapter-Ending Review Tools	772
	Suggested Plan for Final Review/Study	772
	Summary	772
Appendix A	Answers to the “Do I Know This Already?” Quizzes and Review Questions	773
	Glossary	790
	Index	809
Online Elements:		
Appendix B	Study Planner	

About the Author

Ross Brunson has more than 30 years of experience as a Linux and open-source trainer, training manager, and certification architect, and is the author of the now-classic *LPIC-1 Exam Cram 2*, several iterations of the *CompTIA Linux+ Cert Guide*, and dozens of technical courses for major organizations.

Ross is currently the Education Architect at Grafana Labs (www.grafana.com), where he focuses on building a learning framework and training offerings that help employees and customers make the best use of Grafana to observe, troubleshoot, and maintain their environments.

Previously, Ross was a Senior Technical Training Engineer for NGINX, where he completely redid the Fundamentals learning track, authored a number of Getting Started guides, and taught a number of customer engagements to help new NGINX customers take full advantage of the platform.

Before NGINX, Ross enjoyed a few years at Linux Academy/A Cloud Guru where as a Senior Training Architect, he authored the SUSE Certified Administrator and Engineer courses, did the Red Hat Certified System Administrator Labs, created many additional courses on systemd, VIM and the screen command, and wrote and reviewed way too many exam questions.

Ross has also put in a tour of duty as the Certification Architect at SUSE, where he helped redesign and modernize the entire certification program. He has also spent five years as the Director of Member Services for the Linux Professional Institute, where he contributed to placing several LPI courses into the Cisco Networking Academy, conducted dozens of train-the-trainer sessions, and provided sales enablement support for the worldwide Master Affiliate network, spanning more than 100 countries and nearly a million certified professionals.

Ross holds a number of key IT certifications and is author of several successful technical books and dozens of technical courses for major organizations (including the first U.S. LPI Certification Bootcamps). He is skilled at both contributing to and building community around IT products.

Ross lives in Paradise Valley, Montana, with his family and enjoys traveling far and wide, participating in hiking, winter sports, photography, and playing the drums with great vigor (although not everyone around him appreciates it).

Dedication

My heartfelt thanks to all of my mentors and friends who have helped me get where I am: Andres Fortino, Arnold Villeneuve, Ken Haug, Ted Jordan, Edward Denzler, and many more. I am eternally grateful for the love and support of my wife and daughter, who understand what it means when “daddy is writing” and still love me anyway.

I also want to shout out to all our previous edition readers who made this book likely and possible. I love the emails and photos of you all and your certifications; it really makes a difference when we get something that lets us know we are somehow making even a tiny difference in someone’s career and life.

I want to hear from YOU. Let me know what you liked, what I can improve, and how you’re doing. Please send pics of you and your study tools, you and your certification, and so on...

—Ross E. Brunson, July 2023

Acknowledgments

This book is a result of the concerted efforts of many dedicated people, without whom this book would not be a reality. I would like to thank the technical reviewer, Casey Boyles, whose efforts and patience made this a better book for all to use, and to Chris Cleveland, who helped me navigate the adjustments to new CompTIA Linux+ versions over the years.

Much thanks to William (Bo) Rothwell for the courseware and writing MACHINE that he is, and for being a great author and technical editor over these many years—I couldn’t have done it without you, buddy!

Thanks also to Nancy Davis, Executive Editor, for her help and continuous support during the development of this book. I wish to also express my appreciation to Mary Beth Ray, executive editor at Pearson/Cisco Press (retired), for her confidence in me throughout years of working on book projects.

Much thanks to Ellie Bru for both her superb editorial skills and acumen, but especially her good humor and geek-wrangling skills; it is a pleasure to work with her on every book!

In addition, many thanks to Dr. James Stanger for being such a great supporter of the world of Linux and open source. He’s a good friend and a hugely relevant person in the world of getting our customers and attendees the skills they need!

It has been a huge undertaking to pull together all the pieces of this project. It is due to the dedication of those mentioned above that this book is not only large in scope but high in quality. It is my sincerest hope that our combined efforts will help you, the readers and users of this book, achieve your goals in an IT career.

About the Technical Reviewer

Casey Boyles started working in the IT field more than 28 years ago and quickly began to work with distributed application and database development. Casey later moved on to technical training and development; he specializes in full stack Internet application development, database architecture, and systems security. Casey typically spends his time smoking cigars while “reading stuff and writing stuff.”

We Want to Hear from You!

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

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

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

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

Email: community@informat.com

Reader Services

Register your copy of *CompTIA Linux+ XK0-005 Cert Guide* at www.pearsonitcertification.com for convenient access to downloads, updates, and corrections as they become available. To start the registration process, go to www.pearsonitcertification.com/register and log in or create an account*. Enter the product ISBN 9780137866885 and click Submit. When the process is complete, you will find any available bonus content under Registered Products.

*Be sure to check the box that you would like to hear from us to receive exclusive discounts on future editions of this product.

Introduction

In mid-2022, CompTIA released a new version of the Linux+ certification exam, labeled XK0-005. To throw a monkey wrench is to goof up or confuse or sabotage. The gears of other content authors may be messed up by this change, but I remain unaffected.

In particular, the new exam version has more DevOps-related and cloud-specific topic areas, allocates more space to Git and containers, and adds the revised Troubleshooting domain, which covers significant space on the exam and comprises approximately 28% of the scoring.

Most of the Linux+ exam will be multiple choice, much like the previous exams. However, you should also be prepared for a handful of scenario questions in which you will be asked to answer some questions based on a particular situation. In addition, you'll encounter some simulation questions, where you're running what appears to be a command-line terminal and you have to answer the question by actually typing the right commands and so forth. (Please note, you can use the **commandname help** option for *all* of these simulation questions, which will really help you puzzle out what the questions require!)

Use this book as a reference to all of the key exam-testable topics. This book provides an excellent roadmap on your journey to learning Linux and passing the Linux+ certification exam.

Study hard and study well. Pore over the exam objectives, and if you don't know something, I guarantee that you will see it on the exam, so make sure to locate the topic or term in this book's TOC or index and read the relevant material.

Good luck!

—Ross E. Brunson, July 5, 2023

Goals and Methods

The number-one goal of this book is a simple one: to help you pass the CompTIA Linux+ XK0-005 certification exam.

Because the CompTIA Linux+ certification exam now stresses problem-solving abilities and reasoning more than memorization of terms and facts, my goal is to help you master and understand the required objectives for the exam.

To aid you in mastering and understanding the Linux+ certification exam objectives, this book uses the following methods:

- **Opening topics list:** The list at the beginning of each chapter defines the topics to be covered in the chapter, followed by identification of the corresponding CompTIA Linux+ objective.
- **Foundation Topics:** The body of the chapter explains the topics from both hands-on and theory-based standpoints, including in-depth descriptions, tables, and figures that help you build your knowledge so that you can pass the Linux+ exam. The chapters are broken down into several topics each.
- **Key Topics:** Key Topics icons indicate important figures, tables, and lists of information that you should know for the exam. They are interspersed throughout the chapter and are listed in table format at the end of the chapter.
- **Key Terms:** Key terms without definitions are listed at the end of each chapter. Write down the definition of each term and check your work against the key terms in the glossary.
- **Review Questions:** These quizzes and answers with explanations are meant to gauge your knowledge of the subjects covered in the chapter. If an answer to a question doesn't come readily to you, be sure to review that portion of the chapter.
- **Practice Exams:** The practice exams are included in the Pearson Test Prep practice test software. These exams test your knowledge and skills in a realistic testing environment. Take them after you have read through the entire book. Master one, then move on to the next.

The Linux+ Domains and Objectives

The Linux+ XK0-005 exam consists of the following domains and objectives:

1.0 System Management (32% of the exam)

- 1.1 Summarize Linux fundamentals
- 1.2 Given a scenario, manage files and directories
- 1.3 Given a scenario, configure and manage storage using the appropriate tools
- 1.4 Given a scenario, configure and use the appropriate processes and services
- 1.5 Given a scenario, use the appropriate networking tools or configuration files
- 1.6 Given a scenario, build and install software
- 1.7 Given a scenario, manage software configurations

2.0 Security (21% of the exam)

- 2.1 Summarize the purpose and use of security best practices in a Linux environment
- 2.2 Given a scenario, implement identity management
- 2.3 Given a scenario, implement and configure firewalls
- 2.4 Given a scenario, configure and execute remote connectivity for system management
- 2.5 Given a scenario, apply the appropriate access controls

3.0 Scripting, Containers, and Automation (19% of the exam)

- 3.1 Given a scenario, create simple shell scripts to automate common tasks
- 3.2 Given a scenario, perform basic container operations
- 3.3 Given a scenario, perform basic version control using Git
- 3.4 Summarize common infrastructure as code technologies
- 3.5 Summarize container, cloud, and orchestration concepts

4.0 Troubleshooting (28% of the exam)

- 4.1 Given a scenario, analyze and troubleshoot storage issues
- 4.2 Given a scenario, analyze and troubleshoot network resource issues
- 4.3 Given a scenario, analyze and troubleshoot central processing unit (CPU) and memory issues
- 4.4 Given a scenario, analyze and troubleshoot user access and file permissions
- 4.5 Given a scenario, use systemd to diagnose and resolve common problems with a Linux system.

Be sure to visit CompTIA's web page at <https://certification.comptia.org/certifications/linux> to ensure that you have the latest information for the CompTIA Linux+ exam.

How This Book Maps to the Exam Objectives

All exam objectives are covered in this book and each chapter is devoted to a specific exam objective. But, in the interest of presenting a logical learning path, the order of the content in each chapter does not exactly match the order of the topics listed within the corresponding objective. To help you focus on the exam objectives for

which you might need some additional learning and preparation, this table shows you which chapters cover the various exam objectives:

Chapter	Exam Objective(s) Covered
Chapter 1, “Understanding Linux Fundamentals”	1.1
Chapter 2, “Managing Files and Directories”	1.2
Chapter 3, “Configuring and Managing Storage”	1.3
Chapter 4, “Managing Processes and Services”	1.4
Chapter 5, “Using Network Tools and Configuration Files”	1.5
Chapter 6, “Building and Installing Software”	1.6
Chapter 7, “Managing Software Configurations”	1.7
Chapter 8, “Understanding Linux Security Best Practices”	2.1
Chapter 9, “Implementing Identity Management”	2.2
Chapter 10, “Implementing and Configuring Firewalls”	2.3
Chapter 11, “Using Remote Connectivity for System Management”	2.4
Chapter 12, “Understanding and Applying Access Controls”	2.5
Chapter 13, “Automating Tasks via Shell Scripting”	3.1
Chapter 14, “Performing Basic Container Operations”	3.2
Chapter 15, “Performing Basic Version Control Using Git”	3.3
Chapter 16, “Understanding Infrastructure as Code”	3.4
Chapter 17, “Understanding Containers, Cloud, and Orchestration”	3.5
Chapter 18, “Analyzing and Troubleshooting Storage Issues”	4.1
Chapter 19, “Analyzing and Troubleshooting Network Resource Issues”	4.2
Chapter 20, “Analyzing and Troubleshooting CPU and Memory Issues”	4.3
Chapter 21, “Analyzing and Troubleshooting User and File Permissions”	4.4
Chapter 22, “Analyzing and Troubleshooting Common Problems Using Systemd”	4.5

Book Features

To help you customize your study time using this book, the core chapters have several features that help you make the best use of your time:

- **Foundation Topics:** These core sections of each chapter explain the concepts that are important to the chapter.
- **Exam Preparation Tasks:** This section lists a series of study activities that you should do at the end of the chapter:
 - **Review All Key Topics:** The Key Topic icon appears next to the most important items in the “Foundation Topics” section of the chapter. The “Review All Key Topics” activity lists the key topics from the chapter, along with their page numbers. Although the contents of the entire chapter could be on the exam, you should definitely know the information listed in each key topic, so be sure to review them.
 - **Define Key Terms:** Although the Linux+ exam is unlikely to ask an open-ended question such as “Define this term,” the exam does require that you learn and know a lot of industry-related terminology. This section lists the most important terms from the chapter, asking you to write a short definition and compare your definition to the glossary definition at the end of the book.
 - **Review Questions:** Confirm that you understand the content that is covered in the chapter by answering these questions and reading the answer explanations.
- **Web-Based Practice Exams:** The companion website includes the Pearson Cert IT certification test engine, which allows you to take practice exams. Use it to prepare with a sample exam and to pinpoint topics where you need more study.

What’s New?

If you are used to the objectives of the older Linux+ exam and the content of the previous version of this book, you should read the following which describes how both the exam objectives and the layout of this book have changed.

For more information about how the CompTIA Linux+ certification can help your career or to download the latest official objectives, access CompTIA’s Linux+ web page at <https://www.comptia.org/certifications/linux>.

As the Linux+ objectives are now presented in an order that makes sense from a learning perspective, this book is patterned with each chapter taking on an objective topic in its entirety. (Thanks, CompTIA!)

However, as a long-time technical instructor who likes things to make sense even within a chapter, I have taken some liberties with the in-chapter order of each objective's contents, to ensure that everything flows nicely as you read and study so that you truly understand the subtopics.

You might be wondering how different the current Linux+ exam compares to the previous version.

As mentioned earlier in this Introduction, the main changes are

- An expanded focus on DevOps and cloud topics
- The new Troubleshooting section

Finally, as with most other CompTIA exams, you can expect a handful of scenario questions. In many cases, you will be asked to configure or manage a system using several steps or to describe a collection of Linux features and how they relate to each other. Note this that is not new, but worth mentioning.

In addition, look for the newer simulation questions, which are just like a Linux terminal session, and if you don't already know how to do these, you will by the end of this book!

The rest of the questions will be multiple-choice questions that require you to choose one, choose two, choose three, or choose all that apply.

Who Should Read This Book?

The CompTIA Linux+ certification exam will verify the successful candidate has the knowledge and skills required to configure, manage, operate, and troubleshoot Linux on-premises and cloud-based server environments, while using security best practices, scripting, containerization, and automation.

The level of knowledge and skills expected of the examinee is equivalent to at least 12 months of hands-on experience working with Linux servers in a junior Linux support engineer or junior cloud/DevOps support engineer job role. Additionally, CompTIA does specifically mention that having the experience of passing the A+, Network+, and Server+ exams is considered a recommended prerequisite for taking the Linux+ exam.

This book is for you if you are attempting to attain a position in the IT field or if you want to keep your skills sharp or perhaps retain your job if your company mandates that you take the latest Linux+ exam.

Strategies for Exam Preparation

Strategies for exam preparation will vary depending on your existing skills, knowledge, and equipment available. The ideal exam preparation would consist of building a few virtual machines from scratch and installing and configuring the operating systems covered.

The next best step you can take is to read through the chapters in this book, jotting notes down with key concepts or configurations on a notepad. Each chapter contains a quiz near the end of the chapter that you can use to test your knowledge of the chapter's topics.

Try *all* of the commands you see, look through the configuration files, experiment on your virtual machines, and use the snapshot and rollback feature that is on every virtualization software's toolbar these days—it'll make for a much more pleasant experience when you can try out commands and then revert to a previous snapshot.

After you have read through the book, take a look at the current exam objectives for the CompTIA Linux+ certification exam, listed at <https://www.comptia.org/certifications/linux>. If there are any areas shown in the certification exam outline that you would still like to study, find the appropriate sections in this book and review them.

When you feel confident in your skills, attempt the practice exams included on this book's companion website. As you work through a practice exam, note the areas where you lack confidence and review those concepts or configurations in the book. After you have reviewed the areas, work through the practice exam a second time and rate your skills. Keep in mind that the more you work through the practice exams, the more familiar the questions will become.

After you have worked through each practice exam a second time and feel confident with your skills, schedule the real CompTIA Linux+ (XK0-005) exam through Pearson VUE (<https://home.pearsonvue.com/>). To prevent the information from evaporating out of your mind, you should typically take the exam within a week of when you consider yourself ready to take the exam.

My usual advice for all my certification classes and courses stands: Drink a liter of water and have a nice ripe banana before you go take the exam. The exam is a long one, and you need your brain to function well; the water will help keep that computer in between your ears humming along, and the nutrients (particularly the niacin) in the banana will help you concentrate.

I can't tell you how many pictures of readers I have been sent with their liter of water, a banana, a newly achieved certification, and a BIG SMILE!

In fact, look me up on LinkedIn, at <https://www.linkedin.com/in/rossbrunson/>, and message me with your picture of your water, banana, and certification! I'll be sure to include you in any giveaways of book copies and so forth.

Companion Website

Register this book to get access to the Pearson IT certification test engine and other study materials, as well as additional bonus content. Check this site regularly for new and updated postings written by the author that provide further insight into the more troublesome topics on the exam. Be sure to check the box indicating that you would like to hear from us to receive updates and exclusive discounts on future editions of this product or related products.

