



Save 10%
on Exam
Voucher

See Inside

CompTIA®

Linux+™

Portable Command Guide

All the commands for the CompTIA XK0-004 exam in one compact, portable resource

PEARSON IT
CERTIFICATION

SECOND EDITION



WILLIAM “BO” ROTHWELL

FREE SAMPLE CHAPTER

SHARE WITH OTHERS



What Do You Want to Do Today?

| What Do You Want to Do? | Chapter | Page |
|---|----------------|-------------|
| Access software repositories | 7 | 88 |
| Archive and restore files | 20 | 268 |
| Compress files | 20 | 269 |
| Configure a boot loader | 1 | 6 |
| Configure bonding on network interfaces | 3 | 34 |
| Configure console redirection | 14 | 190 |
| Configure localization | 6 | 75 |
| Configure PAM | 16 | 214 |
| Configure remote desktop | 14 | 189 |
| Configure shell variables | 25 | 328 |
| Configure system date and time | 6 | 70 |
| Configure virtual machines | 5 | 67 |
| Configure wireless networking | 3 | 25 |
| Disable services | 17 | 242 |
| Display process states | 22 | 297 |
| Display user information | 8 | 96 |
| Edit files with the vi editor | 9 | 111 |
| Install software | 7 | 78 |
| Learn the boot process | 1 | 1 |
| Learn about basic partitions | 4 | 38 |
| List loaded kernel modules | 2 | 12 |
| Load kernel modules into memory | 2 | 13 |
| Manage command output | 9 | 117 |
| Manage file and directory permissions | 15 | 194 |
| Manage filesystem quotas | 8 | 98 |
| Manage Git repositories | 26 | 343 |
| Manage Linux devices | 13 | 176 |
| Manage processes | 22 | 300 |

(Continues on the inside back cover)

CompTIA® Linux+™ Portable Command Guide

All the commands for the CompTIA XK0-004
exam in one compact, portable resource

William “Bo” Rothwell



Pearson

CompTIA® Linux+™ Portable Command Guide

All the commands for the CompTIA XK0-004 exam in one compact, portable resource

William “Bo” Rothwell

Copyright © 2020 Pearson

Published by:

Pearson IT Certification

221 River Street

Hoboken, NJ 07030 USA

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage and retrieval system, without written permission from the publisher, except for the inclusion of brief quotations in a review.

ScoutAutomatedPrintCode

Library of Congress Control Number: 2019908202

ISBN-13: 978-0-13-559184-0

ISBN-10: 0-13-559184-8

Warning and Disclaimer

This book is designed to provide information about the CompTIA Linux+ exam. 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 is provided on an “as is” basis. The authors and Pearson shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book or from the use of the discs or programs that may accompany it.

The opinions expressed in this book belong to the author and are not necessarily those of Pearson.

Trademark Acknowledgments

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

Editor-in-Chief

Mark Taub

Product Line Manager

Brett Bartow

Executive Editor

Mary Beth Ray

Acquisitions Editor

Paul Carlstroem

Development Editor

Christopher Cleveland

Managing Editor

Sandra Schroeder

Project Editor

Mandie Frank

Copy Editor

Bart Reed

Technical Editor

Casey Boyles

Editorial Assistant

Cindy Teeters

Designer

Chuti Prasertsith

Composition

codeMantra

Indexer

Ken Johnson

Proofreader

Karen Davis

Credits

- Figure 1-2 Screenshot of GNU GRUB © 2010-2018 Free Software Foundation, Inc.
Figure 1-3 Screenshot of GNU GRUB © 2010-2018 Free Software Foundation, Inc.
Figure 3-1 Screenshot of NMTUI interface © 2005–2014 The GNOME Project
Figure 3-2 Screenshot of NMTUI interface © 2005–2014 The GNOME Project
Figure 7-1 Screenshot of Aptitude Utility © The Distro Tracker Developers
Figure 9-1 Screenshot of GNU nano © Chris Allegretta
Figure 9-2 Screenshot of vi Editor © Bill Joy
Figure 21-1 Screenshot of Iftop © Paul Warren
Figure 21-2 Screenshot of Wireshark © Wireshark Foundation
Figure 21-3 Screenshot of mtr © BitWizard
Figure 22-1 Screenshot of Top © William LeFebvre
Cover Pressmaster/Shutterstock

This page intentionally left blank

Contents at a Glance

Introduction xxx

Part I: Hardware and System Configuration

- CHAPTER 1** Explain Linux boot process concepts 1
- CHAPTER 2** Given a scenario, install, configure, and monitor kernel modules 11
- CHAPTER 3** Given a scenario, configure and verify network connection parameters 17
- CHAPTER 4** Given a scenario, manage storage in a Linux environment 37
- CHAPTER 5** Compare and contrast cloud and virtualization concepts and technologies 59
- CHAPTER 6** Given a scenario, configure localization options 69

Part II: Systems Operation and Maintenance

- CHAPTER 7** Given a scenario, conduct software installations, configurations, updates, and removals 77
- CHAPTER 8** Given a scenario, manage users and groups 93
- CHAPTER 9** Given a scenario, create, modify, and redirect files 109
- CHAPTER 10** Given a scenario, manage services 143
- CHAPTER 11** Summarize and explain server roles 157
- CHAPTER 12** Given a scenario, automate and schedule jobs 165
- CHAPTER 13** Explain the use and operation of Linux devices 173
- CHAPTER 14** Compare and contrast Linux graphical user interfaces 187

Part III: Security

- CHAPTER 15** Given a scenario, apply or acquire the appropriate user and/or group permissions and ownership 193
- CHAPTER 16** Given a scenario, configure and implement appropriate access and authentication methods 213
- CHAPTER 17** Summarize security best practices in a Linux environment 235
- CHAPTER 18** Given a scenario, implement logging services 247
- CHAPTER 19** Given a scenario, implement and configure Linux firewalls 257
- CHAPTER 20** Given a scenario, backup, restore, and compress files 267

Part IV: Linux Troubleshooting and Diagnostics

- CHAPTER 21** Given a scenario, analyze system properties and remediate accordingly 277
- CHAPTER 22** Given a scenario, analyze system processes in order to optimize performance 297
- CHAPTER 23** Given a scenario, analyze and troubleshoot user issues 307
- CHAPTER 24** Given a scenario, analyze and troubleshoot application and hardware issues 313

Part V: Automation and Scripting

- CHAPTER 25** Given a scenario, deploy and execute basic BASH scripts 327
- CHAPTER 26** Given a scenario, carry out version control using Git 343
- CHAPTER 27** Summarize orchestration processes and concepts 351
- APPENDIX** Create your own journal 355
 - Index 359

Table of Contents

Introduction xxx

Part I: Hardware and System Configuration

CHAPTER 1 Explain Linux boot process concepts 1

| | |
|------------------------|---|
| Boot Loaders | 1 |
| GRUB | 2 |
| GRUB2 | 2 |
| Boot Options | 3 |
| UEFI/EFI | 4 |
| PXE | 5 |
| NFS | 5 |
| Boot from ISO | 5 |
| Boot from HTTP/FTP | 5 |
| File Locations | 6 |
| /etc/default/grub | 6 |
| /etc/grub2.cfg | 7 |
| /boot | 7 |
| /boot/grub | 7 |
| /boot/grub2 | 7 |
| /boot/efi | 7 |
| Boot Modules and Files | 7 |
| Commands | 8 |
| vmlinuz | 9 |
| vmlinux | 9 |
| Kernel Panic | 9 |

CHAPTER 2 Given a scenario, install, configure, and monitor kernel modules 11

| | |
|----------|----|
| Commands | 11 |
| lsmod | 11 |
| insmod | 12 |
| modprobe | 13 |
| modinfo | 13 |
| dmesg | 14 |
| rmmod | 14 |
| depmod | 15 |

- Files 15
 - /usr/lib/modules/[kernelversion] 15
 - /usr/lib/modules 16
 - /etc/modprobe.conf 16
 - /etc/modprobe.d/ 16

CHAPTER 3 Given a scenario, configure and verify network connection parameters 17

- Diagnostic Tools 18
 - ping 18
 - netstat 18
 - nslookup 19
 - dig 19
 - host 20
 - route 21
 - ip 21
 - ethtool 22
 - ss 24
 - iwconfig 25
 - nmcli 26
 - brctl 27
 - nmtui 27
- Configuration Files 28
 - /etc/sysconfig/network-scripts/ 28
 - /etc/sysconfig/network 29
 - /etc/hosts 29
 - /etc/network 30
 - /etc/nsswitch.conf 30
 - /etc/resolv.conf 31
 - /etc/netplan 31
 - /etc/sysctl.conf 32
 - /etc/dhcpd.conf 32
- Bonding 34
 - Aggregation 34
 - Active/Passive 35
 - Load Balancing 36

| | | |
|----------------------|---|----|
| CHAPTER 4 | Given a scenario, manage storage in a Linux environment | 37 |
| Basic Partitions | | 38 |
| Raw Devices | | 39 |
| GPT | | 39 |
| MBR | | 39 |
| Filesystem Hierarchy | | 40 |
| Real Filesystems | | 40 |
| Virtual Filesystems | | 40 |
| Relative Paths | | 41 |
| Absolute Paths | | 41 |
| Device Mapper | | 41 |
| LVM | | 42 |
| Multipath | | 44 |
| Tools | | 44 |
| XFS Tools | | 44 |
| LVM Tools | | 45 |
| EXT Tools | | 45 |
| Commands | | 45 |
| Location | | 54 |
| /etc/fstab | | 54 |
| /dev/ | | 54 |
| /dev/mapper | | 55 |
| /dev/disk/by- | | 55 |
| /etc/mtab | | 56 |
| /sys/block | | 56 |
| /proc/partitions | | 57 |
| /proc/mounts | | 57 |
| Filesystem Types | | 58 |
| ext3 | | 58 |
| ext4 | | 58 |
| xfs | | 58 |
| nfs | | 58 |
| smb | | 58 |
| cifs | | 58 |
| ntfs | | 58 |

CHAPTER 5 Compare and contrast cloud and virtualization concepts and technologies 59

- Templates 60
 - VM 60
 - OVA 60
 - OVF 61
 - JSON 61
 - YAML 61
 - Container Images 62
- Bootstrapping 62
 - Cloud-init 62
 - Anaconda 63
 - Kickstart 63
- Storage 63
 - Thin vs. Thick Provisioning 63
 - Persistent Volumes 64
 - Blob 64
 - Block 64
- Network Considerations 65
 - Bridging 65
 - Overlay Networks 65
 - NAT 65
 - Local 65
 - Dual-Homed 66
- Types of Hypervisors 66
- Tools 67
 - libvirt 67
 - virsh 67
 - vmm 67

CHAPTER 6 Given a scenario, configure localization options 69

- File Locations 70
 - /etc/timezone 70
 - /usr/share/zoneinfo 70
- Commands 70
 - localectl 70
 - timedatectl 71
 - date 72
 - hwclock 73

| | |
|-----------------------|----|
| Environment Variables | 73 |
| LC_* | 73 |
| LC_ALL | 74 |
| LANG | 74 |
| TZ | 75 |
| Character Sets | 75 |
| UTF-8 | 75 |
| ASCII | 75 |
| Unicode | 75 |

Part II: Systems Operation and Maintenance

CHAPTER 7 Given a scenario, conduct software installations, configurations, updates, and removals 77

| | |
|----------------------|----|
| Package Types | 78 |
| .rpm | 78 |
| .deb | 78 |
| .tar | 78 |
| .tgz | 78 |
| .gz | 78 |
| Installation Tools | 78 |
| RPM | 78 |
| dpkg | 79 |
| APT | 80 |
| YUM | 83 |
| DNF | 86 |
| Zypper | 86 |
| Build Tools | 86 |
| Commands | 86 |
| ldd | 87 |
| Compilers | 88 |
| Shared Libraries | 88 |
| Repositories | 88 |
| Configuration | 88 |
| Creation | 89 |
| Syncing | 89 |
| Locations | 89 |
| Acquisition Commands | 89 |
| wget | 89 |
| curl | 90 |

| | | |
|-----------------------------------|--|-----|
| CHAPTER 8 | Given a scenario, manage users and groups | 93 |
| Creation | 94 | |
| useradd | 94 | |
| groupadd | 94 | |
| Modification | 94 | |
| usermod | 94 | |
| groupmod | 95 | |
| passwd | 95 | |
| chage | 95 | |
| Deletion | 96 | |
| userdel | 96 | |
| groupdel | 96 | |
| Queries | 96 | |
| id | 96 | |
| whoami | 96 | |
| who | 97 | |
| w | 97 | |
| last | 98 | |
| Quotas | 98 | |
| User Quotas | 98 | |
| Group Quotas | 102 | |
| Profiles | 102 | |
| BASH Parameters | 103 | |
| Important Files and File Contents | 106 | |
| /etc/passwd | 106 | |
| /etc/group | 107 | |
| /etc/shadow | 107 | |
| | | |
| CHAPTER 9 | Given a scenario, create, modify, and redirect files | 109 |
| Text Editors | 110 | |
| nano | 110 | |
| vi | 111 | |
| File Readers | 114 | |
| grep | 114 | |
| cat | 115 | |
| tail | 115 | |
| head | 116 | |
| less | 116 | |
| more | 116 | |

| | |
|-------------------------------|-----|
| Output Redirection | 117 |
| < | 117 |
| > | 117 |
| | 118 |
| << | 119 |
| >> | 119 |
| 2> | 119 |
| &> | 119 |
| stdin | 119 |
| stdout | 119 |
| stderr | 119 |
| /dev/null | 119 |
| /dev/tty | 120 |
| xargs | 121 |
| tee | 122 |
| Here Documents | 122 |
| Text Processing | 123 |
| grep | 123 |
| tr | 123 |
| echo | 124 |
| sort | 124 |
| awk | 125 |
| sed | 126 |
| cut | 128 |
| egrep | 128 |
| wc | 128 |
| File and Directory Operations | 129 |
| touch | 129 |
| mv | 129 |
| cp | 129 |
| rm | 130 |
| scp | 130 |
| ls | 131 |
| rsync | 131 |
| mkdir | 132 |
| rmdir | 132 |
| ln | 132 |
| unlink | 135 |
| inodes | 135 |

- find 136
- locate 138
- grep 138
- which 139
- whereis 139
- diff 140
- updatedb 140
- Bonus: regex 141

