



Red Hat

Red Hat Enterprise Linux 8

System Design Guide

Designing a RHEL 8 system

Red Hat Enterprise Linux 8 System Design Guide

Designing a RHEL 8 system

Legal Notice

Copyright © 2020 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux® is the registered trademark of Linus Torvalds in the United States and other countries.

Java® is a registered trademark of Oracle and/or its affiliates.

XFS® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This content covers how to start using Red Hat Enterprise Linux 8. To learn about Red Hat Enterprise Linux technology capabilities and limits, see <https://access.redhat.com/articles/rhel-limits>.

Table of Contents

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION	32
PART I. DESIGN OF INSTALLATION	33
CHAPTER 1. INTRODUCTION	34
1.1. SUPPORTED ARCHITECTURES	34
1.2. INSTALLATION TERMINOLOGY	34
CHAPTER 2. QUICK INSTALLATION	35
2.1. AVAILABLE INSTALLATION METHODS	35
2.2. INSTALLING RHEL USING AN ISO IMAGE FROM THE CUSTOMER PORTAL	35
2.3. REGISTERING AND INSTALLING RHEL FROM THE CDN USING THE GUI	37
2.3.1. What is the Content Delivery Network	37
2.3.2. Registering and installing RHEL from the CDN	38
2.3.2.1. Installation source repository after system registration	41
2.3.3. Verifying your system registration from the CDN	41
2.3.4. Unregistering your system from the CDN	42
CHAPTER 3. PREPARING FOR YOUR INSTALLATION	45
3.1. RECOMMENDED STEPS	45
3.2. SYSTEM REQUIREMENTS	45
3.3. INSTALLATION BOOT MEDIA OPTIONS	45
3.4. TYPES OF INSTALLATION ISO IMAGES	46
3.5. DOWNLOADING THE INSTALLATION ISO IMAGE	47
3.5.1. Downloading an ISO image from the Customer Portal	47
3.5.2. Downloading an ISO image using curl	47
3.6. CREATING A BOOTABLE INSTALLATION MEDIUM	48
3.6.1. Creating a bootable DVD or CD	48
3.6.2. Creating a bootable USB device on Linux	49
3.6.3. Creating a bootable USB device on Windows	50
3.6.4. Creating a bootable USB device on Mac OS X	51
3.7. PREPARING AN INSTALLATION SOURCE	53
3.7.1. Types of installation source	53
3.7.2. Specify the installation source	54
3.7.3. Ports for network-based installation	54
3.7.4. Creating an installation source on an NFS server	55
3.7.5. Creating an installation source using HTTP or HTTPS	56
3.7.6. Creating an installation source using FTP	58
CHAPTER 4. BOOTING THE INSTALLATION	61
4.1. BOOT MENU	61
4.2. TYPES OF BOOT OPTIONS	62
4.3. EDITING BOOT OPTIONS	63
Editing the boot: prompt in BIOS	63
Editing the > prompt	63
Editing the GRUB2 menu	64
4.4. BOOTING THE INSTALLATION FROM A USB, CD, OR DVD	64
4.5. BOOTING THE INSTALLATION FROM A NETWORK USING PXE	65
CHAPTER 5. CUSTOMIZING YOUR INSTALLATION USING THE GUI	67
5.1. GRAPHICAL INSTALLATION WORKFLOW	67
5.2. CONFIGURING LANGUAGE AND LOCATION SETTINGS	67
5.3. THE INSTALLATION SUMMARY WINDOW	68

5.4. CONFIGURING LOCALIZATION OPTIONS	69
5.4.1. Configuring keyboard, language, and time and date settings	69
5.5. CONFIGURING SYSTEM OPTIONS	71
5.5.1. Configuring installation destination	71
5.5.1.1. Configuring boot loader	74
5.5.2. Configuring Kdump	75
5.5.3. Configuring network and host name options	76
5.5.3.1. Adding a virtual network interface	77
5.5.3.2. Editing network interface configuration	78
5.5.3.3. Enabling or Disabling the Interface Connection	78
5.5.3.4. Setting up Static IPv4 or IPv6 Settings	79
5.5.3.5. Configuring Routes	79
5.5.3.6. Additional resources	80
5.5.4. Configuring Connect to Red Hat	80
5.5.4.1. Introduction to System Purpose	80
5.5.4.2. Configuring Connect to Red Hat options	81
5.5.4.3. Installation source repository after system registration	83
5.5.4.4. Verifying your system registration from the CDN	83
5.5.4.5. Unregistering your system from the CDN	84
5.5.5. Configuring Security Policy	85
5.5.5.1. About security policy	85
5.5.5.2. Configuring a security policy	86
5.5.5.3. Related information	86
5.6. CONFIGURING SOFTWARE OPTIONS	86
5.6.1. Configuring installation source	87
5.6.2. Configuring software selection	89
5.7. CONFIGURING STORAGE DEVICES	90
5.7.1. Storage device selection	91
5.7.2. Filtering storage devices	91
5.7.3. Using advanced storage options	92
5.7.3.1. Discovering and starting an iSCSI session	92
5.7.3.2. Configuring FCoE parameters	94
5.7.3.3. Configuring DASD storage devices	95
5.7.3.4. Configuring FCP devices	95
5.7.4. Installing to an NVDIMM device	96
5.7.4.1. Criteria for using an NVDIMM device as an installation target	97
5.7.4.2. Configuring an NVDIMM device using the graphical installation mode	97
5.8. CONFIGURING MANUAL PARTITIONING	98
5.8.1. Starting manual partitioning	99
5.8.2. Adding a mount point file system	100
5.8.3. Configuring a mount point file system	101
5.8.4. Customizing a partition or volume	101
5.8.5. Preserving the /home directory	104
5.8.6. Creating software RAID	105
5.8.7. Creating an LVM logical volume	106
5.8.8. Configuring an LVM logical volume	107
5.9. STARTING THE INSTALLATION PROGRAM	108
5.9.1. Beginning installation	108
5.9.2. Configuring a root password	108
5.9.3. Creating a user account	109
5.9.3.1. Editing advanced user settings	110
5.9.4. Graphical installation complete	111