To access this companion website, follow these steps:

- Step 1.** Go to www.pearsonitcertification.com/register and log in or create a new account.
- Step 2.** Enter the ISBN: **9780137866885**.
- Step 3.** Answer the challenge question as proof of purchase.
- Step 4.** Click the Access Bonus Content link in the Registered Products section of your account page to be taken to the page where your downloadable content is available.

Please note that many of the companion content files—especially image and video files—are very large.

If you are unable to locate the files for this title by following these steps, please visit www.pearsonITcertification.com/contact and select the Site Problems/Comments option. Our customer service representatives will assist you.

Pearson Test Prep Practice Test Software

As noted previously, this book comes complete with the Pearson Test Prep practice test software, including two full exams. These practice tests are available to you either online or as an offline Windows application. To access the practice exams that were developed with this book, please see the instructions below.

How to Access the Pearson Test Prep (PTP) App

You have two options for installing and using the Pearson Test Prep application: a web app and a desktop app. To use the Pearson Test Prep application, start by finding the registration code that comes with the book. You can find the code in these ways:

- You can get your access code by registering the print ISBN (9780137866885) on pearsonitcertification.com/register. Make sure to use the print book ISBN, regardless of whether you purchased an eBook or the print book. After you register the book, your access code will be populated on your account page

under the Registered Products tab. Instructions for how to redeem the code are available on the book's companion website by clicking the Access Bonus Content link.

- **Premium Edition:** If you purchase the Premium Edition eBook and Practice Test directly from the Pearson IT Certification website, the code will be populated on your account page after purchase. Just log in at pearsonitcertification.com, click Account to see details of your account, and click the digital purchases tab.

NOTE After you register your book, your code can always be found in your account under the Registered Products tab.

Once you have the access code, to find instructions about both the PTP web app and the desktop app, follow these steps:

- Step 1.** Open this book's companion website as shown earlier in this Introduction under the heading, "Companion Website."
- Step 2.** Click the **Practice Exams** button.
- Step 3.** Follow the instructions listed there for both installing the desktop app and using the web app.

Note that if you want to use the web app only at this point, just navigate to pearsonstestprep.com, log in using the same credentials used to register your book or purchase the Premium Edition, and register this book's practice tests using the registration code you just found. The process should take only a couple of minutes.

Customizing Your Exams

When you are in the exam settings screen, you can choose to take exams in one of three modes:

- **Study Mode:** This mode allows you to fully customize your exams and review answers as you are taking the exam. This is typically the mode you use first to assess your knowledge and identify information gaps.
- **Practice Exam Mode:** This mode locks certain customization options in order to present a realistic exam experience. Use this mode when you are preparing to test your exam readiness.
- **Flash Card Mode:** This mode strips out the answers and presents you with only the question stem. This mode is great for late-stage preparation, when you really want to challenge yourself to provide answers without the benefit

of seeing multiple-choice options. This mode does not provide the detailed score reports that the other two modes provide, so it is not the best mode for helping you identify knowledge gaps.

In addition to these three modes, you can select the source of your questions. You can choose to take exams that cover all of the chapters, or you can narrow your selection to just a single chapter or the chapters that make up specific parts in the book. All chapters are selected by default. If you want to narrow your focus to individual chapters, simply deselect all the chapters then select only those on which you wish to focus in the Objectives area.

You can also select the exam banks on which to focus. Each exam bank comes complete with a full exam of questions that cover topics in every chapter. You can have the test engine serve up exams from all four banks or just from one individual bank by selecting the desired banks in the exam bank area.

There are several other customizations you can make to your exam from the exam settings screen, such as the time allowed to take the exam, the number of questions served up, whether to randomize questions and answers, whether to show the number of correct answers for multiple-answer questions, and whether to serve up only specific types of questions. You can also create custom test banks by selecting only questions that you have marked or questions on which you have added notes.

Updating Your Exams

If you are using the online version of the Pearson Test Prep software, you should always have access to the latest version of the software as well as the exam data. If you are using the Windows desktop version, every time you launch the software while connected to the Internet, it checks if there are any updates to your exam data and automatically downloads any changes that were made since the last time you used the software.

Sometimes, due to a number of factors, the exam data might not fully download when you activate your exam. If you find that figures or exhibits are missing, you might need to manually update your exams. To update a particular exam you have already activated and downloaded, simply select the Tools tab, and click the Update Products button. Again, this is only an issue with the desktop Windows application.

If you wish to check for updates to the Windows desktop version of the Pearson Test Prep exam engine software, simply select the Tools tab and click the Update Application button. Doing so enables you to ensure that you are running the latest version of the software engine.

Credits

Figure 4-1-Figure 4-3, Figure 4-5, Figure 5-5: Linus Torvalds

Figure 5-6, Figure 5-7: Wireshark Foundation

Figure 6-1, Figure 7-3, Figure 15-9, Figure 15-10, Figure 18-1-Figure 18-17,
Figure 19-2-Figure 19-6, Figure 20-1, Figure 22-1: Canonical Ltd

Figure 6-2: Debian

Figure 6-3-Figure 6-10, Figure 7-1: SUSE

Figure 7-2: Microsoft

Figure 13-2: The GNOME Project

Figure 13-3-Figure 13-6: KDE

Figure 14-1-Figure 14-4: Red Hat, Inc

Figure 15-7-Figure 15-8: Atlassian

Figure 18-18: Gparted

Cover credit: Quardia/Shutterstock

Using Remote Connectivity for System Management

Used to be that you could just move your chair and work on a server system, way back in the day, but so very quickly that changed to having all the servers in the server room somewhere else, and not wanting to walk there, or even being in the location or even city or country as the servers!

Having grown up in the era of computing when using Telnet (an unsecure remote connectivity option that preceded SSH) and clear-text FTP was commonplace, I've witnessed the evolution of the world of remote networking from a much kinder and gentler place in which hacking was not very common to the current situation in which hacking is a persistent threat and secure terminal access to connect to remote systems safely and reliably to do work, mainly administering them, is an absolute requirement.

Today, you must have SSH installed and configured to be the most secure you can make it, as described in this chapter. You also need to know the importance of using passphrase authentication instead of password authentication to connect to a remote server or group of servers.

Another topic of great importance discussed in this chapter is the concept of privilege elevation or, as the Linux+ exam objectives state, "executing commands as another user." This requires you to have an understanding of several different tools, both for the exam and as a responsible systems administrator.

The elevation of privilege is even more appropriate in conjunction with SSH because typically you should never allow the root user to sign in over SSH. Gaining access to root-restricted resources means you need to elevate yourself to having root access or equivalent when you get *to* the other system.

"Do I Know This Already?" Quiz

The "Do I Know This Already?" quiz enables you to assess whether you should read this entire chapter or simply jump to the "Exam Preparation Tasks" section for review. If you are in doubt, read the entire chapter. Table 11-1 outlines the major headings in this chapter and the corresponding "Do I Know This Already?" quiz questions. You can find the answers in Appendix A, "Answers to the 'Do I Know This Already?' Quizzes and Review Questions."

Table 11-1 “Do I Know This Already?” Foundation Topics Section-to-Question Mapping

Foundation Topics Section	Questions Covered in This Section
SSH (Secure Shell)	1–3
Executing Commands as Another User	4–6

CAUTION The goal of self-assessment is to gauge your mastery of the topics in this chapter. If you do not know the answer to a question or are only partially sure of the answer, you should mark that question as wrong for purposes of the self-assessment. Giving yourself credit for an answer you correctly guess skews your self-assessment results and might provide you with a false sense of security.

1. You want to use a more secure tool than `rpc` to remotely copy data across the network. Which of the following tools would you use?
 - a. `ssh-add`
 - b. `sftp`
 - c. `ssh-agent`
 - d. `scp`
2. You want to disable Secure Shell logins for all users except the root user. Which of the following files would you create to make this happen?
 - a. `/etc/nossh`
 - b. `/etc/nologin`
 - c. `/etc/disablessh`
 - d. `/etc/sshrootonly`
3. The process of allowing remote-running GUI-based applications to display locally when connected to the remote system via SSH is called _____.
 - a. Remote Display
 - b. SSH GUI Mode
 - c. X11 Forwarding
 - d. Tunnel Mode

4. Which command allows you to execute commands as another user, but only if you know the other user's password?
 - a. **runas**
 - b. **pkexec**
 - c. **sudo**
 - d. **su**

5. Which option to the su command allows you to fully take on the user's account settings, including settings that are applied during the login process?
 - a. **-a**
 - b. **-u**
 - c. **-l**
 - d. **-r**

6. Which file is used to configure sudo access?
 - a. **/etc/config/sudo.config**
 - b. **/etc/default/sudoers**
 - c. **/etc/sudo**
 - d. **/etc/sudoers**

Foundation Topics

SSH (Secure Shell)

As mentioned at the beginning of the chapter, the Telnet protocol sends passwords and data in clear text and shouldn't be trusted for important sessions and tasks. The *Secure Shell (SSH)* suite includes a protocol, a daemon, and client utilities that make your host-to-host shell sessions much more secure—about as secure as being at the physical console.

One of the features that makes SSH desirable as a remote protocol is its end-to-end encryption, which encrypts not only the username and password but also all communications.

The SSH suite replaces **telnet**, as well as **rsh**, **rexec**, **rnp**, and other unsecure utilities. You can use SSH to connect for a shell session, or you can use the **scp** command to remotely transfer files through the secure pipe that SSH builds between the hosts.

Key Topic

SSH Components

SSH includes a number of programs and files:

- **ssb**: Used for remote shell sessions on another host, it replaces the **telnet**, **rsh**, and **rexec** commands.
- **scp**: Used for remote copying operations, it replaces the **rnp** command.
- **sshd**: The SSH daemon.
- **ssh-agent**: Runs as a wrapper to the user's session and provides authentication when requested.
- **ssb-add**: Loads the user's key(s) into the agent.

The SSH package configuration files are somewhat scattered. SSH daemon and global configuration files are kept in the **/etc/ssh** directory, and local or user-specific configuration files are kept in the **~/.ssh** directory for each user.

Key Topic

The global configuration files include

- **/etc/ssb/sshd_config**: This is the main configuration for the **sshd** daemon.
- **/etc/ssh/ssh_host_[dr]sa_key**: These files, the **ssh_host_dsa_key** file and the **ssh_host_rsa_key** file, are in the same directory and are the private parts of the host's key structure and should be protected from public view. The

permissions for these files are 600 or rw for the root user and no permissions for anyone else.

- **/etc/ssh/ssh_host_[dr]sa_key.pub:** These files, the **ssh_host_dsa_key.pub** file and the **ssh_host_rsa_key.pub** file, are in the same directory and are the public parts of the host's key structure. These must be world-readable and write-only by the root user or set to 644.
- **/etc/nologin:** This isn't a part of SSH. However, if it's present, no one can log in via SSH except the root user. Non-root users see the contents of the **/etc/nologin** file and then are denied access to the system.

A couple of special file pairs affect how SSH works, particularly the **/etc/ssh/ssh_known_hosts** and **~/.ssh/known_hosts** files. The global file (**/etc/ssh/ssh_known_hosts**) is used to check the public key of a host attempting to attach via SSH. The local file (**~/.ssh/known_hosts**) is the file from which the client gets the public key of the remote server. If a new connection is begun to a previously unknown host, the user sees a message saying that the host is unknown and asking whether the user wants to store the host's key in his known hosts file. If the user answers in the affirmative, the host's public key is added to the **~/.ssh/known_hosts** file.

The **/etc/ssh/ssh_known_hosts** file should be world-readable and root-writable. The **~/.ssh/known_hosts** file must be owned by and writable for the user.

Key Topic

A file of interest, the **~/.ssh/authorized_keys** file, affects only a particular user's environment. This file is used to store the public keys that can be used for logging in as this user. These keys are matched with the keys presented by an **ssh** or **scp** client upon login request.

The SSH client utilities are versatile, with a number of options available to customize the experience. You just need to know the basics for the Linux+ exam, but this section includes a few fun options.

The SSH client command is used to replace the RSH and Telnet programs specifically. Its syntax is as follows:

Key Topic

```
# ssh -l username remotehost
```

If you don't specify a username with the **-l** option, the **ssh** command assumes that you want to use the name of the account with which you are locally logged in. For example, if you are logged in as the user **ross** and you execute the **ssh** command without the **-l** option, the command attempts to log you in as **ross** on the remote system.

For example, I could attach to the host `mp3server` as the user `snuffy` with this command:

```
# ssh -l snuffy mp3server
```

If I have not connected to this server before, I get a message similar to what's shown here:

```
The authenticity of host 'mp3server (192.168.33.44)' can't be
established.
RSA key fingerprint is 73:4f:fa:b0:42:a4:3a:a8:64:2c:ad:26:1
d:b1: 21:e0.
Are you sure you want to continue connecting (yes/no)?
```

If I answer **yes**, the host's public key is added to my `~/.ssh/known_hosts` file and looks something like this:

```
192.168.3.44 ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAIEA1gFIB9VQpFKWAZUzNM+ac/U81Tk9R8OCFFUkegVJXw
j6nqCISPvY2iJwaukcVvaVAQ+JR3EhvOvh4PhoSg4yzBSUkJ8aUBYoRSGj7PCD+vyWyi19
22HGxWbWooMBAO/Was8I7N0zQ6jxD09qNOHcrIFeU7qbOCrKjQDM08Hqjk0=
```

Rather than work with RCP or FTP for file transfer work, I tend to use SCP. The `scp` command uses the SSH protocol and encrypts the files sent from one host to another host. For example, if I wanted to transfer **file1** from my root user's home directory on my machine to the same location on a host named `remotehost`, I could use the following command:

```
# scp /root/file1 root@remotehost:/root/file1
```

The system would prompt me with the RSA key question (as shown in the previous `ssh` example) if I have not connected to this system previously from this account. I would be prompted for the password, and then the system would transfer the files. The output from a file transfer looks like this:

```
# root@192.168.1.73's password:
mypubkey.txt 100% |*****| 1379 00:00
```

You can copy files from your host to another host, as shown previously, or copy files from a remote host to your system by reversing the source and target specifications.

You can even copy files from one remote system to another remote system. For example, the following command recursively copies the `/data` directory and all its contents from the `remote1` host to the `remote2` host after prompting you for the password for both hosts:

```
# scp -r root@remote1:/data root@remote2:/data
```

Another use of the SSH protocol is to log in to a host and use SSH to forward the output from an X client back to your display. This feature, which can be specifically invoked with the `-x` option, is referred to as an *X11 tunnel*.

SSH allows for skipping the password prompt when signing on between computers, which can be convenient if you use the `ssh` or `scp` command frequently and don't mind the possibility that someone could sit down at your accidentally unlocked station and have her way with your network!

NOTE There has been a lot of talk about why it's important to delete `.rhosts` files from user directories. Basically, if you have a user who has a hostname in her `.rhosts` file and that host also has the user's hostname in its `/etc/hosts_equiv` file, that user can log in without a password by using the `rlogin` command. This would be a security risk, so my advice is to delete these files with the following command:

**Key
Topic**

```
# find /home -iname .rhosts -exec rm -f {} \;
```

This deletes all `.rhosts` files it finds in users' home directories and does not prompt you for each deletion.

NOTE The system-wide configuration for the SSH client is kept in the `/etc/ssh/ssh_config` file, while each user's individual configuration file for the SSH client is kept in the user's `~/.ssh/config` file.

The following example shows the steps required to enable SSH use without a password. In this example I have two machines, `fattyre` and `murphy`, both of which are Linux workstations with the necessary SSH software loaded, as per the defaults. This demonstration assumes that `fattyre` and `murphy` are both in each other's `/etc/hosts` files or resolvable via DNS.

**Key
Topic**

Here's how you can enable SSH use without passwords:

Step 1. Log in to `fattyre` as the root user.

Step 2. For this example, create a new user named `user1`:

```
useradd -m user1
```

Step 3. Set `user1`'s password with the `passwd` command to whatever you want:

```
passwd user1
```

Step 4. Switch to the `user1` user account:

```
su - user1
```

Step 5. Create and set the permissions for the `.ssh` directory:

```
mkdir .ssh ; chmod 700 .ssh
```

Step 6. Generate an RSA key by using the `ssh-keygen` command:

```
ssh-keygen -b 1024 -t rsa
```

Step 7. When prompted for the location for the file, press **Enter** to accept the default.

Step 8. When prompted for a passphrase, enter

```
seatec astronomy
```

Step 9. Reenter the passphrase when prompted.

Step 10. Change to the `.ssh` directory and set the permissions on the `id_rsa.pub` file:

```
cd .ssh ; chmod 644 id_rsa.pub
```

Step 11. Copy the `id_rsa.pub` file to a new file called `authorized_keys`:

```
cp id_rsa.pub authorized_keys
```

NOTE The next steps take place on the host `murphy`.

Step 12. From the host `murphy`, ensure that you can contact the host `fattyre` with a ping:

```
ping fattyre
```

Step 13. Sign on to the host `murphy` as the root user.

Step 14. Add a user named `user2`:

```
useradd -m user2
```

Step 15. Set the password for `user2`:

```
passwd user2
```

Step 16. Enter the password twice to confirm it.