CHAPTER 10 Given a scenario, manage services 143

- Systemd Management 144
 - Systemctl 145
 - Enabled 146
 - Disabled 146
 - Start 146
 - Stop 146
 - Mask 147
 - Restart 147
 - Status 147
 - Daemon-reload 147
- Systemd-analyze blame 148
- Unit Files 148
 - Directory Locations 149
 - Environment Parameters 150
 - Targets 150
 - Hostnamed 150
 - Automount 151
- SysVinit 152
 - chkconfig 153
 - Runlevels 154
 - Service 156

CHAPTER 11 Summarize and explain server roles 157

- NTP 158
- SSH 158
- Web 159
- Certificate Authority 159
- Name Server 159
- DHCP 160

| | |
|--|-----|
| File Servers | 160 |
| Authentication Server | 161 |
| Proxy | 161 |
| Logging | 162 |
| Containers | 162 |
| VPN | 162 |
| Monitoring | 162 |
| Database | 163 |
| Print Server | 163 |
| Mail Server | 163 |
| Load Balancer | 163 |
| Clustering | 164 |
| CHAPTER 12 Given a scenario, automate and schedule jobs | 165 |
| cron | 165 |
| at | 165 |
| crontab | 167 |
| fg | 170 |
| bg | 171 |
| & | 171 |
| kill | 171 |
| Ctrl-c | 172 |
| Ctrl-z | 172 |
| nohup | 172 |
| CHAPTER 13 Explain the use and operation of Linux devices | 173 |
| Types of Devices | 174 |
| Bluetooth | 174 |
| WiFi | 174 |
| USB | 174 |
| Monitors | 174 |
| GPIO | 174 |
| Network Adapters | 175 |
| PCI | 175 |
| HBA | 175 |
| SATA | 175 |
| SCSI | 175 |
| Printers | 175 |

- Video 175
- Audio 175
- Monitoring and Configuration Tools 176
 - lsdev 176
 - lsusb 176
 - lspci 177
 - lsblk 178
 - dmesg 178
 - lpr 179
 - lpq 179
 - abrt 179
 - CUPS 179
 - udevadm 181
- File Locations 182
 - /proc 182
 - /sys 182
 - /dev 183
 - /dev/mapper 183
 - /etc/X11 184
- Hot Pluggable Devices 185
 - /usr/lib/udev/rules.d (System Rules – Lowest Priority) 185
 - /run/udev/rules.d (Volatile Rules) 185
 - /etc/udev/rules.d (Local Administration – Highest Priority) 186
 - /etc/udev/rules.d 186

CHAPTER 14 Compare and Contrast Linux Graphical User Interfaces 187

- Servers 187
 - Wayland 188
 - X11 188
- GUI 188
 - Gnome 189
 - Unity 189
 - Cinnamon 189
 - MATE 189
 - KDE 189
- Remote Desktop 189
 - VNC 190
 - XRDP 190
 - NX 190
 - Spice 190

| | |
|---------------------|-----|
| Console Redirection | 190 |
| SSH Port Forwarding | 190 |
| Accessibility | 192 |

Part III: Security

CHAPTER 15 Given a scenario, apply or acquire the appropriate user and/or group permissions and ownership 193

| | |
|--------------------------------|-----|
| File and Directory Permissions | 194 |
| Read, Write, Execute | 194 |
| User, Group, Other | 194 |
| SUID | 195 |
| Octal Notation | 195 |
| umask | 195 |
| Sticky Bit | 196 |
| GUID | 196 |
| Inheritance | 196 |
| Utilities | 196 |
| chmod | 197 |
| chown | 198 |
| chgrp | 198 |
| getfacl | 199 |
| setfacl | 199 |
| ls | 201 |
| ulimit | 201 |
| chage | 202 |
| Context-Based Permissions | 203 |
| SELinux Configurations | 203 |
| SELinux Policy | 204 |
| SELinux Tools | 205 |
| AppArmor | 208 |
| Privilege Escalation | 209 |
| su | 209 |
| sudo | 209 |
| wheel | 210 |
| visudo | 210 |
| sudoedit | 211 |
| User Types | 211 |
| Root | 211 |
| Standard | 211 |
| Service | 211 |

CHAPTER 16 Given a scenario, configure and implement appropriate access and authentication methods 213

| | |
|----------------------------------|-----|
| PAM | 214 |
| Password Policies | 215 |
| LDAP Integration | 215 |
| User Lockouts | 216 |
| Required, Allowed, or Sufficient | 217 |
| /etc/pam.d/ | 219 |
| pam_tally2 | 220 |
| faillock | 220 |
| SSH | 221 |
| ~/.ssh/ | 221 |
| User-Specific Access | 223 |
| TCP Wrappers | 224 |
| /etc/ssh/ | 225 |
| ssh-copy-id | 227 |
| ssh-keygen | 227 |
| ssh-add | 228 |
| TTYs | 228 |
| /etc/securetty | 229 |
| /dev/tty# | 229 |
| PTYs | 230 |
| PKI | 230 |
| Self-Signed | 231 |
| Private Keys | 231 |
| Public Keys | 231 |
| Hashing | 231 |
| Digital Signatures | 231 |
| Message Digest | 231 |
| VPN as a Client | 231 |
| SSL/TLS | 232 |
| Transport Mode | 232 |
| Tunnel Mode | 232 |
| IPSec | 232 |
| DTLS | 233 |

CHAPTER 17 Summarize security best practices in a Linux environment 235

| | |
|----------------------|-----|
| Boot Security | 236 |
| Boot Loader Password | 236 |
| UEFI/BIOS Password | 236 |

| | |
|--|-----|
| Additional Authentication Methods | 237 |
| Multifactor Authentication | 237 |
| RADIUS | 238 |
| TACACS+ | 238 |
| LDAP | 238 |
| Kerberos | 238 |
| Importance of Disabling Root Login via SSH | 239 |
| Password-Less Login | 239 |
| Enforce Use of PKI | 240 |
| Chroot Jail Services | 240 |
| No Shared IDs | 240 |
| Importance of Denying Hosts | 240 |
| Separation of OS Data from Application Data | 241 |
| Disk Partition to Maximize System Availability | 241 |
| Change Default Ports | 241 |
| Importance of Disabling or Uninstalling Unused and Unsecure Services | 242 |
| FTP | 242 |
| Telnet | 242 |
| Finger | 242 |
| Sendmail | 242 |
| Postfix | 243 |
| Importance of Enabling SSL/TLS | 243 |
| Importance of Enabling auditd | 243 |
| CVE Monitoring | 243 |
| Discouraging Use of USB Devices | 243 |
| Disk Encryption | 244 |
| LUKS | 244 |
| Restrict cron Access | 244 |
| Disable Ctrl-Alt-Del | 244 |
| Add Banner | 245 |
| MOTD | 245 |
| CHAPTER 18 Given a scenario, implement logging services | 247 |
| Key File Locations | 247 |
| /var/log/secure | 247 |
| /var/log/messages | 248 |
| /var/log/[application] | 248 |
| /var/log/kern.log | 249 |

- Log Management 249
 - Third-Party Agents 249
 - logrotate 250
 - /etc/rsyslog.conf 252
 - journald 253
 - journalctl 253
 - lastb 255

CHAPTER 19 Given a scenario, implement and configure Linux firewalls 257

- Access Control Lists 258
 - Source 258
 - Destination 258
 - Ports 258
 - Protocol 258
 - Logging 258
 - Stateful vs. Stateless 258
 - Accept 259
 - Reject 259
 - Drop 259
 - Log 259
- Technologies 259
 - firewalld 259
 - iptables 260
 - ufw 262
 - Netfilter 262
- IP Forwarding 263
- Dynamic Rule Sets 263
 - DenyHosts 263
 - Fail2ban 263
 - IPset 265
- Common Application 265
 - Firewall Configurations 265

CHAPTER 20 Given a scenario, backup, restore, and compress files 267

- Archive and Restore Utilities 268
 - tar 268
 - cpio 268
 - dd 269

| | |
|-----------------------------|-----|
| Compression | 269 |
| gzip | 269 |
| xz | 270 |
| bzip2 | 271 |
| zip | 271 |
| Backup Types | 272 |
| Incremental | 272 |
| Full | 272 |
| Snapshot Clones | 273 |
| Differential | 273 |
| Image | 273 |
| Off-Site/Off-System Storage | 274 |
| SFTP | 274 |
| SCP | 274 |
| rsync | 275 |
| Integrity Checks | 275 |
| MD5 | 275 |
| SHA | 275 |

Part IV: Linux Troubleshooting and Diagnostics

CHAPTER 21 Given a scenario, analyze system properties and remediate accordingly 277

| | |
|--------------------------------------|-----|
| Network Monitoring and Configuration | 278 |
| Latency | 279 |
| Routing | 279 |
| Saturation | 279 |
| Packet Drop | 279 |
| Timeouts | 279 |
| Name Resolution | 280 |
| Localhost vs. Unix Socket | 280 |
| Interface Configurations | 280 |
| Commands | 280 |
| Storage Monitoring and Configuration | 287 |
| iostat | 288 |
| ioping | 288 |
| IO Scheduling | 288 |
| du | 289 |
| df | 289 |
| LVM Tools | 289 |

- fsck 289
- partprobe 289
- CPU Monitoring and Configuration 289
 - /proc/cpuinfo 289
 - uptime 291
 - loadaverage 291
 - sar 291
 - sysctl 292
- Memory Monitoring and Configuration 292
 - swapon 292
 - swapoff 292
 - mkswap 293
 - vmstat 293
 - Out of Memory Killer 294
 - free 294
 - /proc/meminfo 295
 - Buffer Cache Output 295
- Lost Root Password 295
 - Single User Mode 295

CHAPTER 22 Given a scenario, analyze system processes in order to optimize performance 297

- Process Management 297
- Process States 297
 - Zombie 298
 - Uninterruptible Sleep 298
 - Interruptible Sleep 298
 - Running 298
- Priorities 298
- Kill Signals 299
- Commands 300
 - nice 300
 - renice 300
 - top 301
 - time 302
 - ps 303
 - lsof 303
 - pgrep 304
 - pkill 305
- PIDs 305

CHAPTER 23 Given a scenario, analyze and troubleshoot user issues 307

- Permissions 307
 - File 308
 - Directory 308
- Access 309
 - Local 309
 - Remote 309
- Authentication 309
 - Local 310
 - External 310
 - Policy Violations 310
- File Creation 310
 - Quotas 310
 - Storage 311
 - Inode Exhaustion 311
 - Immutable Files 311
- Insufficient Privileges for Authorization 312
- Environment and Shell Issues 312

CHAPTER 24 Given a scenario, analyze and troubleshoot application and hardware issues 313

- SELinux Context Violations 314
- Storage 314
 - Degraded Storage 314
 - Missing Devices 315
 - Missing Mount Point 315
 - Performance Issues 315
 - Resource Exhaustion 316
 - Adapters 316
 - Storage Integrity 317
- Firewall 317
 - Restrictive ACLs 318
 - Blocked Ports 318
 - Blocked Protocols 318
- Permission 318
 - Ownership 319
 - Executables 319
 - Inheritance 319

- Service Accounts 319
- Group Memberships 319
- Dependencies 319
 - Patching 319
 - Update Issues 320
 - Versioning 320
 - Libraries 320
 - Repositories 320
- Troubleshooting Additional Hardware Issues 320
 - Memory 320
 - Printers 321
 - Video 321
 - GPU Drivers 321
 - Communications Ports 321
 - USB 322
 - Keyboard Mapping 324
 - Hardware or Software Compatibility Issues 324
 - Commands 324

Part V: Automation and Scripting

CHAPTER 25 Given a scenario, deploy and execute basic BASH scripts 327

- Shell Environments and Shell Variables 328
 - PATH 329
 - Global 329
 - Local 329
 - export 329
 - env 330
 - set 330
 - printenv 331
 - echo 331
- #!/ bin/bash 332
- Sourcing Scripts 332
- Directory and File Permissions 333
 - chmod 333
- Extensions 333
- Commenting 333
 - # 333

| | |
|---|-----|
| File Globbing | 334 |
| Shell Expansions | 334 |
| \${} | 335 |
| \$() | 335 |
| ` ` | 335 |
| Redirection and Piping | 336 |
| Exit Codes | 336 |
| stderr | 336 |
| stdin | 336 |
| stdout | 336 |
| Metacharacters | 336 |
| Positional Parameters | 337 |
| Looping Constructs | 337 |
| while | 337 |
| for | 337 |
| until | 338 |
| Conditional Statements | 338 |
| if | 339 |
| case | 340 |
| Escaping Characters | 340 |
| CHAPTER 26 Given a scenario, carry out version control using Git | 343 |
| Arguments | 343 |
| clone | 343 |
| push | 344 |
| pull | 344 |
| commit | 345 |
| merge | 345 |
| branch | 347 |
| log | 348 |
| init | 348 |
| config | 349 |
| Files | 349 |
| .gitignore | 349 |
| .git/ | 350 |

CHAPTER 27 Summarize orchestration processes and concepts 351

Agent 351

Agentless 352

Procedures 352

Attributes 352

Infrastructure Automation 352

Infrastructure as Code 352

Inventory 353

Automated Configuration Management 353

Build Automation 353

APPENDIX Create your own journal 355

Index 359

About the Author

At the impressionable age of 14, **William “Bo” Rothwell** crossed paths with a TRS-80 Micro Computer System (affectionately known as a “Trash 80”). Soon after the adults responsible for Bo made the mistake of leaving him alone with the TSR-80, he immediately dismantled it and held his first computer class, showing his friends what made this “computer thing” work.

Since this experience, Bo’s passion for understanding how computers work and sharing this knowledge with others has resulted in a rewarding career in IT training. His experience includes Linux, Unix, IT Security, DevOps, and programming languages such as Perl, Python, Tcl, and BASH. He is the founder and lead instructor of One Course Source, an IT training organization.

About the Technical Reviewer

Casey Boyles started working in the IT field more than 27 years ago and quickly moved on to distributed application and database development. Casey later moved on to technical training and course development, where 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.”

Dedications

As always, I owe more gratitude than I could ever provide to those in my life who support me the most: Sarah, Julia, Mom, and Dad.

Acknowledgments

To everyone at Pearson who helped make this book come to life, I thank you. I know that this is a team effort, and I appreciate everyone's hard work.

Special thanks go to Mary Beth Ray. I'm very sad that this will be our last book together. I have enjoyed working with you the last few years. Good fortune in your new journey in life. :)

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@informit.com

Introduction

In 2019, COMPTIA released a new version of the Linux+ certification exam. This new exam isn't just an update from the last certification, but really a completely new exam.

The previous Linux+ certification was titled “CompTIA Linux+ powered by LPI.” However, the new certification no longer involves LPI (the Linux Professional Institute). As a result, the format and topics are significantly different.