CHAPTER 6. COMPLETING POST-INSTALLATION TASKS	112
6.1. COMPLETING INITIAL SETUP	112
6.2. REGISTERING YOUR SYSTEM USING THE COMMAND LINE	114
6.3. REGISTERING YOUR SYSTEM USING THE SUBSCRIPTION MANAGER USER INTERFACE	115
6.4. REGISTRATION ASSISTANT	116
6.5. CONFIGURING SYSTEM PURPOSE USING THE SYSPURPOSE COMMAND-LINE TOOL	116
6.6. SECURING YOUR SYSTEM	118
6.7. DEPLOYING SYSTEMS THAT ARE COMPLIANT WITH A SECURITY PROFILE RIGHT AFTER AN INSTALLATION	119
6.7.1. Deploying OSPP-compliant RHEL systems using the graphical installation	119
6.7.2. Deploying OSPP-compliant RHEL systems using Kickstart	120
APPENDIX A. TROUBLESHOOTING	122
A.1. TROUBLESHOOTING AT THE START OF THE INSTALLATION PROCESS	122
A.1.1. Dracut	122
A.1.2. Using installation log files	122
A.1.2.1. Creating pre-installation log files	123
A.1.2.2. Transferring installation log files to a USB drive	123
A.1.2.3. Transferring installation log files over the network	124
A.1.3. Detecting memory faults using the Memtest86 application	125
A.1.3.1. Running Memtest86	125
A.1.4. Verifying boot media	126
A.1.5. Consoles and logging during installation	126
A.1.6. Saving screenshots	127
A.1.7. Resuming an interrupted download attempt	127
A.1.8. Cannot boot into the graphical installation	128
A.2. TROUBLESHOOTING DURING THE INSTALLATION	129
A.2.1. Disks are not detected	129
A.2.2. Reporting error messages to Red Hat Customer Support	130
A.2.3. Partitioning issues for IBM Power Systems	131
A.3. TROUBLESHOOTING AFTER INSTALLATION	131
A.3.1. Cannot boot with a RAID card	131
A.3.2. Graphical boot sequence is not responding	131
A.3.3. X server fails after log in	132
A.3.4. RAM is not recognized	133
A.3.5. System is displaying signal 11 errors	133
A.3.6. Unable to IPL from network storage space	134
A.3.7. Using XDMCP	134
A.3.8. Using rescue mode	135
A.3.8.1. Booting into rescue mode	136
A.3.8.2. Using an SOS report in rescue mode	137
A.3.8.3. Reinstalling the GRUB2 boot loader	138
A.3.8.4. Using RPM to add or remove a driver	139
A.3.9. ip= boot option returns an error	140
APPENDIX B. SYSTEM REQUIREMENTS REFERENCE	142
B.1. HARDWARE COMPATIBILITY	142
B.2. SUPPORTED INSTALLATION TARGETS	142
B.3. SYSTEM SPECIFICATIONS	142
B.4. DISK AND MEMORY REQUIREMENTS	143
B.5. RAID REQUIREMENTS	144
APPENDIX C. PARTITIONING REFERENCE	146
C.1. SUPPORTED DEVICE TYPES	146

C.2. SUPPORTED FILE SYSTEMS	146
C.3. SUPPORTED RAID TYPES	147
C.4. RECOMMENDED PARTITIONING SCHEME	148
C.5. ADVICE ON PARTITIONS	150
APPENDIX D. BOOT OPTIONS REFERENCE	153
D.1. INSTALLATION SOURCE BOOT OPTIONS	153
D.2. NETWORK BOOT OPTIONS	157
D.3. CONSOLE BOOT OPTIONS	159
D.4. DEBUG BOOT OPTIONS	162
D.5. STORAGE BOOT OPTIONS	163
D.6. DEPRECATED BOOT OPTIONS	164
D.7. REMOVED BOOT OPTIONS	165
APPENDIX E. CHANGING A SUBSCRIPTION SERVICE	167
E.1. UNREGISTERING FROM SUBSCRIPTION MANAGEMENT SERVER	167
E.1.1. Unregistering using command line	167
E.1.2. Unregistering using Subscription Manager user interface	168
E.2. UNREGISTERING FROM SATELLITE SERVER	168
APPENDIX F. ISCSI DISKS IN INSTALLATION PROGRAM	169
CHAPTER 7. COMPOSING A CUSTOMIZED RHEL SYSTEM IMAGE	170
CHAPTER 8. IMAGE BUILDER DESCRIPTION	171
8.1. INTRODUCTION TO IMAGE BUILDER	171
8.2. IMAGE BUILDER TERMINOLOGY	171
8.3. IMAGE BUILDER OUTPUT FORMATS	171
8.4. IMAGE BUILDER SYSTEM REQUIREMENTS	172
CHAPTER 9. INSTALLING IMAGE BUILDER	173
9.1. IMAGE BUILDER SYSTEM REQUIREMENTS	173
9.2. INSTALLING IMAGE BUILDER IN A VIRTUAL MACHINE	173
CHAPTER 10. CREATING SYSTEM IMAGES WITH IMAGE BUILDER COMMAND-LINE INTERFACE	175
10.1. IMAGE BUILDER COMMAND-LINE INTERFACE	175
10.2. CREATING AN IMAGE BUILDER BLUEPRINT WITH COMMAND-LINE INTERFACE	175
10.3. EDITING AN IMAGE BUILDER BLUEPRINT WITH COMMAND-LINE INTERFACE	176
10.4. CREATING A SYSTEM IMAGE WITH IMAGE BUILDER IN THE COMMAND-LINE INTERFACE	177
10.5. BASIC IMAGE BUILDER COMMAND-LINE COMMANDS	178
10.6. IMAGE BUILDER BLUEPRINT FORMAT	179
10.7. SUPPORTED IMAGE CUSTOMIZATIONS	180
10.8. INSTALLED PACKAGES	184
10.9. ENABLED SERVICES	184
10.10. DISKS AND PARTITIONS CONFIGURATION USING IMAGE BUILDER	185
CHAPTER 11. CREATING SYSTEM IMAGES WITH IMAGE BUILDER WEB CONSOLE INTERFACE	186
11.1. ACCESSING IMAGE BUILDER GUI IN THE RHEL 8 WEB CONSOLE	186
11.2. CREATING AN IMAGE BUILDER BLUEPRINT IN THE WEB CONSOLE INTERFACE	186
11.3. EDITING AN IMAGE BUILDER BLUEPRINT IN THE WEB CONSOLE INTERFACE	187
11.4. ADDING USERS AND GROUPS TO AN IMAGE BUILDER BLUEPRINT IN THE WEB CONSOLE INTERFACE	189
11.5. CREATING A SYSTEM IMAGE WITH IMAGE BUILDER IN THE WEB CONSOLE INTERFACE	191
11.6. ADDING A SOURCE TO A BLUEPRINT	191
11.7. CREATING A USER ACCOUNT FOR A BLUEPRINT	193