Step 17. Switch to the `user2` account:

```
su - user2
```

Step 18. Make a directory and set its permissions with the following command:

```
mkdir .ssh ; chmod 700 .ssh
```

NOTE The next steps take place on the host `fattyre`.

Step 19. From the host **fattyre**, connect to the murphy host as user2:

```
ssh -l user2 murphy
```

Step 20. When prompted about the RSA key, answer **yes** and then enter user2's password.

Step 21. While logged in as user2 on the host murphy via SSH, copy the **authorized_keys** file from the fattyre host with the following command:

```
scp user1@fattyre:~/.ssh/authorized_keys ~/.ssh
```

The output of the **scp** program should look similar to this:

```
authorized_keys 100% |*****| 236 00:00
```

Step 22. Exit user2 on the host murphy and return to being user1 on fattyre.

Step 23. On fattyre as user1, invoke the **ssh-agent** as a wrapper to your shell:

```
ssh-agent $SHELL
```

Step 24. Add your key to the agent:

```
ssh-add
```

Step 25. When prompted for the passphrase, enter the following:

```
no more tears
```

You then see output similar to this:

```
Identity added: /home/ssha/.ssh/id_rsa (/home/ssha/.ssh/id_rsa)
```

Step 26. Try to log in as user2 on murphy and watch what happens:

```
ssh -l user Murphy
```

You shouldn't see any password prompt; you should see only the confirmation of where you last logged in from:

```
Last login: Wed May 26 13:46:55 from fattyre
```

Step 27. If you do see a prompt for the passphrase, enter **no more tears** as you did before.

This is all it takes to get two accounts and machines set up to use SSH utilities without having to enter anything but the **ssh-agent** command along with the passphrase. Remember that **ssh-agent** resides in memory and wraps a security blanket around your shell session, answering any SSH-related security requests for you. The **ssh-add** utility is for adding key information into the agent and doesn't have to be run again as long as your key information remains the same.

NOTE The *ssh-copy-id* command is also a possible choice for sending a user's authorized key to another server. For example, to have the user zakkw's authorized key exist on the Eternal server, you could use the command

```
# ssh-copy -id -i ~/zakkw/.ssh/keyfile zakkw@eternal
```

This will copy the user zakkw's file to the remote server and install it into the **authorized_keys** file, prompting for a password to authenticate the process.

Ideally, this would be performed by the root user for both systems and assumes that you are on a remote system from the Eternal server.

Tunneling

Key Topic

One of the greatest features of SSH is that it can *tunnel*—provide a conduit from inside one network, and even behind a firewall, through to another network. In many cases, using *tunneling* can enable you to do things that either your network administrator doesn't want you to do or you need to do because of an overly restrictive policy, such as at a coffee shop or Internet cafe.

Let's talk about some of the various scenarios where tunneling can come in handy.

X11 Forwarding

X is complex and hard to set up sometimes, and it might seem that tunneling X from another machine to show on yours would be hard too, but *X11 forwarding* is fairly straightforward due to the magical properties of ssh tunneling.

Let's say you have a Linux system named cygnus1 on which you want to run an application in the GUI environment, but you want that application that runs on the remote host to display on your local system.

Here's a possible set of steps you might take:

- Step 1.** On a Mac, download and install XQuartz (<https://www.xquartz.org>).
- Step 2.** Run the command `ssh -X ursulak@cygnus1`.
- Step 3.** After a shell opens in the terminal on the remote host (cygnus1), run the app.
- Step 4.** In a second or two, the remote application, running as a process on the remote host, will display on your system as if it were being run locally.

This is just an example of how forwarding X11 applications from the host they are running on to your local system would work. All sorts of things could go wrong, but

that topic is beyond the scope of the Linux+ exam. The main thing is to understand the concept of X11 forwarding, which we have more than adequately covered.

Port Forwarding

Port forwarding is typically used in scenarios in which there is a need to get around some overly strict or controlling network or firewall. Keep in mind, though, that often there are *very* good reasons for those restrictions and rules being in place, so be responsible and don't willingly cause issues using port forwarding.

Using SSH to forward ports takes several paths, but the main concept is the same: you are using the ssh client on one system to tunnel out to the ssh server on another system and cause services that are represented by a port on the latter system to be mapped, or to appear to be connected, to the other system.

In general, port forwarding occurs in three main ways:

- **Local port forwarding:** This enables you to cause a remote port to be mapped to, and to appear as if it were on, your local system. Kind of like mounting an NFS share locally, mapping a port from a remote system to yours locally effectively makes your local system appear as if it is providing that service.
- **Remote port forwarding:** Flip the scenario around and allow your local system resources to be used by a remote machine. For example, I might map a remote system's port 8080 to my local 5500 port, and anyone connecting to that remote server on the 8080 port will get transported to my local port and service.
- **Dynamic port forwarding:** The term *dynamic port forwarding*, also known as dynamic SOCKS proxying, is a method used to securely tunnel network traffic through a remote server or proxy. Sometimes you don't want to explicitly assign ports and just want the SOCKS proxy on your system to use dynamically assigned local ports and handle all the details. Think of a situation where you need to access ports and protocols that are not allowed through a convention center's network setup. You can use what is effectively a VPN/proxy to drill through the local restrictive network stack and connect to and communicate freely with your desired target hosts, services, and ports.

NOTE The beauty of using SSH tunneling for these purposes is that you don't have to worry that by doing so you are exposing the local network or system unnecessarily; you're using the very secure SSH protocols and stack to do all of this!

Executing Commands as Another User

There are times when you need to execute a command as a different user account. For example, if you log in to the system as a non-root user, but need to execute a command with root privileges.

This section describes methods of running commands as different user accounts, including the **sudo** command, the **su** command and the **pkexec** command.

The sudo Command

The problem with the **su** command is that to provide a user with elevated privileges, you need to provide the user with the root password, which would give that user full administrative access to the system.

Often you want to allow a regular user to execute some commands, but not all commands, as the root user. For example, if a network error occurs on a user's workstation, you might want that user to be allowed to restart the networking service. On some systems, this can be accomplished by executing the following command:

```
# /etc/rc.d/init.d/network restart
```

To execute this command successfully, the user needs to have root privileges. This is where you either give the user the root password (which is not recommended) or you give limited root access the correct and reasonable way, via the **sudo** command and its partner tools.

Instead of providing the user with the root password, you can set up the **sudo** command to allow the user to run just the necessary command. To do this, you need to log in as the root user and then execute the **visudo** command.

```
# visudo
```

This command allows you to edit the **/etc/sudoers** file, the file that allows you to provide root access for specific commands to specific users. The **visudo** command automatically assumes that you want to use the **vi** editor to edit this file. To use a different editor, such as the **nano** editor, execute a command like the following:

```
# export EDITOR=nano
```

Key Topic

NOTE Why use the **visudo** command instead of editing the **/etc/sudoers** file directly? The **visudo** command performs some error checking when you exit the editor to make sure you didn't make formatting mistakes.

The sudoedit Command

One of the conundrums of granting a user access to edit a configuration file is that if you are using **vi/vim**, you are essentially giving the user the ability to run *any* command as root.

To prevent a user from gaining shell access with a simple set of keystrokes from **vi/vim** while running it as root, there exists the **sudoedit** command, which is really just a symbolic link to a function contained in the **sudo** command.

Key Topic

The **sudoedit** command lets you allow a user to use any editor, not just **vi/vim**. It also enables the user to edit the using sudo access, rather than having to log in to the root user account.

When a user edits a file by using the **sudoedit** functionality, a temporary copy of the file(s) is made, and it is owned by the user in question. Since the user is now the owner of the temporary file(s), he can successfully edit the file(s) without having root access. Upon saving the file(s), the temporary copy owned by the user is copied back to the original file location, and the original ownership is restored; the now unnecessary temporary copy is discarded.

To configure **sudoedit**, add the following line to the **/etc/sudoers** file:

```
%newsudo ALL = sudoedit /some/path/to/a/file
```

Configure the newsudo group in **/etc/sudoers** to have the users you want to use **sudoedit**, and then all they need to do is run the command:

```
sudoedit /path/to/that/file
```

The **/etc/sudoers** file has many options. For the Linux+ certification exam, you just need to know how to provide a user with the ability to execute commands as the root user. For example, if you want a user account with the name ross to be able to run all commands as the root user, add the following line:

```
ross ALL=(ALL) ALL
```

To limit a user to a specific command, such as the **/etc/rc.d/init.d/network** command, add the following line:

```
ross ALL=(ALL) /etc/rc.d/init.d/network
```

For a user to execute a command as root, she needs to use the **sudo** command. The syntax is as follows:

```
# sudo /etc/rc.d/init.d/network restart
```

The user is then prompted for her own password (not the root password). If the correct password is given and the access is permitted based on an entry in the **/etc/sudoers** file, the command is executed as the root user. If the user attempts to execute a command that she is not authorized to execute, an error message appears on the screen, and the attempt is logged.

User Privilege Escalation

Users on a Linux system come in the following types, and it is important to know all three types, which type you are logged in, and how to escalate or deescalate your privileges at will by switching from one type to another:

- **Root:** This is the root user, who is the super user and the most privileged user on the system. There should be only one of them, characterized by the name **root** and the UID (user ID) **0**.
- **Standard:** Otherwise known as “regular” or “normal” users, these are the rank-and-file users on the system; they have no special privileges and typically have UIDs that range from 1000 and higher.
- **Service:** These are the accounts that have to exist to ensure that every service or daemon on the system is running as a user, since every process must have a user attached. These accounts are never signed into; they exist in the **/etc/passwd** file and may even have **/bin/nologin** as the specified shell.

The best security practice is to avoid logging in as the root user unless you need to perform specific administration commands. In most cases, you should not log in as the root user directly but rather should gain root access by executing either the **su** command or the **sudo** command.

NOTE The wheel group is an odd thing on the Linux system these days. Traditionally used on Unix systems to allow users to gain root access, the wheel group is often now tied directly to having **sudo** access.

If the wheel group is configured to have privileged access via **sudo** and the **/etc/sudoers** file, then adding a user to the wheel group gives the user those privileges. For example, in our openSUSE system, the wheel group is set up to be able to allow members of that group to execute any command, just as the root would be able to:

```
%wheel  ALL=(ALL) ALL
```

This entry is normally commented out, but it would be very easy to remove the single `#` comment in front of it in the default file to enable the wheel group (and its members) to have full administrative access to the system.

The `su` Command

To gain access to another user account with the `su` command, use the following syntax:

```
su account_name
```



For example, to switch to the root account, execute the following command:

```
# su root
```

This provides you with a non-login shell for the root user. Typically you want a login shell because it provides you with the full user environment (environment variables, shell customizations, and so on). This can be accomplished by using the `-l` option or just a `-` option:

```
# su - root  
# su -l root
```

To gain access to a regular user account, you must provide the account name. However, if you don't provide an account name, the `su` command assumes that you want to switch to the root account. As a result, the following commands are all the same:

- **su - root**
- **su -l root**
- **su -**
- **su -l**

When switching from the root account to a regular user account, no password is required. This means the root user can switch to a regular user account to test that account's features (or troubleshoot problems for the user) without having to change that user's password.

To switch from a regular user account to any other account, you must know the password of the account you are switching to.

NOTE Some distributions' versions of the `su` command allow for the use of `X` and remote `X`; simply use the `suX` command instead of the `su` command. This is most notably present on the openSUSE and SUSE Linux Enterprise distributions.

PolicyKit

PolicyKit, also known as polkit, is a system service in Linux that provides a framework for controlling system-wide privileges and permissions.

PolicyKit exists to provide application-level definition and handling of unprivileged access to privileged processes. For example, you might use PolicyKit to provide a user the ability (and the rights) to perform a task by executing a command with elevated privileges. If you think that sounds like the **sudo** command, it's understandable, because they both have fairly similar goals.

One difference is that PolicyKit is a little easier to use, and certainly less tedious, because you don't have to preface almost everything you do with the **sudo** command.

NOTE The name PolicyKit is used in this book to match the Linux+ exam objectives, but the current package has been renamed Polkit. One of the main positives of PolicyKit is that it's a central place for defining and accessing policies that allow unprivileged users to perform what would normally be privileged actions.

The PolicyKit local configuration is kept in **/etc/polkit-1/localauthority** and uses the common method of include files that contain PolicyKit configuration and end either in **.conf** or, for the more specialized files, **.pkla**.

The following are examples of the types of actions PolicyKit can be configured for:

- Allow the user to configure wireless connections
- Make it possible to mount and unmount USB and other detached media devices
- Let the user manage shutdown, reboot, and hibernate events
- Make devices accessible that are traditionally difficult to access, such as system audio

The pkexec Command

With the previous discussion of the PolicyKit package, **pkexec** makes a lot more sense, as it's the most common way to utilize the PolicyKit rules.

The *pkexec* command, when used to run another command, will execute that command as the targeted user. The user can be specified, but if it is not, **pkexec** attempts to execute the target command as the root user.

For example, to execute the **lemmy.sh** script as the root user, you would type

```
# pkexec lemmy.sh
```



Since a user is not specified, the default for **pkexec** is to attempt to run the subsequent command, script, or executable as the root user.

Summary

This chapter focused on how to remotely and securely connect with systems for the purposes of administering them, using the SSH suite of technologies and the various **ssh**-prefaced commands you learned about in this chapter.

You also learned about the methods for privilege elevation, or running commands or acting as another user, such as **su**, **sudo**, and **pkexec**.

Exam Preparation Tasks

As mentioned in the section “Goals and Methods” in the Introduction, you have a couple of choices for exam preparation: the exercises here, Chapter 23, “Final Preparation,” and the exam simulation questions in the Pearson Test Prep Software Online.

Review All Key Topics

Review the most important topics in this chapter, noted with the Key Topic icon in the left margin of the page. Table 11-2 lists these key topics and the page number on which each is found.



Table 11-2 Key Topics for Chapter 11

Key Topic Element	Description	Page Number
List	Programs and files that SSH includes	408
List	SSH global configuration files	408
Paragraph	Description of the <code>~/.ssh/authorized_keys</code> file	409
Paragraph	Example syntax for connecting to a remote system via the <code>ssh</code> command	409
Note	Example of deleting all the <code>.rhosts</code> files on a given system	411
Step list	Enabling SSH use without passwords	411
Section	Tunneling	414
Note	Reason to use the <code>visudo</code> command instead of editing <code>/etc/sudoers</code> directly	416
Paragraph	Using the <code>sudedit</code> command to allow a user to use any editor	417
Paragraph	Switching to the root account using the <code>su</code> command	419
Paragraph	Example of using the <code>pkexec</code> command to run a script as an different user	420

Define Key Terms

Define the following key terms from this chapter and check your answers in the glossary:

ssh, **ssh-add**, **/etc/ssh/sshd_config**, **known_hosts**, **ssh_config**, **ssh-keygen**, **ssh-copy-id**, tunneling, X11 forwarding, port forwarding, dynamic forwarding, privilege escalation, **su**, **sudo**, **visudo**, **/etc/sudoers**, **sudoedit**, PolicyKit, **pkexec**

Review Questions

The answers to these review questions are in Appendix A.

1. After configuring the PolicyKit rules for your system, what command would you use to use those rules when executing a target command that your current user doesn't have rights to execute alone?
 - a. **sudo**
 - b. **pkexec**
 - c. **suexec**
 - d. **execit**
2. When configuring the **sudo** command, what is the full path and filename of its primary configuration file?

3. You are able to access a remote system using just a passphrase for authentication. What must you have copied from your system to the remote system in order for this to happen?
 - a. Your personal public key
 - b. The system's public key
 - c. The wheel group's public key
 - d. The remote user's private key
4. When configuring your system to allow or deny certain groups or users to sign in via SSH, what is the full path and filename of the configuration file where these settings are kept?