For example, the new certification only requires passing one exam, not two. Additionally, the exam objectives have undergone a major rewrite, and you will see several DevOps-based topics not included in the previous certification.

Additionally, you should be prepared for a handful of scenario questions that you will be asked to answer based on a situation described to you. However, most of the exam will be multiple choice, much like the previous exams.

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

Good luck!

—William “Bo” Rothwell
May 18, 2019

Who Should Read This Book

This book is for those people preparing for the CompTIA Linux+ certification exam, whether through self-study, on-the-job training and practice, or study via a training program. The book provides you with the depth of knowledge to help you pass the exam as well as introduces valuable features of the Linux operating system.

Command Syntax Conventions

The conventions used to present command syntax in this book are as follows:

- **Boldface** indicates commands and keywords that are entered literally as shown. In actual configuration examples and output (not general command syntax), boldface indicates commands that are manually input by the user (such as a **show** command).
- *Italic* indicates arguments for which you supply actual values.
- Vertical bars (|) separate alternative, mutually exclusive elements.
- Square brackets ([]) indicate an optional element.
- Braces ({ }) indicate a required choice.
- Braces within brackets ([{ }]) indicate a required choice within an optional element.

Organization of This Book

Because this book is designed to help you prepare for the CompTIA Linux+ certification exam, I have opted to match the organization of the book to align with the exam objective topics. There are 27 topics for the exam, and those topics match with each of the 27 chapters in this book. This organization should help aid you in your preparation for the exam.

Here is the layout for each section of this book:

Part I: Hardware and System Configuration

- **Chapter 1, “Explain Linux boot process concepts”**—You will learn the process of how the system behaves during the boot process and how the administrator can modify this behavior in this chapter.
- **Chapter 2, “Given a scenario, install, configure, and monitor kernel modules”**—In this chapter, you will explore kernel modules, which are small programs that add to the functionality of the kernel.
- **Chapter 3, “Given a scenario, configure and verify network connection parameters”**—You will learn how to set up network interfaces in this chapter.
- **Chapter 4, “Given a scenario, manage storage in a Linux environment”**—In this chapter, you learn how to create partitions and filesystems using utilities like fdisk and mkfs.
- **Chapter 5, “Compare and contrast cloud and virtualization concepts and technologies”**—The focus of this chapter is the core concepts of virtual OS instances, both on a local system and in the cloud.
- **Chapter 6, “Given a scenario, configure localization options”**—In this chapter, you will discover how to modify the system to behave differently in different locations throughout the world.

Part II: Systems Operation and Maintenance

- **Chapter 7, “Given a scenario, conduct software installations, configurations, updates, and removals”**—You will learn in this chapter how to manage software packages.
- **Chapter 8, “Given a scenario, manage users and groups”**—This chapter focuses on utilities that allow you to add, modify, and delete user and group accounts.
- **Chapter 9, “Given a scenario, create, modify, and redirect files”**—Learn how to create files using commonly used Linux editors as well as sending the output of commands into files.
- **Chapter 10, “Given a scenario, manage services”**—The focus of this chapter is services, which are programs that run on the local system and provide some sort of feature or function to either the local system or remote systems.
- **Chapter 11, “Summarize and explain server roles”**—The understanding of servers and the role they play on Linux systems are explored in this chapter.

- **Chapter 12, “Given a scenario, automate and schedule jobs”**—Learn the process of scheduling programs and applications to run at future times.
- **Chapter 13, “Explain the use and operation of Linux devices”**—Learn how to manage devices, such as USB devices, memory, and physical storage devices.
- **Chapter 14, “Compare and contrast Linux graphical user interfaces”**—Explore the wide variety of graphical user interfaces (GUIs) in this chapter.

Part III: Security

- **Chapter 15, “Given a scenario, apply or acquire the appropriate user and/or group permissions and ownership”**—Building on the concepts you learned in Chapter 8, this chapter will focus on how to manage permissions that provide access to files and directories.
- **Chapter 16, “Given a scenario, configure and implement appropriate access and authentication methods”**—This chapter explores the different authentication methods, a feature that allows users to gain access to the system.
- **Chapter 17, “Summarize security best practices in a Linux environment”**—This chapter provides a high-level overview of important security best practices to implement on Linux systems.
- **Chapter 18, “Given a scenario, implement logging services”**—Learn how system logging is configured in this chapter.
- **Chapter 19, “Given a scenario, implement and configure Linux firewalls”**—The focus of this chapter is the concept and configuration of firewalls on Linux systems.
- **Chapter 20, “Given a scenario, backup, restore, and compress files”**—Protect against data loss by learning how to create backups in this chapter.

Part IV: Linux Troubleshooting and Diagnostics

- **Chapter 21, “Given a scenario, analyze system properties and remediate accordingly”**—This chapter explores the essentials of troubleshooting common problems on Linux systems.
- **Chapter 22, “Given a scenario, analyze system processes in order to optimize performance”**—The focus of this chapter is tools and concepts related to determining if a Linux system is performing at an optimal level.
- **Chapter 23, “Given a scenario, analyze and troubleshoot user issues”**—Learn how to troubleshoot common problems related to user accounts in this chapter.
- **Chapter 24, “Given a scenario, analyze and troubleshoot application and hardware issues”**—This chapter provides you with the concepts and tools needed to discover and resolve issues related to system hardware and applications.

Part V: Automation and Scripting

- **Chapter 25, “Given a scenario, deploy and execute basic BASH scripts”**—Shell scripts, a collection of shell commands, are covered in this chapter.
- **Chapter 26, “Given a scenario, carry out version control using Git”**—In this chapter, you will learn how to manage a Git repository.
- **Chapter 27, “Summarize orchestration processes and concepts”**—In this chapter, you will explore orchestration, which includes the process of quickly and effectively installing and configuring systems.

This page intentionally left blank

Given a scenario, apply or acquire the appropriate user and/or group permissions and ownership

This chapter provides information and commands concerning the following topics:

File and directory permissions

- Read, write, execute
- User, group, other
- SUID
- Octal notation
- umask
- Sticky bit
- GUID
- Inheritance
- Utilities (chmod, chown, chgrp, getfacl, setfacl, ls, ulimit, chage)

Context-based permissions

- SELinux configurations (disabled, permissive, enforcing)
- SELinux policy (targeted)
- SELinux tools (setenforce, getenforce, sestatus, setsebool, getsebool, chcon, restorecon, ls -Z, ps -Z)
- AppArmor (aa-disable, aa-complain, aa-unconfined, /etc/apparmor.d/, /etc/apparmor.d/tunables)

Privilege escalation

- su
- sudo
- wheel
- visudo
- sudoedit

User types

- Root
- Standard
- Service

File and Directory Permissions

The user who owns a file or directory has the ability to allow or deny access to the file or directory using permissions. Additionally, the root user has the ability to change the permissions of any file or directory on the system. This section focuses on these permissions and how to apply them.

Read, Write, Execute

Every file and directory has standard permissions (also called “read, write, and execute” permissions) that either allow or disallow a user access. Using these standard permissions is something that every Linux user should understand how to do, as this is the primary way a user will protect his files from other users.

To view the permissions of a file or directory, use the **ls -l** command:

```
[student@OCS ~]$ ls -l /etc/chrony.keys
-rw-r-----. 1 root chrony 62 May  9 2018 /etc/chrony.keys
```

The first ten characters of the output denote the file type (recall that a hyphen [-] as the character in the first position denotes a plain file, whereas a **d** denotes a directory) and the permissions for the file. Permissions are broken into three sets: the user owner of the file (root in the previous example), the group owner (chrony), and all other users (referred to as “others”).

Each set has three possible permissions: read (symbolized by **r**), write (**w**), and execute (**x**). If the permission is set, the character that symbolizes the permission is displayed. Otherwise, a hyphen (-) character is displayed to indicate that permission is not set. Thus, **r-x** means “read and execute are set, but write is not set.”

What read, write, and execute permissions really mean depends on whether the object is a file or directory. For files, these permissions mean the following:

- **Read:** Can view or copy file contents.
- **Write:** Can modify file contents.
- **Execute:** Can run the file like a program. After you create a program, you must make it executable before you can run it.

For directories, they mean the following:

- **Read:** Can list files in the directory.
- **Write:** Can add and delete files in the directory (requires execute permission).
- **Execute:** Can “**cd**” into the directory or use it in a pathname.

User, Group, Other

See the “Read, Write, Execute” subsection in this chapter.

SUID

The following table describes the special permission sets of **suid**, **sgid**, and the sticky bit:

| | suid | sgid | Sticky Bit |
|--------------------|--|---|--|
| Description | When set on executable files, suid allows a program to access files using the permissions of the user owner of the executable file. | When set on executable files, sgid allows a program to access files using the permissions of the group owner of the executable file. When it's set on directories, all new files in the directory inherit the group ownership of the directory. | When the sticky bit is set on directories, files in the directory can only be removed by the user owner of the file, the owner of the directory, or the root user. |
| Set | chmod u+s file or chmod 4xxx file (xxx refers to regular read, write, and execute permissions.) | chmod g+s file or chmod 2xxx file (xxx refers to regular read, write, and execute permissions.) | chmod o+t file or chmod 1xxx file (xxx refers to regular read, write, and execute permissions). Note: Sticky bit permissions are almost always set to the octal value of 1777. |
| Remove | chmod u-s file or chmod 0xxx file | chmod g-s file or chmod 0xxx file | chmod o-t file or chmod 0xxx file |

See the “chmod” subsection in this chapter for details about the **chmod** command.

Octal Notation

See the “chmod” subsection in this chapter for details about octal notation.

umask

The **umask** command sets default permissions for files and directories. These default permissions are applied only when the file or directory is initially created.

The **umask** command accepts one argument: the mask value. The mask value is an octal value that is applied against the maximum possible permissions for new files or new directories, as shown in the following table:

| Type | Maximum Possible Permission for New Item |
|-----------|--|
| File | rw-rw-rw- |
| Directory | rw-rw-rwX |

Figure 15.1 describes how a umask value of **027** would apply to new files versus how it would apply to new directories.

| Description | File | | | Directories | | |
|---------------|------|-----|-----|-------------|-----|-----|
| Maximum | rw- | rw- | rw- | rwX | rwX | rwX |
| Umask Applied | --- | -M- | MM- | --- | -M- | MM- |
| Result | rw- | r-- | --- | rwX | r-X | --X |

Figure 15-1 How umask Is Applied

NOTE: Each shell has its own umask value. If you change the umask in one shell, it will not affect the umask value in any other shell. To make a persistent change to your umask across logins, add the **umask** command to the `~/.bash_profile` file.

Sticky Bit

See the “SUID” subsection in this chapter for details about the sticky bit.

GUID

See the “SUID” subsection in this chapter for details about GUID.

Inheritance

Unlike in some operating systems, basic and advanced Linux permissions don’t utilize inheritance. The idea of inheritance is when a new file or directory inherits the permissions from the directory that the item is created in.

There is a way to have permissions inherit from the parent directory: by using ACLs (access control lists). See the “setfacl” subsection for details.

Utilities

Several utilities or commands allow you to manage permissions. The utilities that are testable for the Linux+ exam are covered in this section.

chmod

The **chmod** command is used to change permissions on files. It can be used in two ways: symbolic method and octal method. With the octal method, the permissions are assigned numeric values:

- Read = 4
- Write = 2
- Execute = 1

With these numeric values, one number can be used to describe an entire permission set:

- 7 = **rwX**
- 6 = **rw-**
- 5 = **r-X**
- 4 = **r--**
- 3 = **-wX**
- 2 = **-w-**
- 1 = **--X**
- 0 = **---**

So, to change the permissions of a file to **rwXr-X-**, you would execute the following command:

```
chmod 754 filename
```

The following table demonstrates some examples using the octal method:

| Example | Description |
|-----------------------|--|
| chmod 755 file | Sets the permissions of rwXr-X- . |
| chmod 511 file | Sets the permissions of r-X--X--X . |
| chmod 600 file | Sets the permissions of rw----- . |

With octal permissions, you should always provide three numbers, which will change all the permissions. But, what if you only want to change a single permission of the set? For that, use the symbolic method by passing three values to the **chmod** command, as shown in the following table:

| Who | What | Permission |
|-----------------|------|------------|
| u = user owner | + | r |
| g = group owner | - | w |
| o = other | = | x |
| a = all sets | | |

The following demonstrates adding execute permission to all three sets (user owner, group owner, and others) using the symbolic method:

```
[student@OCS ~]$ ls -l display.sh
-rw-rw-r--. 1 student student 291 Apr 30 20:09 display.sh
[student@OCS ~]$ chmod a+x display.sh
[student@OCS ~]$ ls -l display.sh
-rwxrwxr-x. 1 student student 291 Apr 30 20:09 display.sh
```

Here are some important options for the **chmod** command:

| Option | Description |
|-----------|--|
| -R | Recursively apply changes to an entire directory structure. |
| -v | Verbose. Produce output demonstrating the changes that are made. |

chown

The **chown** command is used to change the user owner or group owner of a file or directory. The following table demonstrates different ways to use this command:

| Example | Description |
|--------------------------------|---|
| chown tim abc.txt | Changes the user owner of the abc.txt file to tim user. |
| chown tim:staff abc.txt | Changes the user owner of the abc.txt file to tim user and the group owner to the staff group. |
| chown :staff abc.txt | Changes the group owner of the abc.txt file to the staff group. |

NOTE: Only the root user can change the user owner of a file. To change the group owner of a file, the user who executes the command must own the file and be a member of the group that the ownership is being changed to.

Here are some important options for the **chown** command:

| Option | Description |
|--------------------------------|---|
| -R | Recursively apply changes to an entire directory structure. |
| --reference=<i>file</i> | Change the user and group owner to the ownership of <i>file</i> . |
| -v | Verbose. Produce output demonstrating the changes that are made. |

chgrp

The **chgrp** command is designed to change the group ownership of a file. The syntax of this command is **chgrp** [*options*] *group_name file*. In the following example, the group ownership of the **abc.txt** file is changed to the staff group:

```
[student@OCS ~]$ chgrp staff abc.txt
```

NOTE: To change the group owner of a file, the user who executes the command must own the file and be a member of the group that the ownership is being changed to.