11.8. CREATING A USER ACCOUNT WITH SSH KEY	194
CHAPTER 12. PREPARING AND UPLOADING CLOUD IMAGES WITH IMAGE BUILDER	198
12.1. PREPARING FOR UPLOADING AWS AMI IMAGES	198
12.2. UPLOADING AN AMI IMAGE TO AWS	199
12.3. PREPARING FOR UPLOADING AZURE VHD IMAGES	200
12.4. UPLOADING VHD IMAGES TO AZURE	202
12.5. UPLOADING VMDK IMAGES TO VS SPHERE	203
12.6. UPLOADING QCOW2 IMAGE TO OPENSTACK	205
12.7. PREPARING FOR UPLOADING IMAGES TO ALIBABA	207
12.8. UPLOADING IMAGES TO ALIBABA	208
12.9. IMPORTING IMAGES TO ALIBABA	209
12.10. CREATING AN INSTANCE OF A CUSTOM IMAGE USING ALIBABA	210
CHAPTER 13. PERFORMING AN AUTOMATED INSTALLATION USING KICKSTART	212
CHAPTER 14. KICKSTART INSTALLATION BASICS	213
14.1. WHAT ARE KICKSTART INSTALLATIONS	213
14.2. AUTOMATED INSTALLATION WORKFLOW	213
CHAPTER 15. CREATING KICKSTART FILES	215
15.1. CREATING A KICKSTART FILE WITH THE KICKSTART CONFIGURATION TOOL	215
15.2. CREATING A KICKSTART FILE BY PERFORMING A MANUAL INSTALLATION	215
15.3. CONVERTING A RHEL 7 KICKSTART FILE FOR RHEL 8 INSTALLATION	216
CHAPTER 16. MAKING KICKSTART FILES AVAILABLE TO THE INSTALLATION PROGRAM	217
16.1. PORTS FOR NETWORK-BASED INSTALLATION	217
16.2. MAKING A KICKSTART FILE AVAILABLE ON AN NFS SERVER	217
16.3. MAKING A KICKSTART FILE AVAILABLE ON AN HTTP OR HTTPS SERVER	218
16.4. MAKING A KICKSTART FILE AVAILABLE ON AN FTP SERVER	219
16.5. MAKING A KICKSTART FILE AVAILABLE ON A LOCAL VOLUME	221
16.6. MAKING A KICKSTART FILE AVAILABLE ON A LOCAL VOLUME FOR AUTOMATIC LOADING	222
CHAPTER 17. CREATING INSTALLATION SOURCES FOR KICKSTART INSTALLATIONS	224
17.1. TYPES OF INSTALLATION SOURCE	224
17.2. PORTS FOR NETWORK-BASED INSTALLATION	224
17.3. CREATING AN INSTALLATION SOURCE ON AN NFS SERVER	225
17.4. CREATING AN INSTALLATION SOURCE USING HTTP OR HTTPS	226
17.5. CREATING AN INSTALLATION SOURCE USING FTP	228
CHAPTER 18. STARTING KICKSTART INSTALLATIONS	231
18.1. STARTING A KICKSTART INSTALLATION MANUALLY	231
18.2. STARTING A KICKSTART INSTALLATION AUTOMATICALLY USING PXE	231
18.3. STARTING A KICKSTART INSTALLATION AUTOMATICALLY USING A LOCAL VOLUME	232
CHAPTER 19. CONSOLES AND LOGGING DURING INSTALLATION	234
CHAPTER 20. MAINTAINING KICKSTART FILES	235
20.1. INSTALLING KICKSTART MAINTENANCE TOOLS	235
20.2. VERIFYING A KICKSTART FILE	235
CHAPTER 21. REGISTERING AND INSTALLING RHEL FROM THE CDN USING KICKSTART	236
21.1. REGISTERING AND INSTALLING RHEL FROM THE CDN	236
21.2. VERIFYING YOUR SYSTEM REGISTRATION FROM THE CDN	239
21.3. UNREGISTERING YOUR SYSTEM FROM THE CDN	239

CHAPTER 22. PERFORMING A REMOTE RHEL INSTALLATION USING VNC	241
22.1. OVERVIEW	241
22.2. CONSIDERATIONS	241
22.3. PERFORMING A REMOTE RHEL INSTALLATION IN VNC DIRECT MODE	242
22.4. PERFORMING A REMOTE RHEL INSTALLATION IN VNC CONNECT MODE	243
CHAPTER 23. ADVANCED CONFIGURATION OPTIONS	245
CHAPTER 24. CONFIGURING SYSTEM PURPOSE	246
24.1. OVERVIEW	246
24.2. CONFIGURING SYSTEM PURPOSE IN A KICKSTART FILE	247
24.3. RELATED INFORMATION	248
CHAPTER 25. UPDATING DRIVERS DURING INSTALLATION	249
25.1. PREREQUISITE	249
25.2. OVERVIEW	249
25.3. TYPES OF DRIVER UPDATE	249
25.4. PREPARING A DRIVER UPDATE	250
25.5. PERFORMING AN AUTOMATIC DRIVER UPDATE	251
25.6. PERFORMING AN ASSISTED DRIVER UPDATE	251
25.7. PERFORMING A MANUAL DRIVER UPDATE	252
25.8. DISABLING A DRIVER	252
CHAPTER 26. PREPARING TO INSTALL FROM THE NETWORK USING PXE	254
26.1. NETWORK INSTALL OVERVIEW	254
26.2. CONFIGURING A TFTP SERVER FOR BIOS-BASED CLIENTS	255
26.3. CONFIGURING A TFTP SERVER FOR UEFI-BASED CLIENTS	257
26.4. CONFIGURING A NETWORK SERVER FOR IBM POWER SYSTEMS	260
CHAPTER 27. BOOT OPTIONS	263
27.1. TYPES OF BOOT OPTIONS	263
27.2. EDITING BOOT OPTIONS	263
Editing the boot: prompt in BIOS	263
Editing the > prompt	264
Editing the GRUB2 menu	264
27.3. INSTALLATION SOURCE BOOT OPTIONS	265
27.4. NETWORK BOOT OPTIONS	269
27.5. CONSOLE BOOT OPTIONS	271
27.6. DEBUG BOOT OPTIONS	274
27.7. STORAGE BOOT OPTIONS	275
27.8. KICKSTART BOOT OPTIONS	276
27.9. ADVANCED INSTALLATION BOOT OPTIONS	277
27.10. DEPRECATED BOOT OPTIONS	278
27.11. REMOVED BOOT OPTIONS	279
CHAPTER 28. KICKSTART REFERENCES	281
APPENDIX G. KICKSTART SCRIPT FILE FORMAT REFERENCE	282
G.1. KICKSTART FILE FORMAT	282
G.2. PACKAGE SELECTION IN KICKSTART	283
G.2.1. Package selection section	283
G.2.2. Package selection commands	283
G.2.3. Common package selection options	285
G.2.4. Options for specific package groups	286
G.3. SCRIPTS IN KICKSTART FILE	287

G.3.1. %pre script	287
G.3.1.1. %pre script section options	288
G.3.2. %pre-install script	288
G.3.2.1. %pre-install script section options	288
G.3.3. %post script	289
G.3.3.1. %post script section options	289
G.3.3.2. Example: Mounting NFS in a post-install script	290
G.3.3.3. Example: Running subscription-manager as a post-install script	291
G.4. ANACONDA CONFIGURATION SECTION	291
G.5. KICKSTART ERROR HANDLING SECTION	292
G.6. KICKSTART ADD-ON SECTIONS	292
APPENDIX H. KICKSTART COMMANDS AND OPTIONS REFERENCE	293
H.1. KICKSTART CHANGES	293
H.1.1. auth or authconfig is deprecated in RHEL 8	293
H.1.2. Kickstart no longer supports Btrfs	293
H.1.3. Using Kickstart files from previous RHEL releases	293
H.1.4. Deprecated Kickstart commands and options	293
H.1.5. Removed Kickstart commands and options	294
H.1.6. New Kickstart commands and options	294
H.2. KICKSTART COMMANDS FOR INSTALLATION PROGRAM CONFIGURATION AND FLOW CONTROL	294
H.2.1. autostep	295
H.2.2. cdrom	295
H.2.3. cmdline	295
H.2.4. driverdisk	296
H.2.5. eula	297
H.2.6. firstboot	297
H.2.7. graphical	297
H.2.8. halt	298
H.2.9. harddrive	298
H.2.10. install (deprecated)	299
H.2.11. liveimg	299
H.2.12. logging	300
H.2.13. mediacheck	301
H.2.14. nfs	301
H.2.15. ostreesetup	302
H.2.16. poweroff	302
H.2.17. reboot	303
H.2.18. rhsm	303
H.2.19. shutdown	304
H.2.20. sshpw	304
H.2.21. text	305
H.2.22. url	306
H.2.23. vnc	307
H.2.24. %include	307
H.2.25. %ksappend	307
H.3. KICKSTART COMMANDS FOR SYSTEM CONFIGURATION	308
H.3.1. auth or authconfig (deprecated)	308
H.3.2. authselect	308
H.3.3. firewall	309
H.3.4. group	310
H.3.5. keyboard (required)	310