5. If you invoke the **ssh** command with the **-X** option, what are you likely to be doing after you sign on to the remote system?
 - a. Just standard commands
 - b. Running xeyes locally and displaying remotely
 - c. Running X11 on the remote system and displaying locally
 - d. Running remote X apps that display locally

6. Which of the following commands is specifically designed to make it more secure to edit files when using **sudo** to elevate your privileges?
 - a. **sudo vim**
 - b. **visudo**
 - c. **sudoedit**
 - d. **nanobot**

This page intentionally left blank

Index

Symbols

- && (double ampersands), shell scripting, 483–484
- / (filesystems)
 - block storage, 30
 - checking, 126–127
 - CIFS, 133
 - FHS
 - overview, 6
 - /root directory, 6–7
 - /usr directory, 7–8
 - FUSE, 31
 - applications/uses of, 32
 - purpose for using, 31
 - user requests, 32
 - managing
 - Btrfs tools, 128–130
 - EXT2/3/4 tools, 127
 - fsck tool, 126–127
 - tune2fs command, 128
 - XFS commands, 130–131
 - manually mounting, 121
 - /proc, 36
 - dmidecode command, 37–38
 - ls* commands, 36–37
 - /root directory, 6–7
 - troubleshooting
 - corruption, 638–642
 - mismatches, 642–643
 - unmounting, 121–122
 - /usr directory, 7–8
- . (periods) in filenames, 46–47
- | (pipes), shell scripting, 481–483
- ; (semicolons), shell scripting, 483

Numbers

- 1 command, 21
- 2 command, 21

- 3 command, 21
- 4 command, 21
- 5 command, 21

A

- aa-complain command, 457
- aa-disable command, 456
- aa-unconfined command, 457
- absolute pathnames, 47, 48, 50, 56, 70
- access
 - ACL, 445–446
 - masks, 447–448
 - setting, 446–447
 - troubleshooting, 733
 - viewing, 446
 - AppArmor, 456
 - aa-complain command, 457
 - aa-disable command, 456
 - aa-unconfined command, 457
 - directories, 457
 - command-line utilities, 457
 - context-based access, 448–456
 - file access, troubleshooting, 731–732
 - ACL, 733
 - attribute issues, 733–734
 - blocks, 737–738
 - context issues, 732
 - group access, 732
 - non-policy issue, 734–735
 - passwords, 735–736
 - permissions, 732–733
 - policy issues, 734–735
 - privilege elevation, 736
 - quotas, 736–738
 - file attributes, 442
 - displaying, 442–443
 - key attributes, 443

- removing, 443–444
 - setting, 443
- group file access issues, troubleshooting, 732
- LDAP, 327, 329
- Pearson Test Prep practice exams, 769–770
- permissions, 429
 - chmod command, 432–434
 - finding files by permissions, 444–445
 - granularity issues, 437–438
 - ownership, 434–436
 - SGID permissions, 438–439, 440–441
 - special bit permissions, 438–439
 - sticky bit permissions, 438–439
 - SUID permissions, 438–440
 - trio bits, 429–432
 - troubleshooting, 732–733
- restricting cron job access, 177
- user access, troubleshooting
 - file access issues, 731–738
 - login issues, 728–731
- accounts
 - group accounts
 - adding users to, 364
 - etc/group files, 358–361
 - GID, 358–360
 - group passwords, 345, 360
 - modifying users in, 364–365
 - primary groups, 358
 - removing, 367
 - secondary groups, 358
 - UPG, 359
 - user accounts, 355–358
 - adding users, 361–363
 - etc/passwd files, 355–361
 - modifying users in, 364–365
 - removing, 366–367
- ACL (Access Control Lists), 445–446
 - masks, 447–448
 - setting, 446–447
 - troubleshooting, 733
 - viewing, 446
- ad hoc cron jobs, running, 178–180
- adding
 - repositories, 238–239, 266
 - users to
 - accounts, 361–363
 - group accounts, 364
- advanced directory navigation, 48–49
- aging passwords, 344–345
- ambassador containers, 603
- ampersands (&&), shell scripting, 483–484
- analyzing/troubleshooting
 - bandwidth, 686–687
 - capacity issues
 - causes/symptoms, 633–634
 - diagnosing, 634–638
 - fixing, 634–638
 - collisions, 680–683
 - dropped packets, 680–683
 - errors= option, 661–663
 - filesystems
 - corruption, 638–642
 - mismatches, 642–643
 - firewalls
 - causes/symptoms, 675–676
 - diagnosing issues, 676–678
 - fixing issues, 676–679
 - high latency issues, 670
 - causes/symptoms, 623
 - diagnosing, 624–626
 - fixing, 626–627
 - overview, 623
 - interfaces
 - collisions, 680–683
 - dropped packets, 680–683
 - I/O schedulers, 645–647
 - IOPS, 631
 - irrelevancy of, 632–633
 - overview, 632–633
 - link status
 - data link layer, 685
 - physical layer, 684–685
 - low throughput
 - causes/symptoms, 627–628
 - diagnosing, 628–631
 - fixing, 628–631
 - overview, 627
- LVM
 - causes/symptoms, 657
 - diagnosing, 657–659
 - fixing, 657–659
 - overview, 656–657
- mount options
 - causes/symptoms, 659
 - diagnosing, 660–663
 - errors= option, 661–663
 - fixing, 660–663

- overview, 659
 - UUID, 660–661
 - name resolution, 687–689
 - networks
 - bandwidth, 686–687
 - configuration issues, 670–674
 - firewalls, 675–679
 - interfaces, 679–683
 - link status, 684–685
 - remote system tests, 689–696
 - NVMe
 - causes/symptoms, 647–648
 - diagnosing, 648–649
 - fixing, 648–649
 - overview, 647
 - RAID, 652
 - array health, 655–656
 - causes/symptoms, 653
 - device health, 654
 - diagnosing, 653–656
 - fixing, 653–656
 - monitoring, 654–656
 - SSD
 - causes/symptoms, 649–650
 - diagnosing, 650
 - fixing, 650
 - overview, 649
 - SSD TRIM, 651–652
 - storage issues
 - capacity issues, 633–638
 - filesystems, 638–643
 - high latency, 623–627
 - I/O schedulers, 643–647
 - IOPS, 631–633
 - low throughput, 627–631
 - LVM, 656–659
 - mount options, 659–662
 - NVMe, 647–649
 - RAID, 652–656
 - SSD, 649–652
 - UUID, 660–661
 - Ansible, 581–583
 - apm command, 21
 - AppArmor, 456
 - aa-complain command, 457
 - aa-disable command, 456
 - aa-unconfined command, 457
 - directories, 457
 - AppImage, 269
 - apps (applications)
 - crashes, 756–758
 - sandboxed applications, 268, 269
 - AppImage, 269
 - defined, 268
 - Flatpak, 269
 - Snappd, 270
 - troubleshooting, 756–758
 - APT
 - configuration files, 288
 - repository files, 289
 - archiving files, 63–64
 - listing, 71–72
 - tar command, 64–66, 71
 - arguments (shell scripting), accepting, 499–500
 - arp command, 194, 198–199
 - at command, 178–179
 - attaching to containers, 530
 - attributes (files), 442
 - displaying, 442–443
 - key attributes, 443
 - removing, 443–444
 - setting, 443
 - troubleshooting, 733–734
 - audit2allow command, 455–456
 - authentication, 326
 - biometrics, 327
 - certificates, 323
 - LDAP, 327, 329
 - MFA, 326–327
 - OTP, 327
 - PAM, 327–329
 - configuring, 330–331
 - LDAP integration, 329
 - SSDD, 331–332
 - user lockouts, 329–330
 - SSO, 332–333
 - awk command, 92–94, 500–501
- ## B
- backups, 76
 - clones, 77
 - differential backups, 76–77
 - full backups, 76
 - images, 77
 - incremental backups, 76
 - snapshots, 77
 - bandwidth

- analyzers, 686
- latency, 686
- troubleshooting, 686–687
- .bash_profile scripts, 371
- basic directory navigation, 47–48
- batch command, 179–180
- Before/After directives, 747–748
- bfq I/O schedulers, 645
- bg command, 149, 156
- bind mounts, 606–607
- biometric authentication, 327
- BIOS boot process, 14
- blkid command, 106–108
- block devices, 23
- blocks
 - file access issues, 737–738
 - memory, 718
 - processes, 146
 - storage, 30
- bodily preparations, final exam, 768
- booleans, SELinux, 452–454
- boot process, 9
 - basics, 8
 - BIOS, 14
 - boot loaders and files, 14
 - common commands, 21–22
 - defined, 8
 - dracut framework, 10
 - GRUB, 14–15
 - GRUB2
 - changes made, 15–16
 - command line, 18–19
 - command names, 16–17
 - configuring, 20
 - installing, 17–18
 - kernel images, 19
 - initramfs method, 10
 - initrd method, 9–10
 - ISO files, 13–14
 - MBR, 14
 - NFS, 12–13
 - PXE, 11–12
 - stages of, 8
 - systemd command, 164–165
 - UEFI, 10–11
- bootstrapping, 612–613
- bounce signals, 149
- branches, version control, 556–558, 561–563
- bridging networks, 609–610

- Btrfs tools, 128–130
- budgeting time, final exam, 767
- buffers
 - memory, 718
 - processes, 146
- build command, 533
- Buildah, 533–534
- bunzip2 command, 75–76
- bzip2 command, 75

C

- caches
 - memory, 718
 - processes, 146
- Calico, 609
- capacity issues
 - causes/symptoms, 633–634
 - diagnosing, 634–638
 - fixing, 634–638
- case sensitivity, user login issues, 730
- case statements, 493–495
- cat command, 655
- cd command, 47–48
- CD (Continuous Delivery), 589
- CD (Continuous Deployment), 589
- certificates, PKI
 - authentication, 323
 - authorities, 323–324
 - HTTPS, 325
 - purpose of, 323
 - self-signed certificates, 323
 - SSL, 325–326
 - use cases, 325–326
- cfq I/O schedulers, 644
- change persistent I/O schedulers, 646–647
- changing
 - default firewall policies, 393–394
 - file ownership, 435–436
 - group ownership, 436
 - passwords, 370–374
 - text, vim, 88
- chapter-ending review tools, final exam, 772
- character devices, 23
- chattr command, 442, 443
- check ins/outs, version control, 542–544
- checking filesystems, 126–127
- Chef, 583
- chgrp command, 436
- chmod command, 432

- octal mode, 432–433
 - SUID permissions, 438–439
 - symbolic mode, 433–434
- choosing kernel update methods, 271
- chown command, 435–436
- chrony command, 302–303
- CI/CD (Continuous Integration/Continuous Deployment), 588–590
- CIFS (Common Internet File System), 133
- Cillium, 609
- clones, 77, 548
- cloud-init, 613
- collisions, troubleshooting, 680–683
- columns, cutting, 502–503
- Command mode, vim, 83–84, 86–87, 89
- command-line utilities, 457
- commands
 - executing as another user, 416
 - su command, 419
 - sudo command, 416
 - sudoedit command, 417–418
 - shell scripting
 - multiple command operators, 483–484
 - substituting commands, 484–485
 - using output of another command, 487–488
- committing files, 543, 544–545
- comparing versions, 560
- compressed files, package management, 228
- compressing files, 66–70, 75–76
- conditionals, shell scripting, 488–489
- configuration changes, IaC, 579–580
- configuring
 - APT configuration files, 288
 - crontabs, 170–173
 - nicknames, 175
 - output, 174–175
 - PATH issues, 173–174
 - dates/time zones, 303–304
 - DNF configuration files, 289
 - Git, 549–552
 - GRUB2, 20
 - ifcfg-* files, 195–196
 - kernel, 289
 - modules, 293–300
 - setting kernel parameters, 291–293
 - viewing kernel parameters, 290–291
 - name resolution, 200–202
 - networks, troubleshooting, 670–674
 - network-scripts, 195–196
 - NTP, 301–303
 - PAM, 330–331
 - repositories
 - APT repository files, 289
 - configuration files, 287–289
 - YUM repository files, 287–288
 - RPM packages, 284–287
 - services, 300–301
 - NTP, 301–303
 - SSH, 301
 - software
 - dates/time zones, 303–304
 - kernel options, 289–300
 - repository configuration files, 287–289
 - repository files, 289
 - RPM packages, 284–287
 - services, 300–304
 - updating configuration files, 283–284
 - SSH, 301
 - time-zones, troubleshooting, 759–760
 - YUM
 - configuration files, 288
 - packages, 259–262
- connectivity, remote
 - pkexec command, 420–421
 - PolicyKit, 420
 - SSH, 408
 - components of, 408–414
 - executing commands as another user, 416–418, 419
 - global configuration files, 408–409
 - package configuration files, 408
 - port forwarding, 415
 - SSH client command, 409–410
 - ssh-agent command, 413
 - ssh-copy-id command, 414
 - tunneling, 414–415
 - user privilege escalation, 418–419
 - using without passwords, 411–413
 - X11 forwarding, 414–415
 - X11 tunnels, 411
- containers, 525
 - attaching to, 530
 - Buildah, 533–534
 - detaching from, 530
 - ending executions, 530
 - exiting executions, 530
 - images

- build command, 533
- finding, 526–527
- inspecting, 529
- list command, 534
- operations, 533–534
- pull command, 534
- pulling, 527–528
- push command, 533–534
- rmi command, 534
- running as containers, 529
- viewing, 528
- Kubernetes (K8), 602
 - ambassador containers, 603
 - CRI, 602
 - CRI-O, 602
 - sidecars, 603
 - specialized types, 603
- logs, viewing, 531–532
- naming, 529
- networks, 608
 - bridging networks, 609–610
 - Docker Swarms, 608–609
 - host networks, 610–611
 - overlay networks, 608, 609
 - swotting NAT, 610
- persistent storage, 605, 607
 - bind mounts, 606–607
 - Docker volumes, 605–606
 - PV subsystems, 607–608
 - PVC subsystems, 608
- Podman container tool, 526
- ports
 - exposing, 532–533
 - mapping, 532–533
- registries, 613–614
- removing, 531
- single-node multicontainers, use cases, 604–605
- Skopeo container tool, 526
- three-container node example, 605
- tools
 - installing, 525–526
 - verifying, 526
- two-container node examples, 604–605
- context issues, troubleshooting, 732
- context-based access, SELinux, 448–449
 - audit2allow command, 455–456
 - booleans, 452–454
 - contexts, 454–455
 - modes, 450–451
 - policies, 451–452
- control plane, Kubernetes (K8), 601–602
- convenience crontabs, 176–177
- copying
 - directories, 51–53
 - files, 51–53
 - objects between systems
 - nc command, 81–82
 - rsync command, 79–80
 - scp command, 78–79
- corrupted filesystems, troubleshooting, 638–642
- costs, final exam, 766
- cp command, 48–49, 51–53
- cpio command, 72–73
- CPU
 - process priorities, 713
 - times, 711
 - idle time, 713
 - iowait time, 713
 - measuring, 711–712
 - steal time, 712
 - system time, 712–713
 - user time, 712
 - troubleshooting
 - high CPU utilization, 707
 - high load average, 707–708
 - high run queues, 708–711
 - process priorities, 713
 - runaway processes, 704–705
 - zombie processes, 705–706
 - viewing hardware information, 719–720
- crashes, apps (applications), 756–758
- creating
 - directories, 56–57
 - firewall rules, 385–397
 - LVM, 117
 - software RAID, 114
 - variables, shell scripting, 470–471
- CRI (Container Runtime Interface), 602
- CRI-O (CRI-Open), 602
- cron jobs
 - ad hoc cron jobs, 178–180
 - restricting access, 177
- crontabs
 - configuring, 170–173
 - nicknames, 175
 - output, 174–175
 - PATH issues, 173–174
 - convenience crontabs, 176–177

- syntax checks, 176
- system crontabs, 176
- viewing, 175–176
- curl command, 218–219
- customizing, Pearson Test Prep practice exams, 770–771
- cut command, 502–503

D

- daemons, systemd command, 160
- DAG (Directed Acrylic Graphic), 545
- data flow (proper), 211
 - netstat command, 211–213
 - tcpdump command, 216–217
 - Wireshark, 214–216
- data link layer (links), troubleshooting, 685
- dates/time zones, configuring, 303–304
- dd command, 73–74
- deadline I/O schedules, 644
- Debian
 - interface management, 196–197
 - packages, 228, 229
 - dependency issues, 232–233
 - dpkg command, 231–235
 - installing, 231
 - local packages, 230–231
 - querying, 233–234
 - reconfiguring, 234–235
 - removing, 231–232
- deleting
 - lines, vim, 88–89
 - .rhosts files, 411
 - text, vim, 88–89
- DenyHosts utility, 400
- dependency issues, Debian packages, 232–233
- detaching from containers, 530
- /dev directory
 - device types, 23–24
 - storage, 28
- /dev/null files, 24
- /dev/tty, 480–481
- dev/urandom files, 24
- /dev/zero files, 24
- df command, 125–126
 - capacity issues, 634–636
 - low throughput, 628–629
- difference execution, version control, 558–563
- differential backups, 76–77
- dig command, 205–206, 687–688

- digital signatures, 325
- directories
 - AppArmor, 457
 - copying, 51–53
 - creating, 56–57
 - /dev
 - device types, 23–24
 - storage, 28
 - .git directory, 555
 - LDAP, 327, 329
 - links
 - hard links, 62–63
 - soft links, 60–61
 - symbolic links, 61
 - listing, 49–50
 - moving objects, 54–56
 - navigating
 - advanced navigation, 48–49
 - basic navigation, 47–48
 - ownership, 434–436
 - removing, 56–57
 - removing objects, 57
 - /root, overview, 6–7
 - SGID permissions, 441
 - /usr, 7–8
- disabled mode, SELinux, 450
- disabling
 - insecure services, 342
 - services, 166–167
- disk partitioning, 108
 - fdisk command, 108–112
 - MBR partition tables, 112
 - parted command, 112–113
 - partprobe command, 113–114
- disk usage/storage, monitoring
 - df command, 125–126
 - du command, 124–125
 - iostat command, 123–124
- displaying
 - DMI table information, 37–38
 - file attributes, 442–443
 - file metadata, 58–60
 - memory, 145
 - socket information, 191–192
 - variables, shell scripting, 469–471
- DITKA questions, final exam, 772
- DMI table information, displaying, 37–38
- dmidecode command, 37–38
- DNAT (Destination NAT), 397