Here are some important options for the **chgrp** command:

| Option | Description |
|-------------------------|---|
| -R | Recursively apply changes to an entire directory structure. |
| --reference=file | Change the user and group owner to the ownership of <i>file</i> . |
| -v | Verbose. Produce output demonstrating the changes that are made. |

getfacl

See the “setfacl” subsection next.

setfacl

ACLs (access control lists) allow the owner of a file to give permissions for specific users and groups. The **setfacl** command is used to create an ACL on a file or directory:

```
sarah@OCS:~$ setfacl -m user:dane:r-- sales_report
```

The **-m** option is used to make a new ACL for the file. The format of the argument to the **-m** option is *what:who:permission*. The value for *what* can be one of the following:

- **user** or **u** when applying an ACL for a specific user.
- **group** or **g** when applying an ACL for a specific group.
- **others** or **o** when applying an ACL for “others.”
- **mask** or **m** when setting the mask for the ACL. (The mask will be explained later in this section.)

The value for *who* will be the user or group to which the permission will be applied. The permission can be provided as either a symbolic value (**r--**) or an octal value (**4**).

Once an ACL has been applied on a file or directory, a plus sign (+) character will appear after the permissions when the **ls -l** command is executed, as shown here:

```
sarah@OCS:~$ ls -l sales_report
-rw-rw-r--+ 1 sarah sales 98970 Dec 27 16:45 sales_report
```

To view the ACL, use the **getfacl** command:

```
sarah@OCS:~$ getfacl sales_report
# file: sales_report
# owner: sarah
# group: sarah
user::rw-
user:william:r--
group::rw-
mask::rw-
other::r--
```

The following example demonstrates setting an ACL for a group:

```
sarah@OCS:~$ setfacl -m g:games:6 sales_report
sarah@OCS:~$ getfacl sales_report
# file: sales_report
# owner: sarah
# group: sarah
user::rw-
user:william:r--
group::rw-
group:games:rw-
mask::rw-
other::r--
```

For regular permissions, the **umask** value is used to determine the default permissions applied for new files and directories. For ACLs, you can define a default ACL set for all new files and directories that are created within a shell by using the **-m** option with the **setfacl** command. In this case, the following syntax is used for the argument: **default:what:who:permission**.

The following example will create a default ACL for the **reports** directory:

```
sarah@OCS:~$ mkdir reports
sarah@OCS:~$ setfacl -m default:g:games:r-x reports
sarah@OCS:~$ setfacl -m default:u:bin:rwx reports
sarah@OCS:~$ getfacl reports
# file: reports
# owner: sarah
# group: sarah
user::rwx
group::rwx
other::r-x
default:user::rwx
default:user:bin:rwx
default:group::rwx
default:group:games:r-x
default:mask::rwx
default:other::r-x
```

The following example demonstrates how new files and directories will inherit the ACLs that were created in the commands executed in the previous example:

```
sarah@OCS:~$ mkdir reports/test
sarah@OCS:~$ getfacl reports/test
# file: reports/test
# owner: sarah
# group: sarah
```

```

user::rwx
user:bin:rwx
group::rwx
group:games:r-x
mask::rwx
other::r-x
default:user::rwx
default:user:bin:rwx
default:group::rwx
default:group:games:r-x
default:mask::rwx
default:other::r-x
sarah@OCS:~$ touch reports/sample1
sarah@OCS:~$ getfacl reports/sample1
# file: reports/sample1
# owner: sarah
# group: sarah
user::rw-
user:bin:rwx                #effective:rw-
group::rwx                  #effective:rw-
group:games:r-x            #effective:r--
mask::rw-
other::r--

```

Here are some important options for the **setfacl** command:

| Option | Description |
|-----------|--|
| -b | Remove all ACLs. |
| -d | Set a default ACL on a directory; this will be inherited by any new file or directory created with this directory. |
| -R | Apply recursively. |

ls

See the “Read, Write, Execute” subsection in this chapter to see how the **ls** command is important for displaying permissions.

ulimit

The **ulimit** command lists or sets a user’s account limits:

```

[root@OCS ~]# ulimit -a
core file size          (blocks, -c)  0
data seg size           (kbytes, -d)  unlimited
scheduling priority     (-e)          0
file size               (blocks, -f)  unlimited

```

```

pending signals          (-i)    15439
max locked memory        (kbytes, -l)  64
max memory size          (kbytes, -m)  unlimited
open files               (-n)     1024
pipe size                 (512 bytes, -p)  8
POSIX message queues     (bytes, -q)  819200
real-time priority       (-r)     0
stack size               (kbytes, -s)  8192
cpu time                  (seconds, -t) unlimited
max user processes       (-u)    4096
virtual memory           (kbytes, -v)  unlimited
file locks                (-x)  unlimited

```

These limits are normally configured by the system administrator using a PAM (Pluggable Authentication Modules) configuration file:

```

[root@OCS ~]# tail -n 12 /etc/security/limits.conf
#<domain>          <type>          <item>          <value>
#
#*                  soft           core            0
#*                  hard           rss             10000
#@student          hard           nproc           20
#@faculty          soft           nproc           20
#@faculty          hard           nproc           50
#ftp               hard           nproc           0
# End of file

```

For example, you may want to limit how many concurrent logins an account can have:

```

student            -                maxlogins       4

```

Users rarely use the **ulimit** command to limit their own account, so the options for this command are not as important as understanding what the output displays. Additionally, some of the limits are very rarely used. The commonly used limits are described in the following table:

| Limit | Description |
|------------------|--|
| fsize | Maximum file size allowed in memory |
| cpu | Maximum CPU time allowed |
| nproc | Maximum number of concurrently running processes |
| maxlogins | Maximum number of concurrent logins |

chage

See the “chage” section in Chapter 8, “Given a scenario, manage users and groups.”

Context-Based Permissions

Files and directories may be compromised by users who either do not understand permissions or accidentally provide more access than intended. This is a reflection of an old system administration saying, “If we didn’t have users, nothing would break and the system would be more secure.” Of course, the response to this saying is, “Without users, we wouldn’t have a job!” Users’ mistakes often do provide unintended access to the data that is stored in files.

Note that traditional Linux permissions make use of Discretionary Access Control (DAC), while context-based permissions utilize Mandatory Access Control (MAC). However, when a context-based solution is enabled, DAC still applies (both MAC and DAC are enforced).

Context-based permissions can be configured to accommodate for this flaw by providing an additional level of security when processes (programs) are used to access files. This section covers two commonly used context-based methods: SELinux and AppArmor.

SELinux Configurations

An SELinux security policy can be applied that will require processes to be a part of an SELinux security context (think “security group”) in order to be able to access files and directories. Regular permissions will still be used to further define access, but for accessing the file/directory, this SELinux policy would be applied first.

A bigger concern, and one that most SELinux policies are designed to address, is how daemon (or system) processes present a security risk. Consider a situation where you have many active processes that provide a variety of services. For example, one of these processes may be a web server, as shown in the following example:

```
root@OCS:~# ps -fe | grep httpd
root      1109      1  0  2018 ?        00:51:56 /usr/sbin/httpd
apache    1412    1109  0 Dec24 ?        00:00:09 /usr/sbin/httpd
apache    4085    1109  0 05:40 ?        00:00:12 /usr/sbin/httpd
apache    8868    1109  0 08:41 ?        00:00:06 /usr/sbin/httpd
apache    9263    1109  0 08:57 ?        00:00:04 /usr/sbin/httpd
apache   12388    1109  0 Dec26 ?        00:00:47 /usr/sbin/httpd
apache   18707    1109  0 14:41 ?        00:00:00 /usr/sbin/httpd
apache   18708    1109  0 14:41 ?        00:00:00 /usr/sbin/httpd
apache   19769    1109  0 Dec27 ?        00:00:15 /usr/sbin/httpd
apache   29802    1109  0 01:43 ?        00:00:17 /usr/sbin/httpd
apache   29811    1109  0 01:43 ?        00:00:11 /usr/sbin/httpd
apache   29898    1109  0 01:44 ?        00:00:10 /usr/sbin/httpd
```

Note that in the preceding output, each line describes one Apache Web Server process (**/usr/sbin/httpd**) that is running on the system. The first part of the line is the user who initiated the process. The process that runs as root is only used to spawn additional **/usr/sbin/httpd** processes. The others, however, respond to incoming web page requests from client utilities (web browsers).

Imagine for a moment that a security flaw is discovered in the software for the Apache Web Server that allows a client utility to gain control of one of the `/usr/sbin/httpd` processes and issue custom commands or operations to that process. One of those operations could be to view the content of the `/etc/passwd` file, which would be successful because of the permissions placed on this file:

```
root@OCS:~# ls -l /etc/passwd
-rw-r--r-- 1 root root 2690 Dec 11 2018 /etc/passwd
```

As you can see from the output of the preceding command, all users have the ability to view the contents of the `/etc/passwd` file. Ask yourself this: Do you want some random person (usually called a hacker) to have the ability to view the contents of the file that stores user account data?

With an SELinux policy, the `/usr/sbin/httpd` processes can be “locked down” so each can only access a certain set of files. This is what most administrators use SELinux for: to secure processes that may be compromised by hackers making use of known (or, perhaps, unknown) exploits.

This subsection covers the essentials of managing an SELinux security policy.

disabled

When in disabled mode, SELinux is not functional at all. No checks are performed when users attempt to access files or directories. See the “setenforce” and “getenforce” subsections in this chapter for more details on viewing and changing the SELinux mode.

permissive

When in permissive mode, SELinux performs checks but will never block access to a file or directory. This mode is designed for troubleshooting problems as log messages are created when in this mode. See the “setenforce” and “getenforce” subsections in this chapter for more details on viewing and changing the SELinux mode.

enforcing

When in enforcing mode, SELinux performs checks and will block access to files or directories if necessary. See the “setenforce” and “getenforce” subsections in this chapter for more details on viewing and changing the SELinux mode.

SELinux Policy

An SELinux policy is a collection of rules that determine what restrictions are imposed by the policy. The policy itself is often very complex, and details are beyond the scope of the Linux+ exam. It is, however, important to know that the policy sets the restrictions based on rules.

You should also know that one of the most commonly used policies is the “targeted” policy. This policy normally exists by default on systems that have SELinux installed, and it is typically the default policy that is enabled when SELinux is first enabled.

A targeted policy contains rules designed to protect the system from services, rather than regular users. Each service is assigned one or more security contexts, Boolean values, and additional rules that limit the service's ability to access files and directories.

targeted

See the “SELinux Policy” subsection in this chapter.

SELinux Tools

A large number of tools are used to manage SELinux. This subsection covers the tools you should know for the Linux+ exam.

setenforce

You can disable the security policy (useful when testing a new policy or troubleshooting SELinux problems) with the **setenforce** command:

```
root@OCS:~# setenforce 0
root@OCS:~# getenforce
Permissive
```

While in “Permissive” mode, SELinux will not block any access to files and directories, but warnings will be issued and viewable in the system log files.

getenforce

Use the **getenforce** command to determine the current SELinux mode:

```
root@OCS:~# getenforce
Enforcing
```

The result “Enforcing” means SELinux is installed and the security policy is currently active. See the “disabled,” “permissive,” and “enforcing” subsections in this chapter for more details regarding SELinux modes.

sestatus

The **sestatus** command provides overall status information about SELinux:

```
root@OCS:~# sestatus
SELinux status:                enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:              targeted
Current mode:                    enforcing
Mode from config file:           enforcing
Policy MLS status:               enabled
Policy deny_unknown status:      allowed
Max kernel policy version:       28
```

setsebool

To set an SELinux Boolean, use the **setsebool** command:

```
root@OCS:~# getsebool -a | grep abrt_anon_write
abrt_anon_write --> off
root@OCS:~# setsebool abrt_anon_write 1
root@OCS:~# getsebool -a | grep abrt_anon_write
abrt_anon_write --> on
```

See the “getsebool” subsection next for information about Boolean values.

getsebool

Part of an SELinux security policy includes Booleans. A Boolean is a setting that can be assigned either a true or a false value. This value can affect the behavior of the SELinux policy.

```
root@OCS:~# getsebool -a | head
abrt_anon_write --> off
abrt_handle_event --> off
abrt_upload_watch_anon_write --> on
antivirus_can_scan_system --> off
antivirus_use_jit --> off
auditadm_exec_content --> on
authlogin_nsswitch_use_ldap --> off
authlogin_radius --> off
authlogin_yubikey --> off
awstats_purge_apache_log_files --> off
```

In order to determine what a Boolean is used for, use the **semanage** command:

```
root@OCS:~# semanage boolean -l | head
SELinux boolean                State  Default Description

privoxy_connect_any            (on   ,   on)
  Allow privoxy to connect any
smartmon_3ware                 (off  ,  off) Allow smartmon
  to 3ware
mpd_enable_homedirs            (off  ,  off) Allow mpd to
  enable homedirs
xdm_sysadm_login               (off  ,  off) Allow xdm to
  sysadm login
xen_use_nfs                     (off  ,  off) Allow xen to use nfs
mozilla_read_content           (off  ,  off) Allow mozilla
  to read content
ssh_chroot_rw_homedirs         (off  ,  off) Allow ssh to
  chroot rw homedirs
mount_anyfile                  (on   ,   on) Allow mount to anyfile
```

See the “setsebool” subsection in this chapter for information on how to set a Boolean value.

chcon

Use the **chcon** command to change the context of a file or directory:

```
root@OCS:~# chcon -t user_home_t /var/www/html/index.html
```

See the “ls -Z” subsection in this chapter for more details regarding security contexts.

restorecon

There are SELinux rules that define the default security contexts for a majority of the system files. The **restorecon** command is used to reset the default security context on a file or directory.

Example:

```
root@OCS:~# restorecon /var/www/html/index.html
```

A commonly used option to the **restorecon** command is the **-R** option, which performs the changes recursively on a directory structure.

See the “ls -Z” subsection in this chapter for more details regarding security contexts.

ls -Z

Each process runs with a security context. To see this, use the **-Z** option to the **ps** command (the **head** command is used here simply to limit the output of the command):

```
root@OCS:~# ps -fe | grep httpd | head -2
system_u:system_r:httpd_t:s0 root 1109 1 0 2018 ?
00:51:56 /usr/sbin/httpd
system_u:system_r:httpd_t:s0 apache 1412 1109 0 Dec24 ?
00:00:09 /usr/sbin/httpd
```

The security context (**system_u:system_r:httpd_t:s0**) is complicated, but for understanding the basics of SELinux, the important part is **httpd_t**, which is like a security group or domain. As part of this security domain, the **/usr/sbin/httpd** process can only access files that are allowed by the security policy for **httpd_t**. This policy is typically written by someone who is an SELinux expert, and that expert should have proven experience regarding which processes should be able to access specific files and directories on the system.