H.3.6. lang (required)	311
H.3.7. module	311
H.3.8. repo	312
H.3.9. rootpw (required)	313
H.3.10. selinux	314
H.3.11. services	314
H.3.12. skipx	315
H.3.13. sshkey	315
H.3.14. syspurpose	315
H.3.15. timezone (required)	316
H.3.16. user	317
H.3.17. xconfig	318
H.4. KICKSTART COMMANDS FOR NETWORK CONFIGURATION	318
H.4.1. network	318
H.4.2. realm	322
H.5. KICKSTART COMMANDS FOR HANDLING STORAGE	323
H.5.1. device (deprecated)	323
H.5.2. autopart	324
H.5.3. bootloader (required)	325
H.5.4. zipl	328
H.5.5. clearpart	329
H.5.6. fcoe	330
H.5.7. ignoredisk	331
H.5.8. iscsi	332
H.5.9. iscsiname	333
H.5.10. logvol	333
H.5.11. mount	337
H.5.12. nvdimm	338
H.5.13. part or partition	339
H.5.14. raid	344
H.5.15. reqpart	346
H.5.16. snapshot	347
H.5.17. volgroup	347
H.5.18. zerombr	348
H.5.19. zfcp	349
H.6. KICKSTART COMMANDS FOR ADDONS SUPPLIED WITH THE RHEL INSTALLATION PROGRAM	349
H.6.1. %addon com_redhat_kdump	349
H.6.2. %addon org_fedora_oscap	350
H.7. COMMANDS USED IN ANACONDA	352
H.7.1. pwpolicy	352
H.8. KICKSTART COMMANDS FOR SYSTEM RECOVERY	353
H.8.1. rescue	353
PART II. DESIGN OF SECURITY	355
CHAPTER 29. OVERVIEW OF SECURITY HARDENING IN RHEL	356
29.1. WHAT IS COMPUTER SECURITY?	356
29.2. STANDARDIZING SECURITY	356
29.3. CRYPTOGRAPHIC SOFTWARE AND CERTIFICATIONS	356
29.4. SECURITY CONTROLS	357
29.4.1. Physical controls	357
29.4.2. Technical controls	357
29.4.3. Administrative controls	358

29.5. VULNERABILITY ASSESSMENT	358
29.5.1. Defining assessment and testing	358
29.5.2. Establishing a methodology for vulnerability assessment	360
29.5.3. Vulnerability assessment tools	360
29.6. SECURITY THREATS	360
29.6.1. Threats to network security	360
29.6.2. Threats to server security	361
29.6.3. Threats to workstation and home PC security	362
29.7. COMMON EXPLOITS AND ATTACKS	363
CHAPTER 30. SECURING RHEL DURING INSTALLATION	367
30.1. BIOS AND UEFI SECURITY	367
30.1.1. BIOS passwords	367
30.1.1.1. Non-BIOS-based systems security	367
30.2. DISK PARTITIONING	367
30.3. RESTRICTING NETWORK CONNECTIVITY DURING THE INSTALLATION PROCESS	368
30.4. INSTALLING THE MINIMUM AMOUNT OF PACKAGES REQUIRED	368
30.5. POST-INSTALLATION PROCEDURES	368
CHAPTER 31. USING SYSTEM-WIDE CRYPTOGRAPHIC POLICIES	370
31.1. SYSTEM-WIDE CRYPTOGRAPHIC POLICIES	370
Tool for managing crypto policies	370
Strong crypto defaults by removing insecure cipher suites and protocols	371
Cipher suites and protocols disabled in all policy levels	371
Cipher suites and protocols enabled in the crypto-policies levels	371
31.2. SWITCHING THE SYSTEM-WIDE CRYPTOGRAPHIC POLICY TO MODE COMPATIBLE WITH EARLIER RELEASES	372
31.3. SWITCHING THE SYSTEM TO FIPS MODE	373
31.4. ENABLING FIPS MODE IN A CONTAINER	374
31.5. EXCLUDING AN APPLICATION FROM FOLLOWING SYSTEM-WIDE CRYPTO POLICIES	374
31.5.1. Examples of opting out of system-wide crypto policies	374
31.6. CUSTOMIZING SYSTEM-WIDE CRYPTOGRAPHIC POLICIES WITH POLICY MODIFIERS	375
31.7. CREATING AND SETTING A CUSTOM SYSTEM-WIDE CRYPTOGRAPHIC POLICY	376
31.8. RELATED INFORMATION	377
CHAPTER 32. CONFIGURING APPLICATIONS TO USE CRYPTOGRAPHIC HARDWARE THROUGH PKCS #11 ..	378
32.1. CRYPTOGRAPHIC HARDWARE SUPPORT THROUGH PKCS #11	378
32.2. USING SSH KEYS STORED ON A SMART CARD	378
32.3. USING HSMS PROTECTING PRIVATE KEYS IN APACHE AND NGINX	380
32.4. CONFIGURING APPLICATIONS TO AUTHENTICATE USING CERTIFICATES FROM SMART CARDS	380
32.5. RELATED INFORMATION	381
CHAPTER 33. USING SHARED SYSTEM CERTIFICATES	382
33.1. THE SYSTEM-WIDE TRUST STORE	382
33.2. ADDING NEW CERTIFICATES	382
33.3. MANAGING TRUSTED SYSTEM CERTIFICATES	383
33.4. RELATED INFORMATION	384
CHAPTER 34. SCANNING THE SYSTEM FOR SECURITY COMPLIANCE AND VULNERABILITIES	385
34.1. SECURITY COMPLIANCE TOOLS IN RHEL	385
34.2. RED HAT SECURITY ADVISORIES OVAL FEED	385
34.3. SCANNING THE SYSTEM FOR VULNERABILITIES	386
34.4. SCANNING REMOTE SYSTEMS FOR VULNERABILITIES	387