DNF (Dandified YUM), 262
 DNF configuration files, 289
 Docker Swarms, 608–609
 Docker volumes, 605–606
 domains, final exam, 766–767
 dontaudit policy, 734–735
 downloading software
 curl command, 218–219
 wget command, 218–219
 dpkg command, 231–235
 dracut framework, boot process, 10
 dropped packets, troubleshooting, 680–683
 du command, 124–125, 634, 636–637
 DVCS (Distributed Version Control Systems),
 543–546

E

e2fsck command, mismatched filesystems,
 642–643
 earplugs, 767
 editing files
 awk command, 92–94
 nano, nano, 90–92
 printf command, 94
 vim, 82–90
 egrep command, 510–511
 elevated privileges, troubleshooting, 736
 enabling services, 166–167
 encryption
 LUKS, 123
 PKI, 324–325
 ending container executions, 530
 enforcing mode, SELinux, 450
 environment variables
 shell scripting, 469
 systemd command, 162
 errors (shell scripting)
 finding on demand, 476–477
 redirecting, 479–480
 returning error codes, 499
 errors= option, troubleshooting, 661–663
 etc/group files, 358–361
 etc/passwd files, 355–361
 etc/profile files, 371
 etc/shadow files, 356, 357, 365, 368–370
 etc/skel files, 361, 362–363
 ethtool command, troubleshooting
 interfaces, 681
 links, 684–685

exam preparation
 chapter-ending review tools, 772
 domains, 766–767
 earplugs, 767
 fees, 766
 getting rest, 768
 ID code, 766
 languages, 765
 locking up valuables, 768
 Pearson Test Prep practice exams, 768
 access, 769–770
 customizing, 770–771
 Premium Edition, 771–772
 updating, 771–772
 physical preparations, 768
 practice exams, 768–772
 questions
 budgeting time, 767
 DITKA questions, 772
 number of, 765
 types of, 765
 required passing score, 765
 study trackers, 767
 suggested review/study plans, 772
 taking notes, 768
 time limit, 765
 tips for preparing, 767–768
 travel time, 768
 ExecStart, 747
 ExecStop, 747
 exiting container executions, 530
 expanding variables, shell scripting, 472–473
 exposing ports, 532–533
 EXT2/3/4 tools, 127
 extensions, file, 47

F

fail2ban service, 398–400
 faillog command, 735
 fallback locales, 306
 fcstat command, 108
 fdisk command, 108–112
 fees, final exam, 766
 fg command, 156
 fgrep command, 510–511
 FHS (Filesystem Hierarchy Standard)
 overview, 6
 /root directory, 6–7
 /usr directory, 7–8

- file command, 57–58
- files
 - access, troubleshooting, 731–732
 - ACL, 733
 - attribute issues, 733–734
 - blocks, 737–738
 - context issues, 732
 - group access, 732
 - non-policy issue, 734–735
 - passwords, 735–736
 - permissions, 732–733
 - policy issues, 734–735
 - privilege elevation, 736
 - quotas, 736–738
 - APT
 - configuration files, 288
 - repository files, 289
 - archiving, 63–64
 - listing, 71–72
 - tar command, 64–66, 71
 - attributes, 442
 - displaying, 442–443
 - key attributes, 443
 - removing, 443–444
 - setting, 443
 - backups, 76
 - clones, 77
 - differential backups, 76–77
 - full backups, 76
 - images, 77
 - incremental backups, 76
 - snapshots, 77
 - clones, 77
 - committing, 543, 544–545
 - compressed files, package management, 228
 - compressing, 66–70, 75–76
 - copying, 51–53
 - objects between systems, nc command, 81–82
 - objects between systems, rsync command, 79–80
 - objects between systems, scp command, 78–79
 - descriptors, shell scripting, 475
 - DNF configuration files, 289
 - editing
 - awk command, 92–94
 - nano, nano, 90–92
 - printf command, 94
 - vim, 82–90
 - etc/group files, 358–361
 - etc/passwd files, 355–361
 - etc/profile files, 371
 - etc/shadow files, 356, 357, 365, 368–370
 - etc/skel files, 361, 362–363
 - extensions, 47
 - finding, 515–517
 - finding by permissions, 444–445
 - formatting, printf command, 94
 - global configuration files, SSH, 408–409
 - hidden files, 46–47
 - ifcfg-* files, configuring, 195–196
 - ignoring, Git, 555–556
 - images, 77
 - JSON format, 577–579
 - links
 - hard links, 62–63
 - soft links, 60–61
 - symbolic links, 61
 - listing, 49–50
 - archived files, 71–72
 - cpio command, 72–73
 - dd command, 73–74
 - tar command, 71–72
 - locking, 541
 - merging, 562–568
 - metadata, displaying, 58–60
 - moving objects, 54–56
 - naming
 - periods (.) in filenames, 46–47
 - spaces in filenames, 47
 - navigating, vim, 85–86
 - ownership, 434–436
 - package configuration files, SSH, 408
 - permissions, 429
 - chmod command, 432–434
 - finding files by permissions, 444–445
 - granularity issues, 437–438
 - ownership, 434–436
 - SGID permissions, 438–439, 440–441
 - special bit permissions, 438–439
 - sticky bit permissions, 438–439
 - SUID permissions, 438–440
 - trio bits, 429–432
 - troubleshooting, 732–733
 - removing objects, 57

- repository configuration files, 287–289
- .rhosts files, deleting, 411
- RPM package files, 244–245
- saving, vim, 87
- snapshots, 77
- State files, Salt, 585–586
- storage, 30
- sysctl files, 291–293
- testing, 490–491
- touching, 50–51
- translating, 502
- troubleshooting, 733–734
- types of, 57–58
- YAML format, 576–577
- YUM
 - configuration files, 288
 - repository files, 289
- filesystem table, 118–121
- filesystems (/)
 - block storage, 30
 - checking, 126–127
 - CIFS, 133
 - FHS
 - overview, 6
 - /root directory, 6–7
 - /usr directory, 7–8
 - FUSE, 31
 - applications/uses of, 32
 - purpose for using, 31
 - user requests, 32
 - managing
 - BtrFS tools, 128–130
 - EXT2/3/4 tools, 127
 - fsck tool, 126–127
 - tune2fs command, 128
 - XFS commands, 130–131
 - manually mounting, 121
 - /proc, 36
 - dmidecode command, 37–38
 - ls* commands, 36–37
 - /root directory, 6–7
 - troubleshooting
 - corruption, 638–642
 - mismatches, 642–643
 - unmounting, 121–122
 - /usr directory, 7–8
- filtering packets
 - by destination, 392–393
 - filtering points, 385–388
- important terms, 388–389
- incoming packets, 389–391
 - by multiple criteria, 392
 - outgoing packets, 395–396
 - overview, 385–388
 - by protocol, 391–392
- final exams, preparing for
 - chapter-ending review tools, 772
 - domains, 766–767
 - earplugs, 767
 - fees, 766
 - getting rest, 768
 - ID code, 766
 - languages, 765
 - locking up valuables, 768
 - Pearson Test Prep practice exams, 768
 - access, 769–770
 - customizing, 770–771
 - Premium Edition, 771–772
 - updating, 771–772
 - physical preparations, 768
 - practice exams, 768–772
 - questions
 - budgeting time, 767
 - DITKA questions, 772
 - number of, 765
 - types of, 765
 - required passing score, 765
 - study trackers, 767
 - suggested review/study plans, 772
 - taking notes, 768
 - time limit, 765
 - tips for preparing, 767–768
 - travel time, 768
- find command, 48, 634, 637–638
- finding
 - container images, 526–527
 - errors on demand, shell scripting, 476–477
 - files, 515–517
 - files by permissions, 444–445
- firewalls, 382–383
 - changing default firewall policies, 393–394
 - defined, 382
 - DenyHosts utility, 400
 - fail2ban service, 398–400
 - firewalld command, 383–384
 - IPset utility, 400
 - iptables, 382, 383, 385–397
 - NAT, 397–398

- nftables, 383–384
 - refreshing, 674–675
 - rules
 - creating, 385–397
 - logging, 396–397
 - saving, 394–395
 - stateful firewall rules, 396
 - stateless firewall rules, 396
 - runtime firewalls, 384
 - services, verifying operation, 678–679
 - troubleshooting
 - causes/symptoms, 675–676
 - diagnosing issues, 676–678
 - fixing issues, 676–679
 - ufw, 384–385
 - first-generation version control, 541–542
 - Flannel, 609
 - Flatpak, 269
 - for loops, 496–497
 - force multipliers, vim, 86, 88
 - formatting files, printf command, 94
 - free command, 145, 147–148, 717
 - freshening, RPM packages, 249–250
 - fsck command
 - checking filesystems, 126–127
 - corrupted filesystems, 639–642
 - full backups, 76
 - functional tools, IaC, 581
 - FUSE (Filesystem in Userspace), 31
 - applications/uses of, 32
 - purpose for using, 31
 - user requests, 32
- ## G
- getent command, 729
 - getfacl command, 446–448
 - getting rest, final exam, 768
 - GID (Group ID), 358–360
 - Git, 590
 - branches, 556–558, 561–563
 - clones, 548
 - comparing, versions, 560
 - configuring, 549–552
 - difference execution, 558–563
 - .git directory, 555
 - ignoring files, 555–556
 - installing, 546–547
 - merge command, 590
 - PR, 590
 - rebase command, 590
 - repository hosts, 549
 - stages, 548–549
 - tags, 552–555
 - version control
 - DVCS, 543–546
 - first generation, 541–542
 - merges, 542–543, 544–546
 - second generation, 542–543, 544–545
 - third generation, 543–546
 - vimdiff tool, 567
 - whitespace, version control, 560–561
 - global configuration files, SSH, 408–409
 - globbing, 467–468
 - GParted tool, 655
 - GPT (GUID Partition Tables), 29
 - granularity issues, permissions, 437–438
 - graphical package managers, 242
 - grep command, 498–499, 500–501, 505–514
 - group accounts
 - adding users to, 364
 - etc/group files, 358–361
 - GID, 358–360
 - group passwords, 345, 360
 - modifying users in, 364–365
 - primary groups, 358
 - removing, 367
 - secondary groups, 358
 - UPG, 359
 - group file access issues, troubleshooting, 732
 - group ownership, changing, 436
 - group passwords, 345
 - groupadd command, 364
 - groupdel command, 367
 - groupmod command, 364–365
 - groups command, 732
 - GRUB (Grand Unified Bootloader), 14–15
 - GRUB2
 - changes made, 15–16
 - command line, 18–19
 - command names, 16–17
 - configuring, 20
 - kernel images, 19
 - gunzip command, 75
 - gzip command, 75
- ## H
- hard links, 62–63
 - hardening Linux, 333

- insecure services, disabling/removing, 342
 - kernel security, service accounts, 347
 - passwords
 - aging, 344–345
 - group passwords, 345
 - setting parameters, 343
 - strength of, 333–338
 - umask, 340–342
 - unused packages, removing, 345–347
 - vulnerability scans
 - nc command, 338–340
 - nmap command, 333–338
 - hardware
 - info, listing, 35
 - tokens, 326
 - viewing
 - CPU information, 719–720
 - memory information, 720
 - hashing, PKI, 324–325
 - hddtemp command, 648
 - heredocs (here documents), 477
 - hidden files, 46–47
 - high CPU utilization, troubleshooting, 707
 - high latency
 - causes/symptoms, 623, 670
 - diagnosing, 624–626
 - fixing, 626–627
 - overview, 623
 - high load average, troubleshooting, 707–708
 - high run queues, troubleshooting, 708–711
 - host command, 204–205
 - host networks, 610–611
 - hostname command, 194, 197–198
 - hostnames on systemd systems, 204
 - hping3 command, troubleshooting, bandwidth
 - issues, 686
 - htop command, 144–145
 - HTTPS, PKI certificates, 325
- I**
- IaC (Infrastructure as Code), 580
 - Ansible, 581–583
 - CD, 589
 - Chef, 583
 - CI/CD, 588–590
 - configuration changes, 579–580
 - defined, 579
 - functional tools, 581
 - Git, 590
 - merge command, 590
 - PR, 590
 - rebase command, 590
 - JSON, 577–579
 - procedural tools, 581
 - Puppet, 583
 - SaltStack, 584–586
 - source control, 580
 - Terraform, 586–588
 - YAML, 576–577
 - ID code, final exam, 766
 - id command, 372, 728–729
 - identity management
 - GID, 358–360
 - UID, 355–358
 - user identity query options, 372–374
 - idle time, 713
 - ifcfg command, 194
 - ifcfg-* files, configuring, 195–196
 - ifconfig command, 194, 195
 - ignoring files, version control, 555–556
 - images
 - backups, 77
 - container images
 - build command, 533
 - finding, 526–527
 - inspecting, 529
 - list command, 534
 - operations, 533–534
 - pull command, 534
 - pulling, 527–528
 - push command, 533–534
 - rmi command, 534
 - running as containers, 529
 - viewing, 528
 - incremental backups, 76
 - init command, 21
 - initramfs method, boot process, 10
 - initrd method, boot process, 9–10
 - inodes, 51
 - input/output streams, shell scripting, 475
 - insecure services, disabling/removing, 342
 - Insert mode, vim, 83–84
 - inspecting
 - container images, 529
 - software RAID, 114
 - installing
 - container tools, 525–526
 - Debian packages, 231

- Git, 546–547
 - GRUB2, 17–18
 - packages
 - remote packages, 239–241
 - system upgrades, 241–242
 - with Zypp, 263–265
 - RPM packages, 246–248
 - software from source code, 24–25
 - compilation example, 27–28
 - components of installations, 26
 - makefiles, 26–27
 - tarballs, 25
 - YUM packages, 255–257, 258–259
 - integers, testing, 492
 - interface management, 188
 - arp command, 194, 198–199
 - Debian, 196–197
 - hostname command, 194, 197–198
 - ifcfg command, 194
 - ifcfg-* files, 195–196
 - ifconfig command, 194, 195
 - ip command, 188–191
 - iproute2 toolset, 188–189
 - net-tools suite, 194
 - NetworkManager, 192–194
 - network-scripts, configuring, 195–196
 - route command, 194, 199
 - ss command, 191–192
 - troubleshooting
 - collisions, 680–683
 - dropped packets, 680–683
 - I/O schedulers
 - bfq I/O schedulers, 645
 - cfq I/O schedulers, 644
 - change persistent I/O schedulers, 646–647
 - deadline I/O schedules, 644
 - kyber I/O schedulers, 645
 - mq-deadline I/O schedulers, 645
 - multiqueue I/O schedules, 644
 - none I/O schedulers, 645
 - noop I/O schedulers, 644
 - overview, 644
 - setting, 646
 - troubleshooting, 643, 645–647
 - types of, 644–645
 - viewing, 645
 - IOPS (Input/Output Operations Per Second), 631
 - irrelevancy of, 632–633
 - overview, 632–633
 - iostat command, 123–124, 629–631
 - iotop command, high latency issues, 626–627
 - iowait time, 713
 - ip command, 188–191, 670–672
 - ip route command, troubleshooting network
 - configuration issues, 672–673
 - iproute2 toolset, 188–189
 - IPset utility, 400
 - iptables, packet filtering, 382, 383, 385
 - by destination, 392–393
 - filtering points, 385–388
 - important terms, 388–389
 - incoming packets, 389–391
 - by multiple criteria, 392
 - outgoing packets, 395–396
 - overview, 385–388
 - by protocol, 391–392
 - ISO files, boot process, 13–14
 - Istio service meshes, 611–612
- ## J
- job control, 155–156
 - join command, 514–515
 - journaling, troubleshooting, 760
 - JSON (JavaScript Object Notation), 577–579
- ## K
- ### kernel
- configuring, 289
 - modules, 293–300
 - setting kernel parameters, 291–293
 - viewing kernel parameters, 290–291
 - images, 19
 - live kernel patching, 273–275
 - modules, 293–294
 - lsmod command, 294–295
 - managing, 294–295
 - manually loading/unloading, 296–297
 - modprobe command, 298–300
 - panic, 22
 - causes of, 23
 - getting more information, 22
 - identifying, 22
 - security, service accounts, 347
 - sysctl files, 291–293
 - updating, 270–271
 - choosing update methods, 271