Files and directories also have an SELinux security context that is defined by the policy. To see a security context for a specific file, use the **-Z** option to the **ls** command (note that the SELinux context contains so much data that the filename cannot fit on the same line):

```
root@OCS:~# ls -Z /var/www/html/index.html
unconfined_u:object_r:httpd_sys_content_t:s0/var/www/html/index.html
```

ps -Z

See the “ls -Z” subsection in this chapter.

AppArmor

AppArmor is a MAC system that plays a similar role to SELinux in that it provides a context-based permission model. This subsection describes the key components of AppArmor that are exam testable.

aa-disable

An AppArmor profile is a rule set that describes how AppArmor should restrict a process. A profile can be disabled for a specific profile by using the **aa-disable** command. Here’s an example:

```
root@OCS:~# ln -s /etc/apparmor.d/usr.sbin.mysqld
/etc/apparmor.d/disable
root@OCS:~# apparmor_parser -R /etc/apparmor.d/usr.sbin.mysqld
```

NOTE: To view the status of a profile, use the **aa-status** command. To enable a profile again, use the following commands:

```
root@OCS:~# rm /etc/apparmor.d/disable/usr.sbin.mysqld
root@OCS:~# apparmor_parser -r
/etc/apparmor.d/usr.sbin.mysqld
```

aa-complain

If you need to troubleshoot an AppArmor profile, it is best to put it into complain mode. In this mode, there are no restrictions enforced, but any problems will be reported.

Use the **aa-complain** command to put a profile into complain mode:

```
root@OCS:~# aa-complain /usr/sbin/mysqld
Setting /usr/sbin/mysqld to complain mode.
```

To put the profile back into the “enforcing” mode, use the following command:

```
root@OCS:~# sudo aa-enforce /usr/sbin/mysqld
Setting /usr/sbin/mysqld to enforce mode
```

aa-unconfined

Use the **aa-unconfined** command to list processes that are not restricted by the AppArmor profiles.

/etc/apparmor.d/

The **/etc/apparmor.d** directory is the location of the definitions of the AppArmor profiles. Note that knowing how to create or read these files is beyond the scope of the Linux+ exam, but it is important to know the location of these profiles in order to determine which profiles are available and to use the AppArmor commands, such as the **aa-disable** command.

`/etc/apparmor.d/tunables`

The `/etc/apparmor.d/tunables` directory is the location of files that can be used to fine-tune the behavior of AppArmor. Note that knowing how to create or read these files is beyond the scope of the Linux+ exam.

Privilege Escalation

The concept behind privilege escalation is that a user may need to be able to execute commands using an account that has more privileges than the user's account normally has. For example, a regular user may need to execute a command that requires root user access. There are several techniques that can provide privilege access; this subsection covers the techniques that are exam testable.

su

The `su` command allows a user to shift user accounts:

```
[student@OCS ~]# id
uid=1000(student) gid=1000(student) groups=1000(student)
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[student@localhost ~]# su root
Password:
[root@OCS ~]# id
uid=0(root) gid=0(root) groups=0(root)
context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

One option is permitted when executing the `su` command: the `-` option. When you execute the `su` command with the `-` option, a new login shell will be provided. When you're not using the `-` character, a non-login shell will be provided.

sudo

When properly configured by the administrator, users can use the `sudo` command to run commands as other users (typically as the root user). To execute a command as root, enter the following:

```
sudo command
```

You will be prompted for your own password and, if the settings in the `/etc/sudoers` file are correct, the command will execute correctly. If the settings are not correct, an error message will appear.

The following table describes common options for the **sudo** command:

| Option | Description |
|-----------------------|--|
| -b | Run the command in the background. |
| -e | Run like the sudoedit command. See the “sudoedit” subsection in this chapter. |
| -l | List which commands are allowed for this user. |
| -u <i>user</i> | Run the command as <i>user</i> rather than as the root user. |

Also see the “visudo” section in this chapter for details regarding the **/etc/sudoers** file.

wheel

A common method for providing non-root users with root access is to use the wheel group. If enabled in the **/etc/sudoers** file (normally this line is “commented out”), anyone in the wheel group will have the ability to run any command as the root user via the **sudo** command:

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

visudo

The **/etc/sudoers** file is used to determine which users can use the **sudo** command to execute commands as other users (typically as the root user). To edit this file, you must be logged in as the root user and should use the **visudo** command rather than edit the file directly.

The following table describes important definitions for the **/etc/sudoers** file:

| Option | Description |
|-------------------|---|
| User_Alias | A name that represents a group of users (for example, User_Alias ADMINS = julia, sarah) |
| Cmnd_Alias | A name that represents a group of commands (for example, Cmnd_Alias SOFTWARE = /bin/rpm, /usr/bin/yum). |

The format of an entry for the **/etc/sudoers** file uses the following syntax:

```
user                machine=commands
```

To allow the student user the ability to execute the **/usr/bin/yum** command as the root user, add an entry like the following to the **/etc/sudoers** file:

```
student            ALL=/usr/bin/yum
```

To allow all members of ADMINS the ability to execute all of the **SOFTWARE** command as the root user, add an entry like the following to the **/etc/sudoers** file:

```
ADMINS            ALL=SOFTWARE
```

sudoedit

If you want to edit a file using sudo access, consider using the **sudeoedit** or **sudo -e** command. Using this feature requires having the ability to edit a file using a command designed to edit files (such as **nano**, **vi**, or **vim**).

Example:

```
sudoedit file1
```

Note that the editor that will be chosen depends on variables. The following variables are consulted:

- SUDO_EDITOR
- VISUAL
- EDITOR

If none of these variables is set, then the vi editor is typically the default.

User Types

This section breaks down the different user types you are likely to encounter on Linux-based systems.

Root

The root account is the system administrator account. It is important to note that what makes the root account special is the UID of 0. Any user with a UID of 0 is a full system administrator. As a security note, when you're performing audits, look for any user with a UID of 0, as this is a common hacking technique.

Standard

Any account with a UID of 1000 or higher is considered a standard or regular user account. People are normally assigned standard user accounts so they can log in to the system and perform tasks.

Service

A typical Linux system will have many service user accounts. These service user accounts typically have UID values under 1000, making it easy for an administrator to recognize these as special accounts.

Some of these service accounts are often referred to as “daemon accounts” because they are used by daemon-based software. *Daemons* are programs that run in the background, performing specific system tasks.

Other service accounts may exist to provide features for the operating system. For example, the “nobody” account is used to apply permissions for files that are shared via NFS (Network File System).

Additionally, if you add new software to the system, more users might be added because software vendors make use of both user and group accounts to provide controlled access to files that are part of the software.

Symbols

- & (ampersand), 171
- \$(shell expansion (BASH), 335
- \${ shell expansion (BASH), 335
- #!/bin/bash, 332
- ? (help) command, 47
- ~/, ssh, 221–222

A

- aa-complain command, 208
- aa-disable command, 208
- aa-unconfined command, 208
- absolute paths, 41
- Accept target, firewalls, 259
- access
 - cron access, restricting, 244
 - LDAP, 238
 - local access, 309
 - PAM, 214–221
 - PKI, 230–231
 - PTY, 230
 - remote access, 309
 - SSH, 221–228
 - TACACS+, 238
 - TTY, 228–229
 - user management, 309
 - VPN as a client, 231–233
- accessibility options (desktops), 191–192
- accounts (user/group), managing
 - BASH parameters, 103–106
 - creating accounts, 94
 - deleting accounts, 95–96
 - /etc/group files, 107
 - /etc/passwd files, 106–107
 - /etc/shadow files, 107–108
 - modifying accounts, 94–95
 - profiles, 102–103
 - queries, 96–98
 - quotas, 98–102
- ACL (Access Control Lists), 258
 - getfacl command, 199–201
 - setfacl command, 199–201
 - troubleshooting, 318
- acquisition commands, 89–91
- active/passive backups, 35–36
- adapters (network), 175, 316
- add banners, 245
- agent/agentless monitoring, 351–352
- aggregation, 34–35
- ampersand (&), 171
- Anaconda, 63
- AppArmor, 208–209
- application data, separating from OS data, 241
- apt command, 80–83
- apt-cache command, 81
- apt-get command, 80–81
- archive/restore utilities, 268–269
- arguments
 - BASH, 337
 - git command, 343–349
- arp command, 286
- ASCII character sets, 75
- asymmetric cryptography, 230
- atq command, 166
- atrm command, 166

attributes (orchestration processes), 352
 audio devices, 175
 auditd daemon, 243
 authentication
 asymmetric cryptography, 230
 biometric authentication, 238
 digital signatures (message digests), 230
 external authentication, 310
 Kerberos, 238–239
 LDAP, 238
 local authentication, 309–310
 multifactor authentication, 237–238
 PAM, 214–221
 PKI, 230–231
 PTY, 230
 public/private key authentication, 230
 RADIUS, 238
 self-signed certificates, 230
 SSH, 221–228
 TACACS+ 238
 TTY, 228–229
 user management, 309–310
 VPN as a client, 231–233
 authentication. *See also* security
 authentication servers, 160–161
 authorization, privileges, 312
 automation
 automated configurations (orchestration processes), 353
 build automation (orchestration processes), 353
 jobs, 165–172
 automounts, 151–152
 awk command, 125–126

B

backups
 active/passive backups, 35–36
 differential backups, 273
 full backups, 272–273
 incremental backups, 272
 snapshot clones, 273
 bad blocks (storage), troubleshooting, 317
 bandwidth, network latency, 279
 banners (add), 245
 BASH
 \$() shell expansion, 335
 \${ } shell expansion, 335
 #!/bin/bash, 332
 arguments (positional parameters), 337
 case statements, 340
 chmod command, 333
 comments, 333
 conditional statements, 338–340
 directory permissions, 333
 escaping characters, 340–341
 exit codes, 336
 extensions, 333
 file globs (wildcards), 334
 file permissions, 333
 global entries, 104–106
 if statements, 339–340
 looping constructs, 337–338
 for loops, 337–338
 metacharacters, 336
 positional parameters (arguments), 337
 shell expansions, 334–335
 shell variables, 328–332
 sourcing scripts, 332
 test statements, 338–339
 until loops, 338
 user entries, 104
 user/group accounts, 103–106
 while loops, 337
 basic RE (Regular Expressions), 141
 bg command, 171
 biometric authentication, 238
 BIOS, 4
 passwords, 236–237
 blkid command, 51
 blobs, 64
 block storage, 64
 blocked ports (firewalls), troubleshooting, 318

- blocked protocols (firewalls),
 - troubleshooting, 318
- Bluetooth devices, 174
- bonding
 - active/passive backups, 35–36
 - aggregation, 34–35
 - load balancing, 36
- /boot directory, 7
- /boot/efi directory, 7
- /boot/grub directory, 7
- /boot/grub2 directory, 7
- boot loaders, 1
 - GRUB, 2
 - Boot Menu screen, 3
 - Stanza-editing screen, 4
 - GRUB2, 2
 - passwords, 236
- Boot Menu screen (GRUB), 3
- boot modules/files, 7
 - commands, 8–9
 - initramfs, 9
 - vmlinix, 9
- boot process
 - EFI, 4
 - FTP, booting from, 5
 - HTTP, booting from, 5
 - ISO, booting from, 5
 - NFS, 5
 - options, 3
 - overview of, 2
 - PXE, 5
 - security, 236–237
 - Systemd management, 144–145
 - UEFI, 4
- bootstrapping, 62
 - Anaconda, 63
 - Cloud-Init, 62
 - Kickstart, 63
- bounce keys (keyboards), 192
- Braille displays, 192
- brctl command, 27
- bridging, networks, 65
- buffer cache output, 295
- build automation (orchestration processes), 353
- build tools, software

- commands, 86–88
- compilers, 88
- shared libraries, 88

- bzip2 command, 271

C

- CA (Certificate Authority), 159
- cal command, 121
- case statements (BASH), 340
- cat command, 115
- certificates (self-signed), 230
- chage command, 95
- character sets
 - ASCII, 75
 - Unicode, 75
 - UTF-8, 75
- chattr command, 311–312
- chcon command, 207
- chgrp command, 198–199
- chkconfig command, 153–154
- chmod command, 195, 196–198, 333
- chown command, 198
- chroot jail service, 240
- cifs filesystems, 58
- Cinnamon, 189
- cloud-computing
 - blobs, 64
 - blocks, 64
 - bootstrapping, 62–63
 - bridging, 65
 - dual-homed systems, 66
 - hypervisors, 66
 - libvirt, 67
 - local networks, 65–66
 - NAT, 65
 - networks, 65–66
 - overlay networks, 65
 - PV, 64
 - storage, 63–64
 - templates, 60–62
 - thick provisioning, 63
 - thin provisioning, 63–64
 - virsh, 67
 - VMM, 67

- Cloud-Init, 62
 - clustering, 164
 - at command, 165–167
 - commands
 - acquisition commands, 89–91
 - Boot Menu screen (GRUB), 3
 - boot modules/files, 8–9
 - build tools, software, 86–88
 - diagnostic tools, 18–27
 - EXT tools, 45, 48, 51–53
 - filesystems, 44–53
 - kernel modules, 11–15
 - localization, 70–73
 - LVM tools, 44–45
 - network monitoring/configuration, 280–287
 - process commands, 300–305
 - Stanza-editing screen (GRUB), 4
 - storage, 44–53
 - XFS tools, 44
 - comments (BASH), 333
 - communications ports, troubleshooting, 321–322
 - compatibility (hardware/software), troubleshooting, 324
 - compilers, 88
 - compression utilities, 269–272
 - conditional statements (BASH), 338–340
 - configuration files, 28–33
 - configuring
 - CPU, 289–292
 - CUPS, 179–181
 - dmesg command, 178
 - lsblk command, 178
 - lsdev command, 176
 - lspci command, 177–178
 - lsusb command, 176–177
 - memory, 292–295
 - network interfaces, 280
 - networks, 278–287
 - SELinux, 203–204
 - storage, 287–289
 - udevadm command, 181–182
 - console redirection. *See* port forwarding
 - container images, 62
 - containers, 162
 - context-based permissions, 202–203
 - cp command, 129–130
 - cpio command, 268–269
 - CPU monitoring/configuration, 289–292
 - cron
 - access, restricting, 244
 - ampersand (&), 171
 - atq command, 166
 - atrm command, 166
 - bg command, 171
 - at command, 165–167
 - crontab command, 167–170
 - Ctrl-c, 172
 - Ctrl-z, 172
 - /etc/at.allow files, 166–169
 - /etc/at.deny files, 166–167
 - /etc/cron.deny files, 168–169
 - fg command, 170
 - kill command, 171–172
 - nohup command, 172
 - security, 244
 - Ctrl-Alt-Del, disabling, 244
 - Ctrl-c, 172
 - Ctrl-z, 172
 - CUPS (Common Unix Printing Systems), 179–181
 - cupscctl command, 179–180
 - curl command, 90–91
 - cut command, 127–128
 - CVE monitoring, 243
- ## D
- d command, 46
 - daemons, 211
 - auditd, 243
 - fail2ban, 263–264
 - databases, 162–163
 - date command, 72
 - dd command, 39, 269, 315
 - .deb extension, 78
 - degraded storage, troubleshooting, 314
 - deleting user/group accounts, 95–96

- DenyHost, 263
- dependencies
 - patches, 319
 - troubleshooting, 319–320
- depmod command, 15
- desktops
 - accessibility options, 191–192
 - remote desktops, 189–191
- destination, firewall filtering by, 258
- /dev/ 54–55
- /dev filesystem, 183
- /dev/disk directory, 55–56
- /dev/disk-by, 55–56
- /dev/mapper. *See* device mapper
- /dev/null files, 119–120
- /dev/tty files, 120–121
- /dev/tty# files, 229
- device mappers, 183
 - logical volumes, 43–44
 - LVM, 41–44
 - Multipath, 44
 - physical volumes, 42
 - volume groups, 42–43
- df command, 49
- DHCP (Dynamic Host Configuration Protocol), 159–160
- diagnostic tools, 18–27
- diff command, 140
- differential backups, 273
- dig command, 19–20, 286
- digital signatures (message digests), 230
- directories
 - absolute paths, 41
 - BASH directory permissions, 333
 - /boot directory, 7
 - /boot/efi directory, 7
 - /boot/grub directory, 7
 - /boot/grub2 directory, 7
 - cp command, 129–130
 - /dev/disk directory, 55–56
 - diff command, 140
 - du command, 49
 - /etc/apparmor.d/ directory, 208
 - /etc/apparmor.d/tunables directory, 209
 - /etc/cups directory, 180
 - /etc/grub.d directory, 6
 - /etc/modprobe.d, 16
 - /etc/netplan directory, 31–32
 - /etc/network directory, 30
 - /etc/pam.d/ directory, 219–220
 - /etc/ssh directory, 225–227
 - find command, 134, 136–138
 - hard links, 133–134
 - inodes, 135
 - /lib/modprobe.d, 16
 - ln command, 132–134
 - locate command, 138
 - ls command, 131, 134
 - mkdir command, 132
 - mv command, 129
 - permissions, 194–202, 308
 - /proc/cmdline, 57
 - /proc/cpuinfo, 57
 - /proc/devices, 57
 - /proc/mdstat, 57
 - /proc/meminfo, 57
 - /proc/modules, 57
 - /proc/swaps, 57
 - /proc/sys, 57
 - /proc/vmstat, 57
 - relative paths, 41
 - rm command, 130
 - rmdir command, 132
 - rsync command, 131–132
 - /run/modprobe.d, 16
 - scp command, 130
 - service management, 149–150
 - soft (symbolic) links, 132–133
 - stat command, 135
 - storage directories, 54–57
 - /sys/block, 56
 - /sys/bus, 56
 - /sys/bus/cpu, 56
 - /sys/bus/cpu/devices, 56
 - Systemd management, 149–150
 - touch command, 129
 - unlink command, 135
 - updatedb command, 140
 - /usr/share/doc directory, 32

- `/usr/share/zoneinfo`, 70
 - `/var/log/journal` directory, 254
 - `whereis` command, 139
 - `which` command, 139
- disabled mode (SELinux), 204
- disk encryption, 244
- displays (monitors), 174
 - Braille displays, 192
 - onscreen keyboards, 192
 - screen magnifiers, 192
 - screen readers, 192
- `dmesg` command, 14, 178
- `dmidecode` command, 324
- DNAT (Destination NAT), 262
- `dnf` command, 86
- `dpkg` command, 79–80
- `dracut` command, 8
- drivers (GPU), troubleshooting, 321
- Drop target, firewalls, 259
- DTLS (Datagram Transport Layer Security), 233
- `du` command, 49
- dual-homed systems, 66
- `dumpe2fs` command, 51–52
- dynamic rule sets, firewalls, 263

E

- `e2label` command, 53
- `echo` command, 124, 331–332
- `edquota` command, 99–100
- EFI (Extensible Firmware Interface), 4
- `egrep` command, 128
- email
 - mail servers, 163
 - postfix, 243
 - sendmail, 242–243
- encryption (disk), 244
- enforcing mode (SELinux), 204
- `env` command, 330
- environment variables, 73, 330
 - LANG variable, 74
 - LC_* variable, 73–74
 - LC_ALL variable, 74
 - TZ variable, 74–75

- escaping characters (BASH), 340–341
- `/etc/apparmor.d/` directory, 208
- `/etc/apparmor.d/tunables` directory, 209
- `/etc/apt/sources.list`, 81–82
- `/etc/at.allow` files, 166–169
- `/etc/at.deny` files, 166–167
- `/etc/cron.deny` files, 168–169
- `/etc/cups` directory, 180
- `/etc/default/grub`, 6
- `/etc/dhcpd.conf`, 32–33
- `/etc/fstab`, 54
- `/etc/group` files, 107
- `/etc/grub2.cfg`, 7
- `/etc/grub.d` directory, 6
- `/etc/hosts`, 29
- `/etc/logrotate.conf` files, 250–252
- `/etc/modprobe.conf`, 16
- `/etc/modprobe.d`, 16
- `/etc/mtab`, 56
- `/etc/netplan` directory, 31–32
- `/etc/network` directory, 30
- `/etc/nsswitch.conf`, 30
- `/etc/pam.d/` directory, 219–220
- `/etc/passwd` files, 106–107
- `/etc/rc.local` files, 155
- `/etc/resolv.conf`, 31
- `/etc/rsyslog.conf` files, 252–253
- `/etc/securetty` files, 229
- `/etc/services` files, 265
- `/etc/shadow` files, 107–108
- `/etc/ssh` directory, 225–227
- `/etc/sysconfig/network`, 29
- `/etc/sysconfig/network-scripts`, 28–29
- `/etc/sysctl.conf`, 32
- `/etc/systemd/journald.conf` files, 254
- `/etc/timezone`, 70
- `/etc/udev/rules.d`, 186
- `/etc/X11/xorg.conf` files, 184–185
- `/etc/yum/conf`, 84–85
- `ethtool` command, 22–24
- executables (permissions), troubleshooting, 319
- execute permission, 194
- exit codes (BASH), 336
- `export` command, 329–330

EXT tools, 45
 dumpe2fs command, 51–52
 mkfs command, 48
 resize2fs command, 52
 tune2fs command, 53
 ext3 filesystems, 58
 ext4 filesystems, 58
 extended RE (Regular Expressions), 141
 external authentication, 310

F

fail2ban, 263–264
 faillock, PAM, 220–221
 fdisk command, 46
 fg command, 170
 file globs (wildcards), 334
 file readers
 cat command, 115
 grep command, 114–115
 head command, 116
 less command, 116
 more command, 116
 tail command, 115–116
 file servers, 160
 files
 absolute paths, 41
 archive/restore utilities, 268–269
 backups, 272–274
 BASH file permissions, 333
 /boot directory, 7
 /boot/efi directory, 7
 /boot/grub directory, 7
 /boot/grub2 directory, 7
 boot modules, 7–9
 compression utilities, 269–272
 configuration files, 28–33
 cp command, 129–130
 creating, 310–312
 /dev/tty# files, 229
 diff command, 140
 /etc/apt/sources.list, 81–82
 /etc/at.allow files, 166–169
 /etc/at.deny files, 166–167
 /etc/cron.deny files, 168–169
 /etc/default/grub, 6
 /etc/grub2.cfg, 7
 /etc/grub.d directory, 6
 /etc/logrotate.conf files, 250–252
 /etc/rc.local files, 155
 /etc/rsyslog.conf files, 252–253
 /etc/securetty files, 229
 /etc/services files, 265
 /etc/systemd/journald.conf files, 254
 /etc/X11/xorg.conf files, 184–185
 /etc/yum/conf, 84–85
 file readers, 114–116
 find command, 134, 136–138
 .git file, 350
 .gitignore file, 349–350
 hard links, 133–134
 immutable files, 311–312
 inode exhaustion, 311