34.5. VIEWING PROFILES FOR SECURITY COMPLIANCE	388
34.6. ASSESSING SECURITY COMPLIANCE WITH A SPECIFIC BASELINE	389
34.7. REMEDIATING THE SYSTEM TO ALIGN WITH OSPP	390
34.8. SCANNING THE SYSTEM WITH A CUSTOMIZED PROFILE USING SCAP WORKBENCH	391
34.8.1. Using SCAP Workbench to scan and remediate the system	391
34.8.2. Customizing a security profile with SCAP Workbench	392
34.8.3. Related information	394
CHAPTER 35. CHECKING INTEGRITY WITH AIDE	395
35.1. INSTALLING AIDE	395
35.2. PERFORMING INTEGRITY CHECKS WITH AIDE	395
35.3. UPDATING AN AIDE DATABASE	396
35.4. RELATED INFORMATION	396
CHAPTER 36. ENCRYPTING BLOCK DEVICES USING LUKS	397
36.1. LUKS DISK ENCRYPTION	397
36.2. LUKS VERSIONS IN RHEL 8	398
36.3. OPTIONS FOR DATA PROTECTION DURING LUKS2 RE-ENCRYPTION	399
36.4. ENCRYPTING EXISTING DATA ON A BLOCK DEVICE USING LUKS2	399
36.5. ENCRYPTING EXISTING DATA ON A BLOCK DEVICE USING LUKS2 WITH A DETACHED HEADER	400
CHAPTER 37. CONFIGURING AUTOMATED UNLOCKING OF ENCRYPTED VOLUMES USING POLICY-BASED DECRYPTION	402
37.1. NETWORK-BOUND DISK ENCRYPTION	402
37.2. INSTALLING AN ENCRYPTION CLIENT - CLEVIS	403
37.3. DEPLOYING A TANG SERVER WITH SELINUX IN ENFORCING MODE	404
37.4. ROTATING TANG SERVER KEYS AND UPDATING BINDINGS ON CLIENTS	405
37.5. DEPLOYING AN ENCRYPTION CLIENT FOR AN NBDE SYSTEM WITH TANG	407
37.6. REMOVING A CLEVIS PIN FROM A LUKS-ENCRYPTED VOLUME MANUALLY	408
37.7. DEPLOYING AN ENCRYPTION CLIENT WITH A TPM 2.0 POLICY	409
37.8. CONFIGURING MANUAL ENROLLMENT OF LUKS-ENCRYPTED VOLUMES	410
37.9. CONFIGURING AUTOMATED ENROLLMENT OF LUKS-ENCRYPTED VOLUMES USING KICKSTART	412
37.10. CONFIGURING AUTOMATED UNLOCKING OF A LUKS-ENCRYPTED REMOVABLE STORAGE DEVICE	413
37.11. DEPLOYING HIGH-AVAILABILITY NBDE SYSTEMS	414
37.11.1. High-available NBDE using Shamir's Secret Sharing	414
37.11.1.1. Example 1: Redundancy with two Tang servers	414
37.11.1.2. Example 2: Shared secret on a Tang server and a TPM device	415
37.11.2. DEPLOYMENT OF VIRTUAL MACHINES IN A NBDE NETWORK	415
37.13. BUILDING AUTOMATICALLY-ENROLLABLE VM IMAGES FOR CLOUD ENVIRONMENTS USING NBDE	416
37.14. ADDITIONAL RESOURCES	416
CHAPTER 38. USING SELINUX	417
CHAPTER 39. GETTING STARTED WITH SELINUX	418
39.1. INTRODUCTION TO SELINUX	418
39.2. BENEFITS OF RUNNING SELINUX	419
39.3. SELINUX EXAMPLES	420
39.4. SELINUX ARCHITECTURE AND PACKAGES	420
39.5. SELINUX STATES AND MODES	421
CHAPTER 40. CHANGING SELINUX STATES AND MODES	423
40.1. PERMANENT CHANGES IN SELINUX STATES AND MODES	423

40.2. CHANGING TO PERMISSIVE MODE	423
40.3. CHANGING TO ENFORCING MODE	424
40.4. ENABLING SELINUX ON SYSTEMS THAT PREVIOUSLY HAD IT DISABLED	425
40.5. DISABLING SELINUX	426
40.6. CHANGING SELINUX MODES AT BOOT TIME	427
CHAPTER 41. TROUBLESHOOTING PROBLEMS RELATED TO SELINUX	429
41.1. IDENTIFYING SELINUX DENIALS	429
41.2. ANALYZING SELINUX DENIAL MESSAGES	430
41.3. FIXING ANALYZED SELINUX DENIALS	431
41.4. SELINUX DENIALS IN THE AUDIT LOG	434
41.5. RELATED INFORMATION	435
PART III. DESIGN OF NETWORK	436
CHAPTER 42. OVERVIEW OF NETWORKING TOPICS	437
42.1. IP VERSUS NON-IP NETWORKS	437
Categories of network communication	437
42.2. STATIC VERSUS DYNAMIC IP ADDRESSING	437
42.3. CONFIGURING THE DHCP CLIENT BEHAVIOR	438
Configuring the DHCP timeout	438
Lease renewal and expiration	438
42.3.1. Making DHCPv4 persistent	438
42.4. INFINIBAND AND RDMA NETWORKS	439
42.5. SETTING THE WIRELESS REGULATORY DOMAIN	439
42.6. USING NETWORK KERNEL TUNABLES WITH SYSCTL	439
42.7. MANAGING DATA USING THE NCAT UTILITY	439
Installing ncat	440
Brief selection of ncat use cases	440
CHAPTER 43. NETCONSOLE	442
43.1. CONFIGURING NETCONSOLE	442
CHAPTER 44. GETTING STARTED WITH MANAGING NETWORKING WITH NETWORKMANAGER	443
44.1. OVERVIEW OF NETWORKMANAGER	443
44.1.1. Benefits of using NetworkManager	443
44.2. INSTALLING NETWORKMANAGER	443
44.3. CHECKING THE STATUS OF NETWORKMANAGER	444
44.4. STARTING NETWORKMANAGER	444
44.5. NETWORKMANAGER TOOLS	444
44.6. RUNNING DISPATCHER SCRIPTS	445
44.7. USING NETWORKMANAGER WITH SYSCONFIG FILES	445
44.7.1. Legacy network scripts support	446
CHAPTER 45. OVERVIEW OF NETWORK CONFIGURATION METHODS	447
45.1. SELECTING NETWORK CONFIGURATION METHODS	447
CHAPTER 46. CONFIGURING IP NETWORKING WITH NMTUI	448
46.1. GETTING STARTED WITH NMTUI	448
46.1.1. Adding a connection profile using nmtui	449
46.1.2. Applying changes to a modified connection with nmtui	451
CHAPTER 47. GETTING STARTED WITH NMCLI	455
47.1. UNDERSTANDING NMCLI	455
47.2. OVERVIEW OF NMCLI PROPERTY NAMES AND ALIASES	457