- Linux kernel utilities, 272–273
 - manual updates, 271–272
 - no reboot method, 273–275
 - package managers, 272
 - reboot methods, 271–273
 - viewing parameters, 290–291
 - kill command, 149–150
 - killall command, 149–150
 - killing processes, 149–150, 152–153, 716
 - Kubernetes (K8)
 - benefits of, 603–604
 - containers, 602
 - ambassador containers, 603
 - bind mounts, 606–607
 - bridging networks, 609–610
 - CRI, 602
 - CRI-O, 602
 - Docker Swarms, 608–609
 - Docker volumes, 605–606
 - host networks, 610–611
 - networks, 608–611
 - overlay networks, 608, 609
 - persistent storage, 605, 607
 - PV subsystems, 607–608
 - PVC subsystems, 608
 - sidecars, 603
 - single-node multicontainer use cases, 604–605
 - specialized types, 603
 - swotting NAT, 610
 - three-container node example, 605
 - two-container node examples, 604–605
 - control plane, 601–602
 - defined, 600
 - high-level structure, 601
 - nodes, 601, 602
 - Pods, 601, 602
 - service meshes, 611
 - defined, 611
 - example of, 611–612
 - Istio service meshes, 611–612
 - NGINX Service Mesh, 612
 - kyber I/O schedulers, 645
- L**
- languages, final exam, 765
 - last command, 730
 - LastLine mode, vim, 83, 87
 - lastlog command, 730–731
 - latency and bandwidth, 686
 - LDAP (Lightweight Directory Access Protocol), 327, 329
 - libraries, systemd command, 160
 - lines (vim), deleting, 88–89
 - link files, 24
 - links
 - hard links, 62–63
 - soft links, 60–61
 - symbolic links, 61
 - troubleshooting, 683–684
 - data link layer, 685
 - physical layer, 684–685
 - Linux hardening, 333
 - insecure services, disabling/removing, 342
 - kernel security, service accounts, 347
 - passwords
 - aging, 344–345
 - group passwords, 345
 - setting parameters, 343
 - strength of, 333–338
 - umask, 340–342
 - unused packages, removing, 345–347
 - vulnerability scans
 - nc command, 338–340
 - nmap command, 333–338
 - Linux kernel utilities, kernel updates, 272–273
 - list command, 534
 - listing
 - directories, 49–50
 - files, 49–50
 - archived files, 71–72
 - cpio command, 72–73
 - dd command, 73–74
 - tar command, 71–72
 - hardware info, 35
 - repositories, 266
 - services, 166
 - live kernel patching, 273–275
 - load average (high), troubleshooting, 707–708
 - loaders and files, boot, 14
 - loading/unloading kernel modules, 296–297
 - local Debian packages, managing, 230–231
 - local variables, shell scripting, 469
 - localectl command, 307
 - locales
 - contents of, 306
 - fallback locales, 306
 - Linux use of, 307–308

- localectl command, 307
 - representing, 304–305
- locking files, 541
- locking up valuables, final exam, 768
- lockouts, user, 329–330
- logger command, 312
- logging, firewall rules, 396–397
- logging, syslog, 304
 - configuring, 312–315
 - key file locations, 313–314
 - locales
 - contents of, 306
 - fallback locales, 306
 - Linux use of, 307–308
 - localectl command, 307
 - log facilities, 309–310
 - log flows, 310–312
 - representing, 304
 - logger command, 312
 - severity levels, 310
 - systemd command, 308–315
- logical partitions, 29
- login issues, troubleshooting, 728
 - case sensitivity, 730
 - first-time logins, 730
 - inspecting account details, 728–729
 - last command, 730
 - last logins, 730–731
 - lastlog command, 730–731
- login shells, 371–372
- logs, container, 531–532
- loops
 - for loops, 496–497
 - sequences, 497
 - until loops, 498
 - while loops, 498
- low throughput
 - causes/symptoms, 627–628
 - diagnosing, 628–631
 - fixing, 628–631
 - overview, 627
- ls command, 49–50
- ls* command, 36–37
- lsattr command, 442–444, 733–734
- lsblk command, 105–106
- lscpu command, 719–720
- lsmem command, 720
- lsmode command, 294–295
- lsuf command, 153–155

- lsscsi command, 104–105
- LUKS (Linux Unified Key Setup), 123
- LVM (Logical Volume Manager)
 - creating, 117
 - managing, 117–118
 - multipathing, 116–117
 - overview, 114–116
 - troubleshooting
 - causes/symptoms, 657
 - diagnosing, 657–659
 - fixing, 657–659
 - overview, 656–657
- lvs command, 117–118

M

- makefiles, software installations from source
 - code, 26–27
- managing
 - containers, 525
 - installing tools, 525–526
 - verifying tools, 526
 - filesystems
 - Btrfs tools, 128–130
 - EXT2/3/4 tools, 127
 - fsck tool, 126–127
 - tune2fs command, 128
 - XFS commands, 130–131
 - identity
 - GID, 358–360
 - UID, 355–358
 - user identity query options, 372–374
- interfaces, 188
 - arp command, 194, 198–199
 - Debian, 196–197
 - hostname command, 194, 197–198
 - ifcfg command, 194
 - ifcfg-* files, 195–196
 - ifconfig command, 194, 195
 - ip command, 188–191
 - iproute2 toolset, 188–189
 - net-tools suite, 194
 - NetworkManager, 192–194
 - network-scripts, 195–196
 - route command, 194, 199
 - ss command, 191–192
 - troubleshooting, 679–683
- kernel modules, 294–295
- LVM, 117–118
- packages, 228, 229

- common package types, 228
- compressed files, 228
- Debian packages, 228, 229–235
- graphical managers, 242
- kernel updates, 272
- remote packages, 239–241
- removing packages, 242
- repositories, 235–239, 265–268
- RPM packages, 228, 243–255
- system upgrades, 241–242
- YUM packages, 255–262
- Zypp, 262–268
- repositories, with Zypp, 265–268
- systemd command, 165–169
- manually loading/unloading kernel modules, 296–297
- manually mounting filesystems, 121
- mapping ports, 532–533
- masking services, 169
- masks, ACL, 447–448
- MASQUERADE, 397
- MBR (Master Boot Records), 14, 29, 112
- mdadm command, 653, 656
- measuring CPU times, 711–712
- mem=xxxM command, 21–22
- memory
 - blocks, 718
 - buffers, 718
 - caches, 718
 - displaying, 145
 - exhaustion, 713–714
 - free command, 145
 - killing processes, 716
 - leaks, 716
 - NVMe, troubleshooting, 647–649
 - OOM, 714–716
 - OOM Killer, 716
 - pages, 718
 - RAM, swapping, 717–718
 - reclaiming, 716
 - slabs, 718
 - swapping, 717–718, 719
 - troubleshooting
 - exhaustion, 713–714
 - killing processes, 716
 - leaks, 716
 - OOM, 714–716
 - viewing hardware information, 720
- merge command, 590
- merges, version control, 542–543, 544–546, 562–568
- message digests, PKI, 324–325
- message line, vim, 83
- metadata, displaying, 58–60
- MFA (Multifactor Authentication), 326–327
- mirroring, RAID, 34
- mismatched filesystems, troubleshooting, 642–643
- mkdir command, 56
- modifying
 - group accounts, 365–366
 - user accounts, 364–365
- modprobe command, 298–300
- modules, kernel, 293–294
 - lsmod command, 294–295
 - managing, 294–295
 - manually loading/unloading, 296–297
 - modprobe command, 298–300
- monitoring
 - disk usage/storage
 - df command, 125–126
 - du command, 124–125
 - iostat command, 123–124
 - networks, 207
 - proper data flow, 211–217
 - remote host reachability, 207–211
 - RAID, 654–656
- mount command, 660
- mount options, troubleshooting
 - causes/symptoms, 659
 - diagnosing, 660–663
 - errors= option, 661–663
 - fixing, 660–663
 - overview, 659
 - UUID, 660–661
- mount unit files, troubleshooting, 750–752
- mounting devices
 - filesystem table, 118–121
 - manually mounting filesystems, 121
 - systemd command, 122–123
 - unmounting filesystems, 121–122
- moving objects, 54–56
- mq-deadline I/O schedulers, 645
- mtr command, 210–211
- multicontainers

- single-node multicontainers, use cases, 604–605
 - three-container node example, 605
 - two-container node examples, 604–605
 - multipathd command, 132–133
 - multipathing
 - LVM, 116–117
 - NAS, 132–133
 - SAN, 132–133
 - multiple command operators, shell scripting, 483–484
 - multiple periods (.) in filenames, 46–47
 - multiqueue I/O schedules, 644
 - mv command, 54–56
- N**
- name resolution, 199–200
 - configuring, 200–202
 - controlling, 202–203
 - diagram, 200
 - dig command, 205–206
 - host command, 204–205
 - hostnames on systemd systems, 204
 - nslookup command, 205
 - troubleshooting, 687–689, 756
 - whois command, 206–207
 - naming
 - containers, 529
 - files
 - periods (.) in filenames, 46–47
 - spaces in filenames, 47
 - RPM packages, 244–245
 - nano, nano, 90–92
 - NAS (Network-Attached Storage), 131–132
 - multipathing, 132–133
 - NFS, 133
 - Samba, 133
 - NAT (Network Address Translation), 397–398
 - DNAT, 397
 - MASQUERADE, 397
 - SNAT, 397
 - swotting, 610
 - navigating
 - directories
 - advanced navigation, 48–49
 - basic navigation, 47–48
 - files, vim, 85–86
 - nc command, 81–82, 219–220
 - netstat command, 211–213, 681–682
 - net-tools suite, 194
 - networking services, troubleshooting, 746
 - NetworkManager, 192–194
 - networks
 - bridging networks, 609–610
 - configuration issues
 - causes/symptoms, 670
 - diagnosing, 670–674
 - fixing, 670–674
 - troubleshooting, 670–674
 - container networks, 608
 - bridging networks, 609–610
 - Docker Swarms, 608–609
 - host networks, 610–611
 - overlay networks, 608, 609
 - swotting NAT, 610
 - firewalls, troubleshooting, 674–679
 - host networks, 610–611
 - interface management, 188
 - arp command, 194, 198–199
 - Debian, 196–197
 - hostname command, 194, 197–198
 - ifcfg command, 194
 - ifcfg-* files, 195–196
 - ifconfig command, 194, 195
 - ip command, 188–191
 - iproute2 toolset, 188–189
 - net-tools suite, 194
 - NetworkManager, 192–194
 - network-scripts, 195–196
 - route command, 194, 199
 - ss command, 191–192
 - monitoring, 207
 - proper data flow, 211–217
 - remote host reachability, 207–211
 - name resolution, 199–200
 - configuring, 200–202
 - controlling, 202–203
 - diagram, 200
 - dig command, 205–206
 - host command, 204–205
 - hostnames on systemd systems, 204
 - nslookup command, 205
 - whois command, 206–207
 - NAS, 131–132
 - multipathing, 132–133

- NFS, 133
 - Samba, 133
- NAT, swotting, 610
- overlay networks, 608, 609
- remote networking, 217
 - curl command, 218–219
 - nc command, 219–220
 - SSH, 217–218
 - wget command, 218–219
- remote system tests
 - legality of, 689
 - nmap command, 690–693
 - purposes of, 689
 - s_client module, 693–696
 - service discovery scans, 691–692
 - simple system scans, 690–691
 - vulnerability scans, 692–693
- SAN, 131–132
 - multipathing, 132–133
 - NFS, 133
 - Samba, 133
- troubleshooting
 - bandwidth, 686–687
 - configuration issues, 670–674
 - firewalls, 674–679
 - interfaces, 679–683
 - link status, 684–685
 - name resolution, 687–689
 - remote system tests, 689–696
- NFS (Network File System), 12–13, 133
- nftables, 383–384
- NGINX Service Mesh, 284, 612
- nice command, 157, 161
- nicknames, crontab configurations, 175
- nmap command
 - remote system tests, 690–693
 - troubleshooting, firewalls, 676
 - vulnerability scans
 - nc command, 338–340
 - nmap command, 333–338
- nmcli command, 192–194
- no reboot method, kernel updates, 273–275
- nodes, Kubernetes (K8), 601, 602
- none I/O schedulers, 645
- non-policy issue, troubleshooting, 734–735
- noop I/O schedulers, 644
- note-taking, final exam, 768
- nslookup command, 205
- NTP (Network Time Protocol)

- chrony command, 302–303
- configuring, 301–303
- timestamps, 302
- number of questions, final exam, 765
- NVMe, troubleshooting
 - causes/symptoms, 647–648
 - diagnosing, 648–649
 - fixing, 648–649
 - overview, 647
- nvme command, 648–649

O

- object storage, 31
- octal mode, chmod command, 432–433
- OnBootSec, 750
- OnCalendar, 750
- OOM (Out Of Memory), troubleshooting, 714–716
- OOM Killer, 716
- OTP (One-Time Passwords), 327
- output, crontab configurations, 174–175
- output streams, shell scripting, 475
- overlay networks, 608, 609
- ownership, files/directories, 434–436

P

- package configuration files, SSH, 408
- package management, 228, 229
 - common package types, 228
 - compressed files, 228
 - Debian packages, 228, 229
 - dependency issues, 232–233
 - dpkg command, 231–235
 - installing, 231
 - local packages, 230–231
 - querying, 233–234
 - reconfiguring, 234–235
 - removing, 231–232
 - graphical managers, 242
 - kernel updates, 272
 - remote packages, installing, 239–241
 - removing packages, 242
 - repositories, 235–236
 - adding, 238–239
 - configured repositories, 236–238
 - defined, 236
 - listing, 262–263
 - viewing configured repositories, 236–238

- RPM packages, 228, 243
 - files, 244–245
 - freshening, 249–250
 - installing, 246–248
 - naming conventions, 244–245
 - querying, 252–255
 - removing, 250–251
 - rpm command, 245
 - RPM database, 243–244
 - upgrading, 249–250
 - validating, 246
 - verifying integrity, 248–249
- system upgrades, 241–242
- unused packages, removing, 345–347
- YUM packages, 255
 - configuring, 259–262
 - DNF, 262
 - finding packages to install, 258–259
 - installing, 255–257, 258–259
 - updating, 257–258
- ZYpp, 262–263
 - installing packages, 263–265
 - removing packages, 265
 - repositories, 265–268
- packets
 - dropped, 680–683
 - filtering
 - by destination, 392–393
 - filtering points, 385–388
 - important terms, 388–389
 - incoming packets, 389–391
 - by multiple criteria, 392
 - outgoing packets, 395–396
 - overview, 385–388
 - by protocol, 391–392
- pages, 146, 718
- PAM (Pluggable Authentication Modules), 327–329
 - configuring, 330–331
 - LDAP integration, 329
 - SSDD, 331–332
 - user lockouts, 329–330
- panic, kernel, 22
 - causes of, 23
 - getting more information, 22
 - identifying, 22
- panic=#seconds command, 21
- parity, RAID, 35
- parted command
 - disk partitioning, 112–113
 - mismatched filesystems, 643
- partitions, storage, 28
 - disk partitioning, 108
 - fdisk command, 108–112
 - MBR partition tables, 112
 - parted command, 112–113
 - partprobe command, 113–114
 - GPT, 29
 - logical partitions, 29
 - MBR, 29
 - viewing information
 - blkid command, 106–108
 - fcstat command, 108
 - lsblk command, 105–106
- partprobe command, 113–114
- passing score, final exam, 765
- passwords
 - aging, 344–345
 - changing, 370–374
 - etc/passwd files, 355–361
 - group passwords, 345, 360
 - OTP, 327
 - PAM, 328–329
 - setting parameters, 343–345
 - SSH usage without, 411–413
 - strength of, 343–345
 - troubleshooting, 735–736
- paste command, 514–515
- PATH issues, crontab configurations, 173–174
- PATH variables, 471–472
- Pearson Test Prep practice exams, 768
 - access, 769–770
 - customizing, 770–771
 - Premium Edition, 771–772
 - updating, 771–772
- periods (.) in filenames, 46–47
- permissions (files), 429
 - chmod command, 432
 - octal mode, 432–433
 - SUID permissions, 438–439
 - symbolic mode, 433–434
 - finding files by permissions, 444–445
 - granularity issues, 437–438
 - ownership, 434–436
 - SGID permissions, 438–439, 440–441
 - special bit permissions, 438–439