 inodes, 135
 integrity checks, 275
 kernel modules, 15–16
 ln command, 132–134
 localization, 70
 locate command, 138
 log files, 158
 ls command, 131, 134
 mkdir command, 132
 mv command, 129
 off-site/off-system storage, 274–275
 permissions, 194–202, 308
 /proc/cmdline, 57
 /proc/cpuinfo, 57
 /proc/cpunfo files, 289–291
 /proc/devices, 57
 /proc/mdstat, 57
 /proc/meminfo, 57
 /proc/meminfo files, 295
 /proc/modules, 57
 /proc/swaps, 57
 /proc/sys, 57
 /proc/vmstat, 57
 quotas, 310
 redirecting, 117–123
 relative paths, 41
 rm command, 130

- rmdir command, 132
 - rsync command, 131–132
 - scp command, 130
 - soft (symbolic) links, 132–133
 - ssh_conf files, 225–226
 - sshd_conf files, 226–227
 - stat command, 135
 - storage, 54–57, 274–275, 311
 - /sys/block, 56
 - /sys/bus, 56
 - /sys/bus/cpu, 56
 - /sys/bus/cpu/devices, 56
 - text editors, 110–114
 - text processing, 123–128
 - touch command, 129
 - unlink command, 135
 - updatedb command, 140
 - /var/log/[application] files, 248
 - /var/log/kern.log files, 249
 - /var/log/messages files, 248
 - /var/log/secure files, 247–248
 - whereis command, 139
 - which command, 139
- filesystems, 56
- absolute paths, 41
 - cifs, 58
 - commands, 44–53
 - /dev/54–55
 - /dev filesystem, 183
 - dumpe2fs command, 51–52
 - e2label command, 53
 - /etc/fstab, 54
 - ext3, 58
 - ext4, 58
 - fsck command, 52–53
 - hierarchy of, 40–41
 - lsblk command, 51
 - mkfs command, 48
 - mount command, 49–50
 - nfs, 58
 - ntfs, 58
 - /proc, 182
 - /proc/mounts, 57
 - real filesystems, 40
 - relative paths, 41
 - smb, 58
 - /sys, 56, 182
 - umount command, 50–51
 - usrquota mount option, 98–99
 - virtual filesystems, 40–41
 - xfs, 58
- find command, 134, 136–138
- finger command, 242
- firewalls
- Accept target, 259
 - ACL, 258, 318
 - blocked ports, troubleshooting, 318
 - blocked protocols, troubleshooting, 318
 - DenyHost, 263
 - Drop target, 259
 - dynamic rule sets, 263
 - /etc/services files, 265
 - fail2ban, 263–264
 - filtering by
 - destination, 258
 - logs, 258
 - ports, 258
 - protocols, 258
 - source, 258
 - firewalld, 259–260
 - IP forwarding, 263
 - IPSet, 265
 - iptables, 260–262
 - Log target, 259
 - Netfilter, 262–263
 - packet filtering, 260–262
 - privileged ports, 265
 - Reject target, 259
 - remote access, 309
 - run time rules, 260
 - stateful/stateless rules, 258–259
 - troubleshooting, 317–318
 - ufw, 262
 - zones, 260
- free command, 294–295
- fsck command, 52–53
- FTP (File Transfer Protocol)
- booting from, 5
 - servers, 160, 242
- full backups, 272–273

G

getenforce command, 205
 getfacl command, 199–201
 getsebool command, 206–207
 Git
 arguments, 343–349
 git branch command, 347–348
 git clone command, 343–344
 git commit command, 345
 git config command, 349
 .git file, 350
 .gitignore file, 349–350
 git init command, 348
 git merge command, 345–346
 git mergetool command, 346–347
 git pull command, 344
 git push command, 344
 git status command, 346
 GNOME, 189
 GPIO (General Purpose Input/Output),
 174–175
 GPT (GUID Partition Tables), 39
 GPU drivers, troubleshooting, 321
 grep command, 114–115, 118
 group management
 BASH parameters, 103–106
 creating groups, 94
 deleting accounts, 96
 /etc/group files, 107
 /etc/passwd files, 106–107
 /etc/shadow files, 107–108
 modifying accounts, 94–95
 profiles, 102–103
 queries, 96–98
 quotas, 102
 group membership permissions, 319
 groupadd command, 94
 groupdel command, 96
 groupmod command, 95
 GRUB (GRand Unified Bootloader), 2
 Boot Menu screen, 3
 Stanza-editing screen, 4
 GRUB2 (GRand Unified Bootloader 2), 2
 grub2-install command, 8

grub2-mkconfig command, 8–9
 GUI (Graphical User Interface), 188–189
 Cinnamon, 189
 GNOME, 189
 KDE Plasma, 189
 MATE, 189
 NX, 190
 remote desktops, 189–190
 servers, 187–188
 Spice, 190
 Unity, 189
 VNC, 190
 Wayland, 188
 X11, 188
 XRDP, 190
 .gz extension, 78
 gzip command, 269–270

H

hard disks
 device names, 39
 partitions, 241
 hard links, 133–134
 hardware issues, troubleshooting, 320–325
 hardware tokens, multifactor
 authentication, 237
 hashing, 230
 HBA (Host Bus Adapters), 175, 317
 head command, 116
 help (?) command, 47
 here documents, 122–123
 HOME variable, 328
 host command, 20, 287
 hosted hypervisors, 66
 hostnamectl command, 150–151
 hosts, denying, 240–241, 263
 hot pluggable devices, 185–186
 HTTP (HyperText Transfer Protocol),
 booting from, 5
 hwclock command, 72–73
 hypervisors
 hosted hypervisors, 66
 native hypervisors, 66
 types of, 66

I

IaC (Infrastructure as Code), orchestration processes, 352

ID (Identification)
 PID, 305
 shared ID and security, 240

id command, 96

ID variable, 328

if statements (BASH), 339–340

ifcfg-etho, 28

ifcfg-interface, 28–29

iftop command, 281

ifup-wireless, 28

image backups, 273–274

immutable files, 311–312

incremental backups, 272

infrastructure automation (orchestration processes), 352

inheritance, permissions, 196

initramfs, 9

inodes, 135, 311

insmod command, 12

installations
 Anaconda, 63
 Kickstart, 63
 make install command, 87
 packages, 78–86

instances (orchestration processes), 353

integrity checks, 275

interfaces (network), configuring, 280

interruptible sleep state (processes), 298

inventories (orchestration processes), 353

IO scheduling, 288–289

ioping command, 288

iostat command, 48–49

IP addresses, name resolution, 280

ip command, 21–22, 287

IP forwarding, 263

IPA (Identity, Policy and Audit), 161

iperf command, 282

IPsec (Internet Protocol Security), 232–233

IPSet, 265

ipset command, 282–283

iptables, 260–262

ISO, booting from, 5

iwconfig command, 25–26

J

JavaScript, JSON, 61

jobs
 automating, 165–172
 scheduling, 165–172

journalctl command, 253–254

journald, 253

JSON, 61

K

KDE Plasma, 189

Kerberos, 161, 238–239

kernel modules
 boot modules/files, 7–9
 commands, 11–15
 files, 15–16

kernel panic, 9

keyboards
 bounce keys, 192
 mapping, 324
 mouse keys, 192
 onscreen keyboards, 192
 repeat keys, 192
 slow keys, 192
 sticky keys, 192
 toggle keys, 192
 troubleshooting, 324

Kickstart, 63

kill command, 171–172, 299–300

kinit command, 239

klist command, 239

L

l command, 46

LANG variable, 74

last command, 98

- lastb command, 255
 - latency, networks, 279
 - LC_* variable, 73–74
 - LC_ALL variable, 74
 - LDAP (Lightweight Directory Access Protocol), 161, 215–216, 238
 - ldd command, 87–88
 - less command, 116
 - /lib/modprobe.d, 16
 - libraries (shared), 88
 - libvirt, 67
 - links
 - hard links, 133–134
 - soft (symbolic) links, 132–133
 - ln command, 132–134
 - load balancing, 36, 163–164
 - loadaverage command, 291
 - local access, 309
 - local authentication, 309–310
 - local networks, 65–66
 - local port forwarding, 190–191
 - local variables, 328–330
 - localectl command, 70–71
 - localhost, 280
 - localization
 - character sets, 75
 - commands, 70–73
 - environment variables, 73–75
 - /etc/timezone, 70
 - /usr/share/zoneinfo, 70
 - locate command, 138
 - log files, timestamps, 158
 - Log target, firewalls, 259
 - logging, 161–162
 - /etc/logrotate.conf files, 250–252
 - /etc/rsyslog.conf files, 252–253
 - /etc/systemd/journald.conf files, 254
 - firewall filtering by logs, 258
 - journalctl command, 253–254
 - journald, 253
 - lastb command, 255
 - logrotate command, 250–252
 - managing, 249–255
 - syslog, 249–250, 252–253
 - systemd-journald service, 254
 - /var/log/[application] files, 248
 - /var/log/journal directory, 254
 - /var/log/kern.log files, 249
 - /var/log/messages files, 248
 - /var/log/secure files, 247–248
 - logical devices, df command, 49
 - logical partitions, 38
 - logical volumes, 43–44, 52
 - logins
 - password-less logins, 239
 - root logins, disabling, 239
 - LOGNAME variable, 328
 - looping constructs (BASH), 337–338
 - for loops (BASH), 337–338
 - lost root password, 295
 - lpadmin command, 179
 - ls command, 118, 131, 134
 - lsattr command, 311
 - lsblk command, 51, 178
 - lsdev command, 176
 - lshw command, 325
 - lsmod command, 11–12
 - lsOF command, 303–304
 - lspci command, 177–178, 316
 - lsusb command, 176–177, 322–323
 - ls-Z command, 207
 - LUKS (Linux Unified Key Setup), 244
 - lvdisplay command, 45
 - lvextend command, 45
 - LVM (Logical Volume Manager), 41–44
 - snapshot clones,
 - tools, 44–45
- ## M
- m command, 46
 - mail servers, 163
 - make command, 86–87
 - make install command, 87
 - managing
 - automated configurations
 - (orchestration processes), 353
 - logs, 249–255
 - permissions, 196–202
 - processes, 297–298

- services
 - automounts, 151–152
 - chkconfig command, 153–154
 - directories, 149–150
 - hostnamectl command, 150–151
 - Systemd management, 144–148
 - SysVinit, 152–156
 - unit files, 148–150
 - Systemd
 - boot process, 144–145
 - directories, 149–150
 - systemctl command, 145–148
 - systemd-analyze command, 147–148
 - targets, 144–145
 - unit files, 148–149
 - user/group accounts
 - BASH parameters, 103–106
 - creating accounts, 94
 - deleting accounts, 95–96
 - /etc/group files, 107
 - /etc/passwd files, 106–107
 - /etc/shadow files, 107–108
 - modifying accounts, 94–95
 - privilege escalation, 209–211
 - profiles, 102–103
 - queries, 96–98
 - quotas, 98–102
 - root accounts, 211
 - service accounts, 211–212
 - standard accounts, 211
 - MASQUERADE, 263
 - MATE, 189
 - MBR (Master Boot Records), 39
 - md5sum command, 275
 - mdadm command, 45–46
 - memory
 - buffer cache output, 295
 - OOM Killer, 294
 - troubleshooting, 320
 - memory monitoring/configuration, 292–295
 - message digests (digital signatures), 230
 - metacharacters (BASH), 336
 - mkdir command, 132
 - mkfs command, 48
 - mkinitrd command, 8
 - mkpart command, 47
 - mkpartfs command, 47
 - mkswap command, 293
 - modinfo command, 13–14
 - modprobe command, 13
 - monitoring, 162
 - agent/agentless monitoring, 351–352
 - CPU, 289–292
 - CUPS, 179–181
 - CVE monitoring, 243
 - dmesg command, 178
 - lsblk command, 178
 - lsdev command, 176
 - lspci command, 177–178
 - lsusb command, 176–177
 - memory, 292–295
 - networks, 278–287
 - storage, 287–289
 - udevadm command, 181–182
 - monitors (displays), 174
 - Braille displays, 192
 - onscreen keyboards, 192
 - screen magnifiers, 192
 - screen readers, 192
 - more command, 116
 - MOTD (Message of the Day), 245
 - mount command, 49–50
 - mount points (partitions), troubleshooting, 315
 - mouse keys (keyboards), 192
 - mtr command, 285–286
 - multifactor authentication, 237
 - biometrics, 238
 - OTP, 237
 - tokens, 237
 - Multipath, 44
 - mv command, 129
- ## N
- n command, 46
 - name resolution, 280

name servers, 159
naming conventions, hard disk devices, 39
nano editor, 110–111
NAT (Network Address Translation), 65

- DNAT, 262
- MASQUERADE, 263
- SNAT, 262

native hypervisors, 66
netcat command, 284–285
Netfilter, 262–263
netplan apply command, 32
netstat command, 18–19, 281
network adapters, 175
networks

- adapters, 175, 316
- bonding, 34–36
- bridging, 65
- cloud-computing, 65–66
- configuring, 278–287
- dual-homed systems, 66
- interface configurations, 280
- latency, 279
- local networks, 65–66
- localhost, 280
- monitoring, 278–287
- name resolution, 280
- NAT, 65
- overlay networks, 65
- packet drops, 279
- remote access, 309
- routing, 279
- saturation, 279
- timeouts, 279
- Unix sockets, 280
- virtualization, 65–66
- VPN, 231–233

NFS (Network File Systems), 5, 160
nfs filesystems, 58
nice command, 300
nl command, 119
nmap command, 280–281
nmcli command, 26, 287
nmtui command, 27
nohup command, 172
nslookup command, 19, 286

ntfs filesystems, 58
NTP (Network Transfer Protocol), 158
NX, 190

O

octal notation. *See* chmod permissions
off-site/off-system storage, 274–275
OLDPWD variable, 328
onscreen keyboards, 192
OOM (Out of Memory) Killer, 294
orchestration processes

- agent/agentless monitoring, 351–352
- attributes, 352
- automated configurations, 353
- build automation, 353
- IaC, 352
- infrastructure automation, 352
- instances, 353
- inventories, 353
- procedures, 352

OS (Operating Systems), separating data from application data, 241
OTP (One-Time Passwords), 237
output redirection, files, 116–123
OVA, 60
overlay networks, 65
OVF (Open Virtualization Format), 61
ownership permissions, troubleshooting, 318–319

P

p command, 46
packages. *See also* repositories; software

- apt command, 80–83
- apt-cache command, 81
- apt-get command, 80–81
- .deb extension, 78
- dnf command, 86
- dpkg command, 79–80
- .gz extension, 78
- installation tools, 78–86
- rpm command, 78–79

- .rpm extension, 78
- .tar extension, 78
- .tgz extension, 78
- yum command, 83–86
- yum list command, 83
- yumdownloader command, 84
- zypper command, 86
- packet drops, 279
- packet filtering, 260–262
- PAM (Pluggable Authentication Modules), 214–215
 - control values, 217–219
 - /etc/pam.d/ directory, 219–220
 - faillock, 220–221
 - LDAP integration, 215–216
 - pam_tally2, 220
 - password policies, 215
 - types of, 217
- parted command, 46–47
- partitions, 38
 - df command, 49
 - fdisk command, 46
 - GPT, 39
 - iostat command, 48–49
 - logical partitions, 38
 - MBR, 39
 - mkfs command, 48
 - mount points, troubleshooting, 315
 - parted command, 46–47
 - primary partitions, 38
 - /proc/partitions, 57
 - raw devices, 39
 - security, 241
 - structure of, 38–39
- partprobe command, 289
- passive/active backups, 35–36
- passwd command, 95
- passwords
 - BIOS, 236–237
 - boot loaders, 236
 - chage command, 95
 - group accounts, 95
 - lost root password, 295
 - OTP, 237
 - PAM, 215
 - passwd command, 95
 - password-less logins, 239
 - UEFI, 236–237
 - user accounts, 95
- patches, troubleshooting, 319
- PATH variable, 328–329
- PCI (Peripheral Component Interconnect), 175
- performance
 - kill signals, 299–300
 - PID, 305
 - process commands, 300–305
 - process management, 297–298
 - storage, troubleshooting, 315
- permissions
 - AppArmor, 208–209
 - BASH directory/file permissions, 333
 - chgrp command, 198–199
 - chmod command, 195, 196–198
 - chown command, 198
 - context-based permissions, 202–209
 - directories, 194–202, 308
 - executables, troubleshooting, 319
 - execute, 194
 - files, 194–202, 308
 - getfacl command, 199–201
 - group membership, 319
 - inheritance, 196
 - managing, 196–202
 - ownership permissions, troubleshooting, 318–319
 - read, 194
 - SELinux
 - configuring, 203–204
 - policies, 204–205
 - tools, 205–208
 - setfacl command, 199–201
 - sgid, 194–195
 - sticky bit, 194–195
 - suid, 194–195
 - troubleshooting, 318–319
 - ulimit command, 201–202
 - umask command, 195–196
 - user management, 307–308
 - write, 194

- permissive mode (SELinux), 204
- pgrep command, 304–305
- physical volumes, 42
- PID (Process ID), 305
- ping command, 18, 287
- PKI (Public Key Infrastructure), 230–231, 240
- pkill command, 305
- policies
 - PAM password policies, 215
 - SELinux policies, 204–205
 - violations, 310
- port forwarding
 - local port forwarding, 190–191
 - remote port forwarding, 191
 - X11 forwarding, 191
- ports
 - blocked ports (firewalls),
 - troubleshooting, 318
 - communications ports,
 - troubleshooting, 321–322
 - firewall filtering by, 258
 - privileged ports, firewalls, 265
 - security, 241–242
- positional parameters (arguments), BASH, 337
- postfix, 243
- primary partitions, 38