47.3. BRIEF SELECTION OF NMCLI COMMANDS	459
47.4. SETTING A DEVICE MANAGED OR UNMANAGED WITH NMCLI	462
47.5. CREATING A CONNECTION PROFILE WITH NMCLI	463
47.6. USING THE NMCLI INTERACTIVE CONNECTION EDITOR	464
47.7. MODIFYING A CONNECTION PROFILE WITH NMCLI	466
CHAPTER 48. GETTING STARTED WITH CONFIGURING NETWORKING USING THE GNOME GUI	468
48.1. CONNECTING TO A NETWORK USING THE GNOME SHELL NETWORK CONNECTION ICON	468
48.2. CREATING A NETWORK CONNECTION USING CONTROL-CENTER	469
CHAPTER 49. CONFIGURING IP NETWORKING WITH IFCFG FILES	470
49.1. CONFIGURING AN INTERFACE WITH STATIC NETWORK SETTINGS USING IFCFG FILES	470
49.2. CONFIGURING AN INTERFACE WITH DYNAMIC NETWORK SETTINGS USING IFCFG FILES	470
49.3. MANAGING SYSTEM-WIDE AND PRIVATE CONNECTION PROFILES WITH IFCFG FILES	471
CHAPTER 50. GETTING STARTED WITH IPVLAN	472
50.1. IPVLAN OVERVIEW	472
50.2. IPVLAN MODES	472
50.3. OVERVIEW OF MACVLAN	472
50.4. COMPARISON OF IPVLAN AND MACVLAN	472
50.5. CONFIGURING IPVLAN NETWORK	473
50.5.1. Creating and configuring the IPVLAN device using iproute2	473
CHAPTER 51. CONFIGURING VIRTUAL ROUTING AND FORWARDING (VRF)	475
51.1. TEMPORARILY REUSING THE SAME IP ADDRESS ON DIFFERENT INTERFACES	475
51.2. RELATED INFORMATION	476
CHAPTER 52. SECURING NETWORKS	477
CHAPTER 53. USING SECURE COMMUNICATIONS BETWEEN TWO SYSTEMS WITH OPENSSH	478
53.1. SSH AND OPENSSH	478
53.2. CONFIGURING AND STARTING AN OPENSSH SERVER	479
53.3. USING KEY PAIRS INSTEAD OF PASSWORDS FOR SSH AUTHENTICATION	480
53.3.1. Setting an OpenSSH server for key-based authentication	480
53.3.2. Generating SSH key pairs	481
53.4. USING SSH KEYS STORED ON A SMART CARD	482
53.5. MAKING OPENSSH MORE SECURE	484
53.6. CONNECTING TO A REMOTE SERVER USING AN SSH JUMP HOST	486
53.7. ADDITIONAL RESOURCES	487
CHAPTER 54. PLANNING AND IMPLEMENTING TLS	489
54.1. SSL AND TLS PROTOCOLS	489
54.2. SECURITY CONSIDERATIONS FOR TLS IN RHEL 8	489
54.2.1. Protocols	490
54.2.2. Cipher suites	490
54.2.3. Public key length	490
54.3. HARDENING TLS CONFIGURATION IN APPLICATIONS	491
54.3.1. Configuring the Apache HTTP server	491
54.3.2. Configuring the Nginx HTTP and proxy server	492
54.3.3. Configuring the Dovecot mail server	492
CHAPTER 55. CONFIGURING A VPN WITH IPSEC	494
55.1. LIBRESWAN AS AN IPSEC VPN IMPLEMENTATION	494
55.2. INSTALLING LIBRESWAN	495
55.3. CREATING A HOST-TO-HOST VPN	495

55.4. CONFIGURING A SITE-TO-SITE VPN	496
55.5. CONFIGURING A REMOTE ACCESS VPN	497
55.6. CONFIGURING A MESH VPN	498
55.7. METHODS OF AUTHENTICATION USED IN LIBRESWAN	500
55.8. DEPLOYING A FIPS-COMPLIANT IPSEC VPN	501
55.9. RELATED INFORMATION	504
CHAPTER 56. CONFIGURING MACSEC	505
56.1. INTRODUCTION TO MACSEC	505
56.2. USING MACSEC WITH NMCLI TOOL	505
56.3. USING MACSEC WITH WPA_SUPPLICANT	505
56.4. RELATED INFORMATION	506
CHAPTER 57. USING AND CONFIGURING FIREWALLD	507
57.1. WHEN TO USE FIREWALLD, NFTABLES, OR IPTABLES	507
57.2. GETTING STARTED WITH FIREWALLD	507
57.2.1. firewalld	507
57.2.2. Zones	507
57.2.3. Predefined services	509
57.3. INSTALLING THE FIREWALL-CONFIG GUI CONFIGURATION TOOL	509
57.4. VIEWING THE CURRENT STATUS AND SETTINGS OF FIREWALLD	509
57.4.1. Viewing the current status of firewalld	510
57.4.2. Viewing current firewalld settings	510
57.4.2.1. Viewing allowed services using GUI	510
57.4.2.2. Viewing firewalld settings using CLI	510
57.5. STARTING FIREWALLD	512
57.6. STOPPING FIREWALLD	512
57.7. RUNTIME AND PERMANENT SETTINGS	512
57.8. VERIFYING THE PERMANENT FIREWALLD CONFIGURATION	513
57.9. CONTROLLING NETWORK TRAFFIC USING FIREWALLD	514
57.9.1. Disabling all traffic in case of emergency using CLI	514
57.9.2. Controlling traffic with predefined services using CLI	514
57.9.3. Controlling traffic with predefined services using GUI	515
57.9.4. Adding new services	515
57.9.5. Controlling ports using CLI	516
57.9.5.1. Opening a port	516
57.9.5.2. Closing a port	517
57.9.6. Opening ports using GUI	517
57.9.7. Controlling traffic with protocols using GUI	517
57.9.8. Opening source ports using GUI	518
57.10. WORKING WITH FIREWALLD ZONES	518
57.10.1. Listing zones	518
57.10.2. Modifying firewalld settings for a certain zone	518
57.10.3. Changing the default zone	518
57.10.4. Assigning a network interface to a zone	519
57.10.5. Assigning a zone to a connection using nmcli	519
57.10.6. Manually assigning a zone to a network connection in an ifcfg file	520
57.10.7. Creating a new zone	520
57.10.8. Zone configuration files	520
57.10.9. Using zone targets to set default behavior for incoming traffic	521
57.11. USING ZONES TO MANAGE INCOMING TRAFFIC DEPENDING ON A SOURCE	521
57.11.1. Using zones to manage incoming traffic depending on a source	521
57.11.2. Adding a source	521

57.11.3. Removing a source	522
57.11.4. Adding a source port	522
57.11.5. Removing a source port	522
57.11.6. Using zones and sources to allow a service for only a specific domain	523
57.11.7. Configuring traffic accepted by a zone based on a protocol	523
57.11.7.1. Adding a protocol to a zone	523
57.11.7.2. Removing a protocol from a zone	524
57.12. CONFIGURING IP ADDRESS MASQUERADING	524
57.13. PORT FORWARDING	524
57.13.1. Adding a port to redirect	525
57.13.2. Redirecting TCP port 80 to port 88 on the same machine	525
57.13.3. Removing a redirected port	525
57.13.4. Removing TCP port 80 forwarded to port 88 on the same machine	526
57.14. MANAGING ICMP REQUESTS	526
57.14.1. Listing and blocking ICMP requests	526
57.14.2. Configuring the ICMP filter using GUI	528
57.15. SETTING AND CONTROLLING IP SETS USING FIREWALLD	529
57.15.1. Configuring IP set options using CLI	529
57.16. PRIORITIZING RICH RULES	531
57.16.1. How the priority parameter organizes rules into different chains	531
57.16.2. Setting the priority of a rich rule	531
57.17. CONFIGURING FIREWALL LOCKDOWN	532
57.17.1. Configuring lockdown with using CLI	532
57.17.2. Configuring lockdown whitelist options using CLI	532
57.17.3. Configuring lockdown whitelist options using configuration files	534
57.18. LOG FOR DENIED PACKETS	535
57.19. RELATED INFORMATION	535
Installed documentation	535
Online documentation	536
CHAPTER 58. GETTING STARTED WITH NFTABLES	537
58.1. INTRODUCTION TO NFTABLES	537
58.2. WHEN TO USE FIREWALLD, NFTABLES, OR IPTABLES	537
58.3. CONVERTING IPTABLES RULES TO NFTABLES RULES	538
58.4. WRITING AND EXECUTING NFTABLES SCRIPTS	538
58.4.1. The required script header in nftables script	538
58.4.2. Supported nftables script formats	539
58.4.3. Running nftables scripts	539
58.4.4. Using comments in nftables scripts	540
58.4.5. Using variables in an nftables script	540
Variables with a single value	541
Variables that contain an anonymous set	541
58.4.6. Including files in an nftables script	541
58.4.7. Automatically loading nftables rules when the system boots	542
58.5. DISPLAYING NFTABLES RULE SETS	542
58.6. CREATING AN NFTABLES TABLE	543
58.7. CREATING AN NFTABLES CHAIN	543
58.8. ADDING A RULE TO AN NFTABLES CHAIN	544
58.9. INSERTING A RULE INTO AN NFTABLES CHAIN	545
58.10. CONFIGURING NAT USING NFTABLES	546
58.10.1. The different NAT types: masquerading, source NAT, and destination NAT	546
58.10.2. Configuring masquerading using nftables	546
58.10.3. Configuring source NAT using nftables	547