- sticky bit permissions, 438–439
- SUID permissions, 438–440
- trio bits, 429–432
- troubleshooting, 732–733
- permissive mode, SELinux, 450
- persistent storage, containers, 605, 607
 - bind mounts, 606–607
 - Docker volumes, 605–606
 - PV subsystems, 607–608
 - PVC subsystems, 608
- pgrep command, 150–152
- physical layer (links), troubleshooting, 684–685
- physical preparations, final exam, 768
- PID, killing processes, 149–150
- pidof command, 151
- ping command, troubleshooting, 207–208
 - bandwidth issues, 686
 - firewalls, 676, 677
 - name resolution, 687, 689
 - network configuration issues, 672
- pipes (|), shell scripting, 481–483
- pkexec command, 420–421
- PKI (Public Key Infrastructure), 323
 - certificates
 - authentication, 323
 - authorities, 323–324
 - HTTPS, 325
 - purpose of, 323
 - self-signed certificates, 323
 - SSL, 325–326
 - use cases, 325–326
 - digital signatures, 325
 - encryption, 324–325
 - hashing, 324–325
 - message digests, 324–325
 - private keys, 324
 - public keys, 324
- pkill command, 150–152
- playbooks, Ansible, 581–583
- Podman container tool, 526
- Pods, Kubernetes (K8), 601, 602
- policies
 - dontaudit policy, 734–735
 - SELinux policies, 451–452
 - troubleshooting, 734–735
- PolicyKit, 420
- ports
 - exposing, 532–533
 - forwarding, 415
 - mapping, 532–533
- PR (Pull Requests), 591
- practice exams, 768–772
- preparing for exams
 - chapter-ending review tools, 772
 - domains, 766–767
 - earplugs, 767
 - fees, 766
 - getting rest, 768
 - ID code, 766
 - languages, 765
 - locking up valuables, 768
 - Pearson Test Prep practice exams, 768
 - access, 769–770
 - customizing, 770–771
 - Premium Edition, 771–772
 - updating, 771–772
 - physical preparations, 768
 - practice exams, 768–772
 - questions
 - budgeting time, 767
 - DITKA questions, 772
 - number of, 765
 - types of, 765
 - required passing score, 765
 - study trackers, 767
 - suggested review/study plans, 772
 - taking notes, 768
 - time limit, 765
 - tips for preparing, 767–768
 - travel time, 768
- primary groups, 358
- printf command, 94
- prioritizing, processes, 157–159, 713
- private keys, PKI, 324
- privileges
 - elevation, troubleshooting, 736
 - user privilege escalation, 418–419
- /proc filesystem, 36
 - dmidecode command, 37–38
 - ls* commands, 36–37
- procedural tools, IaC, 581
- processes
 - blocks, 146
 - bounce signals, 149
 - buffers, 146
 - caches, 146

- interpreting displayed data from free command, 147–148
 - job control, 155–156
 - killing, 149–150, 152–153, 716
 - pages, 146
 - prioritizing, 157–159, 713
 - querying tables, 150–152
 - resource usage, 153–155
 - runaway processes, 704–705
 - sending signals, 148–149
 - pgrep command, 150–152
 - pkill command, 150–152
 - services
 - crontab configurations, 170–175
 - disabling, 166–167
 - enabling, 166–167
 - listing, 166
 - masking, 169
 - querying status, 167–169
 - scheduling, 170–180
 - starting, 167–169
 - state of, 165
 - stopping, 167–169
 - slabs, 146
 - systemd command, 159–161
 - boot process, 164–165
 - daemons, 160
 - environment variables, 162
 - libraries, 160
 - managing, 165–169
 - requires, 163–164
 - runlevels, 163
 - targets, 163
 - units, 161–162
 - wants, 163–164
 - units, state of, 165
 - viewing, 142
 - free command, 145, 147–148
 - htop command, 144–145
 - ps command, 142–144
 - pstree command, 143–144
 - zombie processes, 150, 705–706
 - ps command, 142–144
 - pstree command, 143–144
 - public keys, PKI, 324
 - pull command, 534
 - pulling container images, 527–528
 - Puppet, 583
 - push command, 533–534
 - PV subsystems, 607–608
 - PVC subsystems, 608
 - pvs command, 117
 - PXE boot process, 11–12
- ## Q
- qbang, 87
 - querying
 - Debian packages, 233–234
 - process tables, 150–152
 - RPM packages, 252–255
 - service status, 167–169
 - questions, final exam
 - budgeting time, 767
 - DITKA questions, 772
 - number of, 765
 - types of, 765
 - quitting vim, 87
 - quotas, troubleshooting, 736–738
- ## R
- RAID (Redundant Array of Independent Disks), 32–33
 - creating devices, 33–34
 - mirroring, 34
 - parity, 35
 - RAID0, 33
 - RAID1, 33
 - RAID3, 33
 - RAID5, 33
 - software RAID
 - creating, 114
 - inspecting, 114
 - striping, 34–35
 - troubleshooting, 652
 - array health, 655–656
 - causes/symptoms, 653
 - device health, 654
 - diagnosing, 653–656
 - fixing, 653–656
 - monitoring, 654–656
 - RAM, swapping, 717–718
 - rebase command, 590
 - reboot methods
 - kernel updates, 271
 - manual updates, 271–272

- recipes, Chef, 583
- reclaiming memory, 716
- reconfiguring Debian packages, 234–235
- redirecting streams, 478, 480
 - /dev/tty, 480–481
 - standard errors, 479–480
 - standard input, 478
 - standard output, 478–479
- refreshing
 - firewalls, 674–675
 - repositories, 267–268
- registries, container, 613–614
- regular expressions, shell scripting, 511–514
- relative paths, 47, 48, 50
- reloading services, 283–284
- remote connectivity
 - pkexec command, 420–421
 - PolicyKit, 420
 - SSH, 408
 - components of, 408–414
 - executing commands as another user, 416–418, 419
 - global configuration files, 408–409
 - package configuration files, 408
 - port forwarding, 415
 - SSH client command, 409–410
 - ssh-agent command, 413
 - ssh-copy-id command, 414
 - tunneling, 414–415
 - user privilege escalation, 418–419
 - using without passwords, 411–413
 - X11 forwarding, 414–415
 - X11 tunnels, 411
- remote hosts, reachability (network monitoring), 207
 - mtr command, 210–211
 - ping command, 207–208
 - traceroute command, 208–210
- remote networking, 217
 - curl command, 218–219
 - nc command, 219–220
 - SSH, 217–218
 - wget command, 218–219
- remote packages, installing, 239–241
- remote system tests
 - legality of, 689
 - nmap command, 690–693
 - purposes of, 689
 - s_client module, 693–696
 - service discovery scans, 691–692
 - simple system scans, 690–691
 - vulnerability scans, 692–693
- removing
 - containers, 531
 - Debian packages, 231–232
 - directories, 56–57
 - file attributes, 443–444
 - group accounts, 367
 - insecure services, 342
 - objects, 57
 - packages, 242, 265
 - Debian packages, 231–232
 - RPM packages, 250–251
 - unused packages, 345–347
 - user accounts, 366–367
- renice command, 153, 157–158, 161
- replacing text, vim, 88
- repositories, 235–236
 - adding, 238–239, 266
 - APT repository files, 289
 - configuration files, 287–288
 - APT configuration files, 288
 - DNF configuration files, 289
 - YUM configuration files, 288
 - configured repositories, 236–238
 - defined, 236
 - hosts, Git, 549
 - listing, 266
 - managing with Zypp, 265–268
 - refreshing, 267–268
 - viewing configured repositories, 236–238
 - YUM repository files, 287–288
- required passing score, final exam, 765
- requires, systemd command, 163–164
- Requires directives, 749
- resolvectl command, 202–203
- resource usage, processes, 153–155
- rest (final exam), getting, 768
- restarting services, 283
- restricting cron job access, 177
- review/study plans, final exam, 772
- .rhosts files, deleting, 411
- rm command, 57
- rmdir command, 56–57
- rmi command, 534
- ro command, 21–22
- /root directory, overview, 6–7
- route command, 194, 199

routing

- ip route command, 672–673
- traceroute command, 673–674

rpm command, 245

RPM packages, 228, 243

- configuring, 284–287
- files, 244–245
- freshening, 249–250
- installing, 246–248
- naming conventions, 244–245
- querying, 252–255
- removing, 250–251
- rpm command, 245
- RPM database, 243–244
- upgrading, 249–250
- validating, 246
- verifying integrity, 248–249

rsync command, 79–80

rules (firewalls)

- creating, 385–397
- saving, 394–395

run queues (high), troubleshooting, 708–711

runaway processes, troubleshooting, 704–705

runlevels, systemd command, 163

running

- ad hoc cron jobs, 178–180
- images as containers, 529
- scripts, 473–474
- services, firewalls, 678–679

runtime firewalls, 384

rw command, 21–22

S

s_client module, remote system tests, 693–696

SaltStack, 584–586

Samba, 133

SAN (Storage Area Networks), 131–132

- multipathing, 132–133
- NFS, 133
- Samba, 133

sandboxed applications, 268, 269

- AppImage, 269
- defined, 268
- Flatpak, 269
- Snapd, 270

saving

- files, vim, 87
- firewall rules, 394–395

scans

- remote systems
 - service discovery scans, 691–692
 - simple system scans, 690–691
 - vulnerability scans, 692–693
- vulnerability scans
 - nc command, 338–340
 - nmap command, 333–338

scheduling services, 170–180

scoring, final exam, 765

scp command, 78–79

scripts, network-scripts, 195–196

SCSI device information, viewing, 104–105

searching in vim, 89–90

secondary groups, 358

second-generation version control, 542–543, 544–545

security

- authentication, 326
 - biometrics, 327
 - certificates, 323
 - LDAP, 327, 329
 - MFA, 326–327
 - OTP, 327
 - PAM, 327–332
 - SSO, 332–333
- etc/group files, 358–361
- etc/passwd files, 355–361
- etc/profile files, 371
- etc/shadow files, 356, 357, 365, 368–370
- etc/skel files, 361, 362–363
- firewalls, 382–383
 - changing default firewall policies, 393–394
 - creating rules, 385–397
 - defined, 382
 - DenyHosts utility, 400
 - fail2ban service, 398–400
 - firewalld command, 383–384
 - IPset utility, 400
 - iptables, 382, 383, 385–397, 398
 - logging rules, 396–397
 - NAT, 397–398
 - nftables, 383–384
 - packet filtering, 385–397
 - refreshing, 674–675
 - runtime firewalls, 384
 - saving rules, 394–395

- stateful firewall rules, 396
- stateless firewall rules, 396
- troubleshooting, 674–679
- ufw, 384–385
- hardening Linux, 333
 - disabling/removing insecure services, 342
 - password strength, 333–338
 - vulnerability scans, 333–340
- identity management
 - GID, 358–360
 - UID, 355–358
- kernel, service accounts, 347
- locking up valuables, final exam, 768
- login shells, 371–372
- LUKS, 123
- passwords
 - aging, 344–345
 - changing, 370–374
 - etc/passwd files, 355–361
 - group passwords, 345, 360
 - OTP, 327
 - PAM, 328–329
 - setting parameters, 343
 - SSH usage without, 411–413
 - strength of, 343–345
 - troubleshooting, 735–736
- PKI, 323
 - certificates, 323–324, 325–326
 - digital signatures, 325
 - encryption, 324–325
 - hashing, 324–325
 - message digests, 324–325
 - private keys, 324
 - public keys, 324
 - service accounts, 347
 - Shadow Suite, 368–370
- sed (Stream Editor), 503–505
- self-signed certificates, 323
- SELinux (Security-Enhanced Linux), 449–450
 - audit2allow command, 455–456
 - booleans, 452–454
 - contexts, 454–455
 - modes, 450–451
 - policies, 451–452
- semicolons (;), shell scripting, 483
- sending signals to processes, 148–149
 - pgrep command, 150–152
 - pkill command, 150–152
- sequences, loops, 497
- service accounts, security, 347
- service meshes, 611
 - defined, 611
 - example of, 611–612
 - Istio service meshes, 611–612
 - NGINX Service Mesh, 612
- services
 - configuring, 300–301
 - NTP, 301–303
 - SSH, 301
 - crontabs, configuring, 170–175
 - disabling, 166–167
 - discovery scans, 691–692
 - enabling, 166–167
 - fail2ban service, 398–400
 - firewall operations, troubleshooting, 678–679
 - insecure services, disabling/removing, 342
 - listing, 166
 - masking, 169
 - networking services, troubleshooting, 746
 - querying status, 167–169
 - reloading, 283–284
 - restarting, 283
 - scheduling, 170–180
 - start times, troubleshooting, 760
 - starting, 167–169
 - state of, 165
 - stopping, 167–169
- setfacl command, 446
- setting
 - ACL, 446–447
 - file attributes, 443
 - I/O schedulers, 646
 - kernel parameters, 291–293
 - password parameters, 343
- SGID permissions, 438–439, 440–441
- Shadow Suite, 368–370
- shell scripting
 - accepting arguments, 499–500
 - ampersands (&&), 483–484
 - awk command, 500–501
 - basics of, 466–467
 - case statements, 493–495
 - conditionals, 488–489
 - cut command, 502–503
 - cutting columns, 502–503
 - egrep command, 510–511
 - fgrep command, 510–511

- file descriptors, 475
- finding
 - errors on demand, 476–477
 - files, 515–517
- globbing, 467–468
- good script design, 474–475
- grep command, 498–499, 500–501, 505–514
- heredocs, 477
- input/output streams, 475
- interactions with other programs, 498–499
- join command, 514–515
- loops
 - for loops, 496–497
 - sequences, 497
 - until loops, 498
 - while loops, 498
- multiple command operators, 483–484
- output of another command, using, 487–488
- paste command, 514–515
- pipes (`|`), 481–483
- processing output, 485–486
- regular expressions, 511–514
- returning error codes, 499
- running scripts, 473–474
- sed, 503–505
- semicolons (`;`), 483
- splitting streams, 485
- stderr command, 476
- stdin command, 475
- stdout command, 475–476
- stream redirection, 478, 480
 - `/dev/tty`, 480–481
 - standard errors, 479–480
 - standard input, 478
 - standard output, 478–479
- substituting commands, 484–485
- switch statements, 495–496
- tee command, 485
- testing
 - combining multiple tests, 493
 - files, 490–491
 - integers, 492
 - strings, 491–492
- tr command, 502
- translating files, 502
- variables
 - creating, 470–471
 - displaying, 469–471
 - environment variables, 469
 - expanding variables, 472–473
 - local variables, 469
 - PATH variables, 471–472
 - SHELL variables, 472
 - xargs command, 485–486
- SHELL variables, 472
- sidecars, 603
- signals
 - bounce signals, 149
 - sending to processes, 148–149
 - pgrep command, 150–152
 - pkill command, 150–152
- SIGHUP, 148
- SIGINT, 148
- SIGKILL, 148
- SIGSTOP, 148
- SIGTERM, 148
- SIGTSTP, 148
- signatures, digital, 325
- simple system scans, 690–691
- single or 1 command, 21
- single-node multicontainers, use cases, 604–605
- Skopeo container tool, 526
- slabs
 - memory, 718
 - processes, 146
- smartctl command, 653, 654
- smartd command, 653–654
- SMB (Server Message Blocks), 133
- Snapd, 270
- snapshots, 77
- SNAT (Source NAT), 397
- sockets, displaying information, 191–192
- soft links, 60–61
- software
 - configuration files, updating, 283–284
 - dates/time zones, configuring, 303–304
 - downloading
 - curl command, 218–219
 - wget command, 218–219
 - installing from source code, 24–25
 - compilation example, 27–28
 - components of installations, 26
 - makefiles, 26–27
 - tarballs, 25
 - kernel configurations, 289
 - modules, 293–300

- setting kernel parameters, 291–293
 - viewing kernel parameters, 290–291
- kernel updates, 270–271
 - choosing update methods, 271
 - Linux kernel utilities, 272–273
 - live kernel patching, 273–275
 - manual updates, 271–272
 - no reboot method, 273–275
 - package managers, 272
 - reboot methods, 271–273
- package management, 228, 229
 - common package types, 228
 - compressed files, 228
 - Debian packages, 228
 - graphical managers, 242
 - kernel updates, 272
 - remote packages, 239–241
 - removing packages, 242
 - repositories, 235–239, 265–268
 - RPM packages, 228, 243–255
 - system upgrades, 241–242
 - YUM packages, 255–262
 - ZYpp, 262–268
- repository configuration files, 287–288
 - APT configuration files, 288
 - DNF configuration files, 289
 - YUM configuration files, 288
- repository files
 - APT repository files, 289
 - YUM repository files, 287–288
- RPM packages, configuring, 284–287
- sandboxed applications, 268, 269
 - AppImage, 269
 - defined, 268
 - Flatpak, 269
 - Snapd, 270
- services
 - configuring, 300–304
 - reloading, 283–284
 - restarting, 283
- syslog, 304
 - configuring, 312–315
 - key file locations, 313–314
 - locales, 304–308
 - log facilities, 309–310
 - log flows, 310–312
 - logger command, 312
 - severity levels, 310
 - systemd command, 308–315
- software RAID
 - creating, 114
 - inspecting, 114
- software tokens, 326–327
- source code, software installations, 24–25
 - compilation example, 27–28
 - components of installations, 26
 - makefiles, 26–27
 - tarballs, 25
- source control, IaC, 580
- spaces in filenames, 47
- special bit permissions, 438–439
- splitting streams, shell scripting, 485
- ss command, 191–192
- SSD TRIM, 651
 - garbage collection, 651–652
 - TRIM Helper, 652
- SSD, troubleshooting
 - causes/symptoms, 649–650
 - diagnosing, 650
 - fixing, 650
 - overview, 649
 - SSD TRIM, 651–652
- SSDD (System Security Services Daemon), 331–332
- SSH (Secure Shell), 217–218, 408
 - components of, 408–414
 - configuring, 301
 - executing commands as another user, 416
 - su command, 419
 - sudo command, 416
 - sudoedit command, 417–418
 - global configuration files, 408–409
 - package configuration files, 408
 - port forwarding, 415
 - SSH client command, 409–410
 - ssh-agent command, 413
 - ssh-copy-id command, 414
 - tunneling, 414–415
 - user privilege escalation, 418–419
 - using without passwords, 411–413
 - X11 forwarding, 414–415
 - X11 tunnels, 411
- SSL (Secure Socket Layer), PKI certificates, 325–326
- SSO authentication, 332–333
- stages, Git, 548–549