- print command, 47
- print servers, 163
- printenv command, 331
- printers, 175
 - CUPS, 179–181
 - lpadmin command, 179
 - troubleshooting, 320–321
- private/public key authentication, 230
- privileged ports, firewalls, 265
- privileges
 - authorization, 312
 - escalation, 209–211
 - user management, 310–312
- /proc filesystem, 182
- /proc/cmdline, 57
- /proc/cpuinfo, 57
- /proc/cpunfo files, 289–291
- /proc/devices, 57
- /proc/mdstat, 57
- /proc/meminfo, 57
- /proc/meminfo files, 295
- /proc/modules, 57
- /proc/mounts, 57
- /proc/partitions, 57
- /proc/swaps, 57
- /proc/sys, 57
- /proc/vmstat, 57
- procedures (orchestration processes), 352
- processes
 - ampersand (&), 171
 - bg command, 171
 - Ctrl-c, 172
 - Ctrl-z, 172
 - fg command, 170
 - interruptible sleep, 298
 - kill command, 171–172
 - kill signals, 299–300
 - lsof command, 303–304
 - nice command, 300
 - nohup command, 172
 - orchestration processes, 351–353
 - pgrep command, 304–305
 - PID, 305
 - pkill command, 305
 - priorities, 298
 - ps command, 303
 - renice command, 300–301
 - running, 298
 - states of, 297–298
 - time command, 302–303
 - top command, 301–302
 - uninterruptible sleep, 298
 - zombie processes, 298
- profiles (user/group accounts), 102–103
- protocols (firewalls)
 - filtering by, 258
 - troubleshooting, 318
- proxy servers, 161
- ps command, 303
- PS1 variable, 328
- PTY, 230
- public/private key authentication, 230

PV (Persistent Volumes), 64
 pvdisplay command, 45
 PWD variable, 328
 PXE (Preboot Execution Environment), 5

Q

q command, 46
 queries (user/group accounts), 96–98
 quit command, 47
 quotacheck command, 99
 quotas
 edquota command, 99–100
 files, 310
 group quotas, 102
 quota command, 100–101
 quotacheck command, 99
 quotaon command, 101–102
 repquota command, 101
 user quotas, 98–102

R

RADIUS (Remote Authentication Dial-In Service), 161, 238
 RAID devices
 mdadm command, 45–46
 troubleshooting, 317
 raw command, 39
 raw devices, 39
 RE (Regular Expressions), 141
 read permission, 194
 real filesystems, 40
 redirecting files, 116–123
 Reject target, firewalls, 259
 relative paths, 41
 remote access, 309
 remote desktops, 189–191
 remote port forwarding, 191
 remote systems, acquisition commands, 89–91
 renice command, 300–301
 repeat keys (keyboards), 192
 repositories. *See also* packages; software configuring, 88

 creating, 88–89
 locations, 89
 syncing, 89
 repquota command, 101
 resize2fs command, 52
 resource exhaustion (storage),
 troubleshooting, 316
 restore/archive utilities, 268–269
 restorecon command, 207
 rm command, 47, 130
 rmdir command, 132
 rmmmod command, 14–15
 root logins, disabling, 239
 root passwords (lost), 295
 root user accounts, 211
 route command, 21, 281
 routing, networks, 279
 rpm command, 78–79
 .rpm extension, 78
 rsync command, 131–132
 rsyslogd command, 249–250
 rule sets (dynamic), firewalls, 263
 /run/modprobe.d, 16
 /run/udev/rules.d (volatile rules), 185
 run time rules, firewalls, 260
 runlevel command, 155
 runlevels, SysVinit, 154–155
 running processes, 298

S

Samba, 160
 sar command, 291
 SATA (Serial AT Attachment), 175, 317
 saturation, networks, 279
 scheduling jobs, 165–172
 scp command, 130
 screen magnifiers, 192
 screen readers, 192
 SCSI (Small Computer System Interface), 175, 316
 searches, timestamp, 158
 security. *See also* authentication
 add banners, 245

- application data, separating from OS
 - data, 241
- auditd daemon, 243
- boot process, 236–237
- chroot jail service, 240
- cron access, restricting, 244
- Ctrl-Alt-Del, disabling, 244
- CVE monitoring, 243
- disk encryption, 244
- DTLS, 233
- finger command, 242
- firewalls, 258–265, 309, 317–318
- FTP servers, 242
- hosts, denying, 240–241
- IPsec, 232–233
- Kerberos, 238–239
- LDAP, 238
- logins
 - disabling root logins, 239
 - password-less logins, 239
- LUKS, 244
- MOTD, 245
- OS data, separating from application
 - data, 241
- OTP, 237
- partitions, 241
- password-less logins, 239
- passwords, 95, 295
 - BIOS, 236–237
 - boot loaders, 236
 - OTP, 237
 - PAM, 215
 - UEFI, 236–237
- PKI, 240
- ports, 241–242
- postfix, 243
- RADIUS, 238
- root logins, disabling, 239
- sendmail, 242–243
- services, 309
- shared ID, 240
- SSL, 232, 243
- TACACS+ 238
- telnet servers, 242
- TLS, 232, 243
- unused/unsecure services, disabling/
 - uninstalling, 242–243
- USB devices, 243–244
- sed command, 126–127
- self-signed certificates, 230
- SELinux
 - chcon command, 207
 - configuring, 203–204
 - context violations, 314
 - disabled mode, 204
 - enforcing mode, 204
 - getenforce command, 205
 - getsebool command, 206–207
 - ls-Z command, 207
 - permissive mode, 204
 - policies, 204–205
 - restorecon command, 207
 - sestatus command, 205
 - setenforce command, 205
 - setsebool command, 206
- sendmail, 242–243
- server roles
 - authentication servers, 160–161
 - CA, 159
 - clustering, 164
 - containers, 162
 - databases, 162–163
 - DHCP, 159–160
 - file servers, 160
 - FTP servers, 160
 - IPA, 161
 - Kerberos, 161
 - LDAP, 161
 - load balancing, 163–164
 - logging, 161–162
 - mail servers, 163
 - monitoring, 162
 - name servers, 159
 - NFS, 160
 - NTP, 158
 - print servers, 163
 - proxy servers, 161
 - RADIUS, 161
 - Samba, 160
 - SFTP servers, 160

- SSH, 158–159
- VPN, 162
- Web (World Wide), 159
- servers
 - FTP, 242
 - GUI, 187–188
 - postfix, 243
 - sendmail, 242–243
 - telnet, 242
 - Wayland, 188
 - X11, 188
- service command, 155–156
- service user accounts, 211–212
- services
 - automounts, 151–152
 - chkconfig command, 153–154
 - directories, 149–150
 - /etc/services files, 265
 - hostnamectl command, 150–151
 - journald, 253
 - security, 309
 - Systemd management, 144–148
 - systemd-journald service, 254
 - SysVinit, 152–156
 - unit files, 148–150
 - unused/unsecure services, disabling/
uninstalling, 242–243
- sestatus command, 205
- set command, 328, 330–331
- setenforce command, 205
- setfacl command, 199–201
- setsebool command, 206
- sftp command, 274
- SFTP servers, 160
- sgid permissions, 194–195
- SHA, 275
- shared ID and security, 240
- shared libraries, 88
- shell variables, 328
 - echo command, 331–332
 - env command, 330
 - environment variables, 330
 - export command, 329–330
 - HOME variable, 328
 - ID variable, 328
 - local variables, 328, 329–330
 - LOGNAME variable, 328
 - OLDPWD variable, 328
 - PATH variable, 328–329
 - printenv command, 331
 - PS1 variable, 328
 - PWD variable, 328
 - set command, 328, 330–331
 - user management, 312
- single user mode, 295
- slow keys (keyboards), 192
- smb filesystems, 58
- snapshot clones, backups, 273
- SNAT (Source NAT), 262
- sockets (Unix), 280
- soft (symbolic) links, 132–133
- software. *See also* packages; repositories
 - build tools, 86–88
 - compatibility, 324
 - compilers, 88
 - dependencies, 319–320
 - ldd command, 87–88
 - make command, 86–87
 - make install command, 87
 - patches, troubleshooting, 319
 - shared libraries, 88
 - timeouts, 279
 - tokens, multifactor authentication,
237
 - troubleshooting, 324
 - updating, 320
 - versioning, 320
- sort command, 124–125
- sound devices, 175
- source, firewall filtering by, 258
- source command, 332
- sourcing scripts, BASH, 332
- Spice, 190
- ss command, 24–25
- SSH (Secure Shell), 158–159, 221
 - ~/.ssh, 221–222
 - /etc/ssh directory, 225–227
 - port forwarding, 190–191

- ssh_conf files, 225–226
- ssh-add command, 228
- ssh-copy-id command, 227
- sshd_conf files, 226–227
- ssh-keygen command, 227–228
- TCP wrappers, 224–225
 - user-specific access, 223–224
- SSL (Secure Sockets Layer), 232, 243
- standard user accounts, 211
- Stanza-editing screen (GRUB), 4
- stat command, 135
- stateful/stateless rules, firewalls, 258–259
- statistics, iostat command, 48–49
- sticky bit permissions, 194–195
- sticky keys (keyboards), 192
- storage
 - blobs, 64
 - blocks, 64
 - cloud-computing, 63–64
 - commands, 44–53
 - configuring, 287–289
 - degraded storage, 314
 - device mappers, 41–44
 - files, 311
 - files/directories, 54–57
 - filesystems, 40–41
 - integrity, troubleshooting, 317
 - missing devices, 315
 - monitoring, 287–289
 - mount points, 315
 - off-site/off-system storage, 274–275
 - partitions, 38–39
 - performance, troubleshooting, 315
 - PV, 64
 - resource exhaustion, 316
 - thick provisioning, 63
 - thin provisioning, 63–64
 - troubleshooting, 314–317
 - virtualization, 63–64
- su command, 209
- sudo command, 209–210
- sudocedit command, 210–211
- suid permissions, 194–195
- swapoff command, 292

- swapon command, 292
- symbolic (soft) links, 132–133
- syncing repositories, 89
- /sys filesystem, 56, 182
- /sys/block, 56
- /sys/bus, 56
- /sys/bus/cpu, 56
- /sys/bus/cpu/devices, 56
- syslog, 249–250, 252–253
- syslogd command, 249–250
- systemctl command, 145–148
- Systemd management
 - boot process, 144–145
 - directories, 149–150
 - systemctl command, 145–148
 - systemd-analyze command, 147–148
 - targets, 144–145
 - unit files, 148–149
- systemd-analyze command, 147–148
- systemd-journald service, 254
- SysVinit, 152–153
 - chkconfig command, 153–154
 - /etc/rc.local files, 155
 - runlevel command, 155
 - runlevels, 154–155
 - service command, 155–156
 - telinit command, 155

T

- t command, 46
- TACACS+ (Terminal Access Controller Access-Control System Plus), 238
- tail command, 115–116, 118–119
- tar command, 268
- .tar extension, 78
- targets
 - Accept target, firewalls, 259
 - Drop target, firewalls, 259
 - Log target, firewalls, 259
 - Reject target, firewalls, 259
 - Systemd management, 144–145
- TCP wrappers, 224–225

- `tcpdump` command, 282
- `tee` command, 122
- `telinit` command, 155
- telnet servers, security, 242
- templates
 - container images, 62
 - JSON, 61
 - OVA, 60
 - OVF, 61
 - VM, 60–62
 - YAML, 61–62
- test statements (BASH), 338–339
- text
 - processing
 - `awk` command, 125–126
 - `cut` command, 127–128
 - `echo` command, 124
 - `egrep` command, 128
 - `grep` command, 118
 - `sed` command, 126–127
 - `sort` command, 124–125
 - `tr` command, 117, 123–124
 - `wc` command, 128
 - screen magnifiers, 192
 - screen readers, 192
- text editors
 - nano editor, 110–111
 - vi editor, 111–114
- .tgz extension, 78
- thick provisioning, 63
- thin provisioning, 63–64
- throughput, network latency, 279
- `time` command, 302–303
- `timedatectl` command, 71–72
- timeouts, 279
- timestamps
 - log files, 158
 - searches, 158
 - transaction logs, 158
- TLS (Transport Layer Security), 232, 243
- toggle keys (keyboards), 192
- tokens, multifactor authentication, 237
- `top` command, 301–302
- `touch` command, 129
- `tr` command, 117, 123–124
- `tracpath` command, 287
- `traceroute` command, 285
- transaction logs, timestamps, 158
- transport mode (VPN), 232
- troubleshooting
 - ACL, 318
 - adapters (network), 316
 - bad blocks (storage), 317
 - blocked ports (firewalls), 318
 - blocked protocols (firewalls), 318
 - communications ports, 321–322
 - compatibility (hardware/software), 324
 - dependencies, 319–320
 - firewalls, 317–318
 - GPU drivers, 321
 - hardware issues, 320–325
 - HBA, 317
 - keyboards, 324
 - memory, 320
 - permissions, 318–319
 - ports, 318, 321–322
 - printers, 320–321
 - protocols, 318
 - RAID devices, 317
 - SATA, 317
 - SCSI, 316
 - SELinux context violations, 314
 - software, 319–320, 324
 - storage, 314–317
 - updates (software), 320
 - USB devices, 322–324
 - versioning (software), 320
 - video devices, 321
- `tshark` command, 283–284
- TTY, 228
 - /dev/tty# files, 229
 - /etc/securetty files, 229
- `tty` command, 121
- `tune2fs` command, 53
- tunnel mode (VPN), 232
- TZ variable, 74–75

U

udev

- `/etc/udev/rules.d`, 186
- hot pluggable devices, 185–186
- `/run/udev/rules.d` (volatile rules), 185
- `/usr/lib/udev/rules.d` (system rules -lowest priority), 185

udevadm command, 181–182

UEFI (Unified Extensible Firmware Interface), 4, 236–237

ufw, 262

ulimit command, 201–202

umask command, 195–196

umount command, 50–51

Unicode character sets, 75

uninterruptible sleep state (processes), 298

unit files

- service management, 150
- Systemd management, 148–149

Unity, 189

Unix sockets, 280

unlink command, 135

until loops (BASH), 338

unused/unsecure services, disabling/
uninstalling, 242–243

updatedb command, 140

updating software, 320

uptime command, 291

USB devices, 174

- security, 243–244
- troubleshooting, 322–324

user management

- access, 309
- accounts, shell variables, 312
- authentication, 309–310
- BASH parameters, 103–106
- creating accounts, 94
- deleting accounts, 95–96
- `/etc/group` files, 107
- `/etc/passwd` files, 106–107
- `/etc/shadow` files, 107–108
- modifying accounts, 94–95
- permissions, 307–308

privilege escalation, 209–211

privileges, 310–312

profiles, 102–103

queries, 96–98

quotas, 98–102

root accounts, 211

service accounts, 211–212

single user mode, 295

standard accounts, 211

useradd command, 94

userdel command, 96

usermod command, 94–95

`/usr/lib/modules`, 16

`/usr/lib/modules/[kernelversion]15–16`

`/usr/lib/udev/rules.d` (system rules -lowest priority), 185

`/usr/share/doc` directory, 32

`/usr/share/zoneinfo`, 70

usrquota mount option, 98–99

UTF-8 character sets, 75

UUID, blkid command, 51

V

`/var/log/[application]` files, 248

`/var/log/journal` directory, 254

`/var/log/kern.log` files, 249

`/var/log/messages` files, 248

`/var/log/secure` files, 247–248

versioning (software), troubleshooting, 320

vgdisplay command, 45

vgextend command, 45

vgreduce command, 45

vgremove command, 45

vi editor, 111–114

video devices, 175, 321

virsh, 67

virtual filesystems, 40–41

virtualization

blobs, 64

blocks, 64

bootstrapping, 62–63

bridging, 65

dual-homed systems, 66

hypervisors, 66

- libvirt, 67
- local networks, 65–66
- NAT, 65
- networks, 65–66
- overlay networks, 65
- PV, 64
- storage, 63–64
- templates, 60–62
- thick provisioning, 63
- thin provisioning, 63–64
- virsh, 67
- VMM, 67
- visudo command, 210
- VM (Virtual Machines), 162
 - container images, 62
 - JSON, 61
 - OVA, 60
 - OVF, 61
 - templates, 60–62
 - VMM, 67
 - YAML, 61–62
- vmlinux, 9
- vmlinuz, 9
- VMM (Virtual Machine Manager), 67
- vmstat command, 293–295
- VNC (Virtual Network Computing), 190–191
- volumes
 - groups, 42–43
 - logical volumes, 43–44, 52
 - LVM, 41–44
 - mount points, troubleshooting, 315
 - physical volumes, 42
 - PV, 64
 - resize2fs command, 52
- VPN (Virtual Private Networks), 162, 231–232
 - DTLS, 233
 - IPsec, 232–233
 - SSL, 232
 - TLS, 232
 - transport mode, 232
 - tunnel mode, 232

W

- w command, 46, 47, 97–98
- Wayland, 188
- wc command, 128
- Web (World Wide), 159
- wget command, 89–91
- wheel groups, 210
- whereis command, 139
- which command, 139
- while loops (BASH), 337
- who command, 97
- whoami command, 96–97
- whois command, 287
- WiFi devices, 174
- wildcards (file globs), 334
- wireshark command, 283–284
- write permission, 194

X

- X11, 188, 191
- xargs command, 121–122
- xfs filesystems, 58
- XFS tools, 44
- xfs_info command, 44
- xfs_metadump command, 44
- XRDP, 190
- xz command, 270–271

Y

- YAML, 61–62
- yum command, 83–86
- yum list command, 83
- yumdownloader command, 84

Z

- zip command, 271–272
- zombie processes, 298
- zones, firewalls, 260
- zypper command, 86