58.10.4. Configuring destination NAT using nftables	548
58.11. USING SETS IN NFTABLES COMMANDS	549
58.11.1. Using an anonymous sets in nftables	549
58.11.2. Using named sets in nftables	549
58.11.3. Related information	551
58.12. USING VERDICT MAPS IN NFTABLES COMMANDS	551
58.12.1. Using literal maps in nftables	551
58.12.2. Using mutable verdict maps in nftables	552
58.12.3. Related information	554
58.13. CONFIGURING PORT FORWARDING USING NFTABLES	554
58.13.1. Forwarding incoming packets to a different local port	554
58.13.2. Forwarding incoming packets on a specific local port to a different host	554
58.14. LIMITING THE NUMBER OF CONNECTIONS USING NFTABLES	555
58.15. BLOCKING IP ADDRESSES THAT ATTEMPT MORE THAN TEN NEW INCOMING TCP CONNECTIONS WITHIN ONE MINUTE	556
58.16. DEBUGGING NFTABLES RULES	557
58.16.1. Creating a rule with a counter	557
58.16.2. Adding a counter to an existing rule	557
58.16.3. Monitoring packets that match an existing rule	558
58.17. BACKING UP AND RESTORING NFTABLES RULE SETS	559
58.17.1. Backing up nftables rule sets to a file	559
58.17.2. Restoring nftables rule sets from a file	559
58.18. RELATED INFORMATION	560
PART IV. DESIGN OF HARD DISK	561
CHAPTER 59. OVERVIEW OF AVAILABLE FILE SYSTEMS	562
59.1. TYPES OF FILE SYSTEMS	562
59.2. LOCAL FILE SYSTEMS	562
Available local file systems	563
59.3. THE XFS FILE SYSTEM	563
Performance characteristics	564
59.4. THE EXT4 FILE SYSTEM	564
59.5. COMPARISON OF XFS AND EXT4	565
59.6. CHOOSING A LOCAL FILE SYSTEM	566
59.7. NETWORK FILE SYSTEMS	567
Available network file systems	567
59.8. SHARED STORAGE FILE SYSTEMS	567
Comparison with network file systems	567
Concurrency	567
Performance characteristics	568
Available shared storage file systems	568
59.9. CHOOSING BETWEEN NETWORK AND SHARED STORAGE FILE SYSTEMS	568
59.10. VOLUME-MANAGING FILE SYSTEMS	568
Available volume-managing file systems	568
CHAPTER 60. MOUNTING NFS SHARES	570
60.1. INTRODUCTION TO NFS	570
60.2. SUPPORTED NFS VERSIONS	570
Default NFS version	570
Features of minor NFS versions	570
60.3. SERVICES REQUIRED BY NFS	571
The RPC services with NFSv4	572
60.4. NFS HOST NAME FORMATS	572

60.5. INSTALLING NFS	573
60.6. DISCOVERING NFS EXPORTS	573
60.7. MOUNTING AN NFS SHARE WITH MOUNT	573
60.8. COMMON NFS MOUNT OPTIONS	574
60.9. RELATED INFORMATION	575
CHAPTER 61. EXPORTING NFS SHARES	577
61.1. INTRODUCTION TO NFS	577
61.2. SUPPORTED NFS VERSIONS	577
Default NFS version	577
Features of minor NFS versions	577
61.3. THE TCP AND UDP PROTOCOLS IN NFSV3 AND NFSV4	578
61.4. SERVICES REQUIRED BY NFS	578
The RPC services with NFSv4	579
61.5. NFS HOST NAME FORMATS	579
61.6. NFS SERVER CONFIGURATION	580
61.6.1. The /etc/exports configuration file	580
Export entry	580
Default options	581
Default and overridden options	582
61.6.2. The exportfs utility	582
Common exportfs options	582
61.7. NFS AND RPCBIND	583
61.8. INSTALLING NFS	583
61.9. STARTING THE NFS SERVER	583
61.10. TROUBLESHOOTING NFS AND RPCBIND	584
61.11. CONFIGURING THE NFS SERVER TO RUN BEHIND A FIREWALL	585
61.12. EXPORTING RPC QUOTA THROUGH A FIREWALL	586
61.13. ENABLING NFS OVER RDMA (NFSORDMA)	587
61.14. CONFIGURING AN NFSV4-ONLY SERVER	587
61.14.1. Benefits and drawbacks of an NFSv4-only server	587
61.14.2. NFS and rpcbind	588
61.14.3. Configuring the NFS server to support only NFSv4	588
61.14.4. Verifying the NFSv4-only configuration	589
61.15. RELATED INFORMATION	590
CHAPTER 62. MOUNTING AN SMB SHARE ON RED HAT ENTERPRISE LINUX	591
62.1. SUPPORTED SMB PROTOCOL VERSIONS	591
62.2. UNIX EXTENSIONS SUPPORT	591
62.3. MANUALLY MOUNTING AN SMB SHARE	592
62.4. MOUNTING AN SMB SHARE AUTOMATICALLY WHEN THE SYSTEM BOOTS	593
62.5. AUTHENTICATING TO AN SMB SHARE USING A CREDENTIALS FILE	593
62.6. PERFORMING A MULTI-USER SMB MOUNT	594
62.6.1. Mounting a share with the multiuser option	594
62.6.2. Verifying if an SMB share is mounted with the multiuser option	595
62.6.3. Accessing a share as a user	595
62.7. FREQUENTLY USED MOUNT OPTIONS	595
CHAPTER 63. OVERVIEW OF PERSISTENT NAMING ATTRIBUTES	597
63.1. DISADVANTAGES OF NON-PERSISTENT NAMING ATTRIBUTES	597
63.2. FILE SYSTEM AND DEVICE IDENTIFIERS	597
File system identifiers	598
Device identifiers	598
Recommendations	598