- starting services, 167–169, 760
- stat command, 58–60, 733
- State files, Salt, 585–586
- stateful firewall rules, 396
- stateless firewall rules, 396
- stderr command, 476
- stdin command, 475
- stdout command, 475–476
- steal time, 712
- sticky bit permissions, 438
- stopping, services, 167–169
- storage
 - administrative tasks, 28
 - blocks, 30
 - container persistent storage, 605, 607
 - bind mounts, 606–607
 - Docker volumes, 605–606
 - PV subsystems, 607–608
 - PVC subsystems, 608
 - determining hardware, 104–105
 - /dev directory, 28
 - disk partitioning, 108
 - fdisk command, 108–112
 - MBR partition tables, 112
 - parted command, 112–113
 - partprobe command, 113–114
 - Docker volumes, 605–606
 - files, 30
 - filesystem management
 - Btrfs tools, 128–130
 - EXT2/3/4 tools, 127
 - fsck tool, 126–127
 - tune2fs command, 128
 - XFS commands, 130–131
 - LUKS encryption, 123
 - LVM
 - creating, 117
 - managing, 117–118
 - multipathing, 116–117
 - overview, 114–116
 - methods (overview), 29
 - monitoring
 - df command, 125–126
 - du command, 124–125
 - iostat command, 123–124
 - mounting devices
 - filesystem table, 118–121
 - manually mounting filesystems, 121
 - systemd command, 122–123
 - unmounting filesystems, 121–122
 - NAS, 131–132
 - multipathing, 132–133
 - NFS, 133
 - Samba, 133
 - objects, 31
 - partitions, 28
 - GPT, 29
 - logical partitions, 29
 - MBR, 29
 - viewing information, blkid command, 106–108
 - viewing information, fcstat command, 108
 - viewing information, lsblk command, 105–106
 - SAN, 131–132
 - multipathing, 132–133
 - NFS, 133
 - Samba, 133
 - SCSI storage information, viewing, 104–105
 - software RAID
 - creating, 114
 - inspecting, 114
 - troubleshooting
 - capacity issues, 633–638
 - filesystems, 638–643
 - high latency, 623–627
 - I/O schedulers, 643–647
 - IOPS, 631–633
 - low throughput, 627–631
 - LVM, 656–659
 - mount options, 659–662
 - NVMe, 647–649
 - RAID, 652–656
 - SSD, 649–652
- stream redirection, 478, 480
 - /dev/tty, 480–481
 - standard errors, 479–480
 - standard input, 478
 - standard output, 478–479
- strings, testing, 491–492
- striping, RAID, 34–35
- study trackers, 767
- study/review plans, final exam, 772
- su command, 419

subdirectories

- /root directory, 6–7
- /usr directory, 7–8
- (sub)Directory files, 23–24
- sudo command, 416
- sudoedit command, 417–418
- suggested review/study plans, final exam, 772
- SUID permissions, 438–440
- swapping, 717–718, 719
- Swarms, Docker, 608–609
- switch statements, 495–496
- swotting NAT, 610
- symbolic links, 61
- symbolic mode, chmod command, 433–434
- syntax checks, crontabs, 176
- sysctl files, 291–293
- sysfs, 37
- syslog, 304
 - configuring, 312–315
 - key file locations, 313–314
 - locales
 - contents of, 306
 - fallback locales, 306
 - Linux use of, 307–308
 - localectl command, 307
 - representing, 304
 - log facilities, 309–310
 - log flows, 310–312
 - logger command, 312
 - severity levels, 310
 - systemd command, 308–315
- system boot, 9
 - basics, 8
 - BIOS, 14
 - boot loaders and files, 14
 - common commands, 21–22
 - defined, 8
 - dracut framework, 10
 - GRUB, 14–15
 - GRUB2
 - changes made, 15–16
 - command line, 18–19
 - command names, 16–17
 - configuring, 20
 - installing, 17–18
 - kernel images, 19
 - initramfs method, 10
 - initrd method, 9–10
 - ISO files, 13–14
 - MBR, 14
 - NFS, 12–13
 - PXE, 11–12
 - stages of, 8
 - UEFI, 10–11
- system crontabs, 176
- system time, 712–713
- system updates, 270–271
 - choosing update methods, 271
 - Linux kernel utilities, 272–273
 - live kernel patching, 273–275
 - manual updates, 271–272
 - no reboot method, 273–275
 - package managers, 272
 - reboot methods, 271–273
- system upgrades, package installations, 241–242
- systemctl command, 165, 678–679
- systemd command, 122–123, 159–161
 - boot process, 164–165
 - daemons, 160
 - environment variables, 162
 - libraries, 160
 - managing, 165–169
 - requires, 163–164
 - runlevels, 163
 - syslog, 308–315
 - targets, 163
 - troubleshooting
 - Before/After directives, 747–748
 - ExecStart, 747
 - ExecStop, 747
 - mount unit files, 750–752
 - networking services, 746
 - Requires directives, 749
 - target unit files, 752–756
 - timer unit files, 750
 - types, 748–749
 - unit files, 745–756
 - users, 749
 - Wants directives, 749
 - units, 161–162
 - wants, 163–164
- systemd systems, hostnames, 204

T

- tags, Git, 552–555
- taking notes, final exam, 768
- tar command

- archiving files, 64–66, 71–72
 - compressing files, 66–70
- tarballs, 25
- target unit files, troubleshooting, 752–756
- targets, systemd command, 163
- tcpdump command, 216–217
- tee command, 485
- telnet command, troubleshooting firewalls, 676, 677–678
- Terraform, 586–588
- testing
 - combining multiple tests, shell scripting, 493
 - files, 490–491
 - integers, 492
 - remote systems
 - legality of, 689
 - nmap command, 690–693
 - purposes of, 689
 - s_client module, 693–696
 - service discovery scans, 691–692
 - simple system scans, 690–691
 - vulnerability scans, 692–693
 - strings, 491–492
- text, vim
 - changing, 88
 - deleting, 88–89
 - replacing, 88
- third-generation version control, 543–546
- three-container node example, Kubernetes (K8) multicontainers, 605
- throughput (low)
 - causes/symptoms, 627–628
 - diagnosing, 628–631
 - fixing, 628–631
 - overview, 627
- time budgets, final exam, 767
- time limit, final exam, 765
- time zones/dates, configuring, 303–304
- timedatectl command, 303–304
- timer unit files, troubleshooting, 750
- times, CPU, 711
 - idle time, 713
 - iowait time, 713
 - measuring, 711–712
 - steal time, 712
 - system time, 712–713
 - user time, 712
- timestamps, NTP, 302
- time-zone configurations, troubleshooting, 759–760
- tips for exam preparation, 767–768
- tload command, 710–711
- tokens
 - hardware tokens, 326
 - software tokens, 326–327
- top command, 158–159, 624
- touch command, 50–51
- touching files, 50–51
- tr command, 502
- traceroute command, 208–210, 673–674, 686
- translating files, 502
- travel time, final exam, 768
- TRIM Helper, 652
- trio bits, permissions, 429–432
- troubleshooting
 - ACL, 733
 - apps (applications), 756–758
 - attribute issues, 733–734
 - bandwidth, 686–687
 - capacity issues
 - causes/symptoms, 633–634
 - diagnosing, 634–638
 - fixing, 634–638
 - collisions, 680–683
 - context issues, 732
 - CPU
 - high CPU utilization, 707
 - high load average, 707–708
 - high run queues, 708–711
 - process priorities, 713
 - runaway processes, 704–705
 - zombie processes, 705–706
 - dropped packets, 680–683
 - errors= option, 661–663
 - file access issues, 731–732
 - ACL, 733
 - attribute issues, 733–734
 - blocks, 737–738
 - context issues, 732
 - group access, 732
 - non-policy issue, 734–735
 - passwords, 735–736
 - permissions, 732–733

- policy issues, 734–735
 - privilege elevation, 736
 - quotas, 736–738
- filesystems
 - corruption, 638–642
 - mismatches, 642–643
- firewalls
 - causes/symptoms, 675–676
 - diagnosing issues, 676–678
 - fixing issues, 676–679
- group file access issues, 732
- high CPU utilization, 707
- high latency issues, 670
 - causes/symptoms, 623
 - diagnosing, 624–626
 - fixing, 626–627
 - overview, 623
- high load average, 707–708
- high run queues, 708–711
- interfaces
 - collisions, 680–683
 - dropped packets, 680–683
- I/O schedulers, 645–647
- IOPS, 631
 - irrelevancy of, 632–633
 - overview, 632–633
- journaling, 760
- link status, 683–684
 - data link layer, 685
 - physical layer, 684–685
- login issues, 728
 - case sensitivity, 730
 - first-time logins, 730
 - inspecting account details, 728–729
 - last command, 730
 - last logins, 730–731
 - lastlog command, 730–731
- low throughput
 - causes/symptoms, 627–628
 - diagnosing, 628–631
 - fixing, 628–631
 - overview, 627
- LVM
 - causes/symptoms, 657
 - diagnosing, 657–659
 - fixing, 657–659
 - overview, 656–657
- memory
 - exhaustion, 713–714
 - killing processes, 716
 - leaks, 716
- mount options
 - causes/symptoms, 659
 - diagnosing, 660–663
 - errors= option, 661–663
 - fixing, 660–663
 - overview, 659
 - UUID, 660–661
- name resolution, 687–689, 756
- networking services, 746
- networks
 - bandwidth, 686–687
 - configuration issues, 670–674
 - firewalls, 675–679
 - interfaces, 679–683
 - link status, 684–685
 - name resolution, 687–689
 - remote system tests, 689–696
- non-policy issue, 734–735
- NVMe
 - causes/symptoms, 647–648
 - diagnosing, 648–649
 - fixing, 648–649
 - overview, 647
- OOM, 714–716
- passwords, 735–736
- permissions (files), 732–733
- policy issues, 734–735
- privilege elevation, 736
- quotas, 736–738
- RAID, 652
 - array health, 655–656
 - causes/symptoms, 653
 - device health, 654
 - diagnosing, 653–656
 - fixing, 653–656
 - monitoring, 654–656
- runaway processes, 704–705
- service start times, 760
- SSD
 - causes/symptoms, 649–650
 - diagnosing, 650
 - fixing, 650
 - overview, 649
 - SSD TRIM, 651–652
- storage
 - capacity issues, 633–638
 - filesystems, 638–643

- high latency, 623–627
 - I/O schedulers, 643–647
 - IOPS, 631–633
 - low throughput, 627–631
 - LVM, 656–659
 - mount options, 659–662
 - NVMe, 647–649
 - RAID, 652–656
 - SSD, 649–652
 - systemd command
 - Before/After directives, 747–748
 - ExecStart, 747
 - ExecStop, 747
 - mount unit files, 750–752
 - networking services, 746
 - Requires directives, 749
 - target unit files, 752–756
 - timer unit files, 750
 - types, 748–749
 - unit files, 745–756
 - users, 749
 - Wants directives, 749
 - target unit files, 752–756
 - timer unit files, 750
 - time-zone configurations, 759–760
 - unit files, 745
 - Before/After directives, 747–748
 - ExecStart, 747
 - ExecStop, 747
 - mount unit files, 750–752
 - networking services, 746
 - Requires directives, 749
 - target unit files, 752–756
 - timer unit files, 750
 - types, 748–749
 - users, 749
 - Wants directives, 749
 - user access
 - file access issues, 731–738
 - login issues, 728–731
 - UUID, 660–661
 - zombie processes, 705–706
 - tune2fs command, 128
 - tunneling, 414
 - port forwarding, 415
 - X11 forwarding, 414–415
 - X11 tunnels, 411
 - two-container node examples, Kubernetes
 - (K8) multicontainers, 604–605
- ## U
- UEFI, boot process, 10–11
 - ufw (Uncomplicated Firewalls), 384–385
 - UID (User ID), 355–358
 - umask, 340–342
 - undo operations, vim, 86–87
 - unit files, troubleshooting, 745
 - Before/After directives, 747–748
 - ExecStart, 747
 - ExecStop, 747
 - mount unit files, 750–752
 - networking services, 746
 - Requires directives, 749
 - target unit files, 752–756
 - timer unit files, 750
 - types, 748–749
 - users, 749
 - Wants directives, 749
 - units
 - state of, 165
 - systemd command, 161–162
 - unloading kernel modules, 296–297
 - unmounting filesystems, 121–122
 - unsecure services, disabling/removing, 342
 - until loops, 498
 - unused packages, removing, 345–347
 - updating
 - kernel, 270–271
 - choosing update methods, 271
 - Linux kernel utilities, 272–273
 - live kernel patching, 273–275
 - manual updates, 271–272
 - no reboot method, 273–275
 - package managers, 272
 - reboot methods, 271–273
 - Pearson Test Prep practice exams, 771–772
 - software configuration files, 283–284
 - systems, 270–271
 - choosing update methods, 271
 - Linux kernel utilities, 272–273
 - live kernel patching, 273–275
 - manual updates, 271–272
 - no reboot method, 273–275
 - package managers, 272
 - reboot methods, 271–273
 - YUM packages, 257–258
 - UPG (User Private Groups), 359
 - upgrading
 - RPM packages, 249–250

systems, package installations, 241–242
 user time, 712
 useradd command, 361–363
 userdel command, 366–367
 usermod command, 364–365
 users
 accounts
 adding, 361–363
 group accounts, 364
 inspecting details, 728–729
 removing, 366–367
 UID, 355–358
 etc/passwd files, 355–361
 executing commands as another user, 416
 su command, 419
 sudo command, 416
 sudoedit command, 417–418
 GID, 358–360
 group accounts, removing, 367
 identity query options, 372–374
 lockouts, PAM, 329–330
 modifying in
 group accounts, 364–365
 user accounts, 364–365
 privilege escalation, 418–419
 removing accounts, 366–367
 requests, FUSE, 31
 troubleshooting
 file access issues, 731–738
 login issues, 728–731
 unit files, 749
 UID, 355–358
 /usr directory, 7–8
 UUID, troubleshooting, 660–661

V

validating RPM packages, 246
 valuables (final exam), locking up, 768
 variables, shell scripting
 creating, 470–471
 displaying, 469–471
 environment variables, 469
 expanding variables, 472–473
 local variables, 469
 PATH variables, 471–472
 SHELL variables, 472
 verifying
 container tools, 526

RPM package integrity, 248–249
 service operation, firewalls, 678–679
 version control
 branches, 556–563
 check ins/outs, 542–544
 clones, 548
 comparing, versions, 560
 DVCS, 543–546
 first generation, 541–542
 ignoring files, 555–556
 merges, 542–543, 544–546, 562–568
 repository hosts, 549
 second generation, 542–543, 544–545
 stages, 548–549
 tags, 552–555
 third generation, 543–546
 vimdiff tool, 567
 whitespace, 560–561
 vga command, 21
 vgs command, 117
 viewing
 ACL, 446
 container images, 528
 container logs, 531–532
 crontabs, 175–176
 hardware CPU information, 719–720
 hardware memory information, 720
 I/O schedulers, 645
 kernel parameters, 290–291
 partition information
 blkid command, 106–108
 fcstat command, 108
 lsblk command, 105–106
 processes, 142
 free command, 145
 htop command, 144–145
 ps command, 142–144
 pstree command, 143–144
 repositories (configured), 236–238
 SCSI storage information, 104–105
 vim, 82–83
 changing text, 88
 Command mode, 83–84, 86–87, 89
 deleting text/lines, 88–89
 editing files, 83–84
 force multipliers, 86, 88
 Insert mode, 83–84
 LastLine mode, 83, 87
 message line, 83

- navigating files, 85–86
- qbang, 87
- quitting, 87
- replacing text, 88
- saving files, 87
- searching in, 89–90
- undo operations, 86–87
- vimdiff tool, 567
- vmlinux, 19
- vmlinuxz, 19
- vmstat command
 - high latency issues, 625–626
 - low throughput, 628
- vulnerability scans, 692–693
 - nc command, 338–340
 - nmap command, 333–338

W

- w command, 373–374
- wants, systemd command, 163–164
- Wants directives, 749
- wget command, 218–219
- while loops, 498
- whitelist policies, 177
- whitespace, difference execution, 560–561
- whois command, 206–207
- Wireshark, proper data flow, 214–216

X

- X11 forwarding, 414–415
- X11 tunnels, 411
- xargs command, 485–486
- XFS commands, 130–131

Y

- YAML (YAML Ain't Markup Language), 576–577

YUM

- configuration files, 288
- packages, 255
 - configuring, 259–262
 - DNF, 262
 - finding packages to install, 258–259
 - installing, 255–257, 258–259
 - updating, 257–258
- repository files, 289

Z

- zombie processes, 150, 705–706
- ZYpp, 262–263
 - installing packages, 263–265
 - managing repositories, 265–268
 - removing packages, 265