63.3. DEVICE NAMES MANAGED BY THE UDEV MECHANISM IN /DEV/DISK/	598
63.3.1. File system identifiers	598
The UUID attribute in /dev/disk/by-uuid/	598
The Label attribute in /dev/disk/by-label/	599
63.3.2. Device identifiers	599
The WWID attribute in /dev/disk/by-id/	599
The Partition UUID attribute in /dev/disk/by-partuuid	600
The Path attribute in /dev/disk/by-path/	600
63.4. THE WORLD WIDE IDENTIFIER WITH DM MULTIPATH	600
63.5. LIMITATIONS OF THE UDEV DEVICE NAMING CONVENTION	601
63.6. LISTING PERSISTENT NAMING ATTRIBUTES	601
63.7. MODIFYING PERSISTENT NAMING ATTRIBUTES	603
CHAPTER 64. GETTING STARTED WITH PARTITIONS	604
64.1. VIEWING THE PARTITION TABLE	604
64.1.1. Viewing the partition table with parted	604
64.1.2. Example output of parted print	604
64.2. CREATING A PARTITION TABLE ON A DISK	605
64.2.1. Considerations before modifying partitions on a disk	605
The maximum number of partitions	606
The maximum size of a partition	606
Size alignment	606
64.2.2. Comparison of partition table types	606
64.2.3. Creating a partition table on a disk with parted	607
64.3. CREATING A PARTITION	608
64.3.1. Considerations before modifying partitions on a disk	608
The maximum number of partitions	608
The maximum size of a partition	608
Size alignment	608
64.3.2. Partition types	609
Partition types or flags	609
Partition file system type	609
64.3.3. Creating a partition with parted	610
64.3.4. Setting a partition type with fdisk	611
64.4. REMOVING A PARTITION	612
64.4.1. Considerations before modifying partitions on a disk	612
The maximum number of partitions	613
The maximum size of a partition	613
Size alignment	613
64.4.2. Removing a partition with parted	613
64.5. RESIZING A PARTITION	615
64.5.1. Considerations before modifying partitions on a disk	615
The maximum number of partitions	615
The maximum size of a partition	615
Size alignment	615
64.5.2. Resizing a partition with parted	616
CHAPTER 65. GETTING STARTED WITH XFS	618
65.1. THE XFS FILE SYSTEM	618
Performance characteristics	619
65.2. CREATING AN XFS FILE SYSTEM	619
65.2.1. Creating an XFS file system with mkfs.xfs	619
65.2.2. Creating an XFS file system on a block device using RHEL System Roles	620

65.2.2.1. Example Ansible playbook to create an XFS file system on a block device	620
65.3. BACKING UP AN XFS FILE SYSTEM	621
65.3.1. Features of XFS backup	621
65.3.2. Backing up an XFS file system with xfsdump	621
65.3.3. Additional resources	622
65.4. RESTORING AN XFS FILE SYSTEM FROM BACKUP	622
65.4.1. Features of restoring XFS from backup	623
65.4.2. Restoring an XFS file system from backup with xfsrestore	623
65.4.3. Informational messages when restoring an XFS backup from a tape	624
65.4.4. Additional resources	624
65.5. INCREASING THE SIZE OF AN XFS FILE SYSTEM	625
65.5.1. Increasing the size of an XFS file system with xfs_growfs	625
65.6. COMPARISON OF TOOLS USED WITH EXT4 AND XFS	625
CHAPTER 66. MOUNTING FILE SYSTEMS	627
66.1. THE LINUX MOUNT MECHANISM	627
66.2. LISTING CURRENTLY MOUNTED FILE SYSTEMS	627
66.3. MOUNTING A FILE SYSTEM WITH MOUNT	628
66.4. MOVING A MOUNT POINT	629
66.5. UNMOUNTING A FILE SYSTEM WITH UOUNT	629
66.6. COMMON MOUNT OPTIONS	630
66.7. SHARING A MOUNT ON MULTIPLE MOUNT POINTS	631
66.7.1. Types of shared mounts	631
66.7.2. Creating a private mount point duplicate	631
66.7.3. Creating a shared mount point duplicate	633
66.7.4. Creating a slave mount point duplicate	634
66.7.5. Preventing a mount point from being duplicated	635
66.7.6. Related information	636
66.8. PERSISTENTLY MOUNTING FILE SYSTEMS	636
66.8.1. The /etc/fstab file	636
66.8.2. Adding a file system to /etc/fstab	636
66.8.3. Persistently mounting a file system using RHEL System Roles	637
66.8.3.1. Example Ansible playbook to persistently mount a file system	638
66.9. MOUNTING FILE SYSTEMS ON DEMAND	638
66.9.1. The autofs service	638
66.9.2. The autofs configuration files	639
The master map file	639
Map files	639
The amd map format	640
66.9.3. Configuring autofs mount points	640
66.9.4. Automounting NFS server user home directories with autofs service	641
66.9.5. Overriding or augmenting autofs site configuration files	641
66.9.6. Using LDAP to store automounter maps	643
66.10. SETTING READ-ONLY PERMISSIONS FOR THE ROOT FILE SYSTEM	645
66.10.1. Files and directories that always retain write permissions	645
66.10.2. Configuring the root file system to mount with read-only permissions on boot	646
CHAPTER 67. MANAGING STORAGE DEVICES	648
CHAPTER 68. MANAGING LAYERED LOCAL STORAGE WITH STRATIS	649
68.1. SETTING UP STRATIS FILE SYSTEMS	649
68.1.1. The purpose and features of Stratis	649
68.1.2. Components of a Stratis volume	649
68.1.3. Block devices usable with Stratis	650

Supported devices	650
Unsupported devices	651
68.1.4. Installing Stratis	651
68.1.5. Creating a Stratis pool	651
68.1.6. Creating a Stratis file system	652
68.1.7. Mounting a Stratis file system	653
68.1.8. Persistently mounting a Stratis file system	653
68.1.9. Related information	654
68.2. EXTENDING A STRATIS VOLUME WITH ADDITIONAL BLOCK DEVICES	654
68.2.1. Components of a Stratis volume	654
68.2.2. Adding block devices to a Stratis pool	655
68.2.3. Related information	656
68.3. MONITORING STRATIS FILE SYSTEMS	656
68.3.1. Stratis sizes reported by different utilities	656
68.3.2. Displaying information about Stratis volumes	656
68.3.3. Related information	657
68.4. USING SNAPSHOT ON STRATIS FILE SYSTEMS	657
68.4.1. Characteristics of Stratis snapshots	657
68.4.2. Creating a Stratis snapshot	657
68.4.3. Accessing the content of a Stratis snapshot	658
68.4.4. Reverting a Stratis file system to a previous snapshot	658
68.4.5. Removing a Stratis snapshot	659
68.4.6. Related information	659
68.5. REMOVING STRATIS FILE SYSTEMS	660
68.5.1. Components of a Stratis volume	660
68.5.2. Removing a Stratis file system	660
68.5.3. Removing a Stratis pool	661
68.5.4. Related information	662
CHAPTER 69. GETTING STARTED WITH SWAP	663
69.1. SWAP SPACE	663
69.2. RECOMMENDED SYSTEM SWAP SPACE	663
69.3. ADDING SWAP SPACE	664
69.3.1. Extending swap on an LVM2 logical volume	664
69.3.2. Creating an LVM2 logical volume for swap	665
69.3.3. Creating a swap file	665
69.4. REMOVING SWAP SPACE	666
69.4.1. Reducing swap on an LVM2 logical volume	666
69.4.2. Removing an LVM2 logical volume for swap	667
69.4.3. Removing a swap file	667
CHAPTER 70. DEDUPLICATING AND COMPRESSING STORAGE	669
CHAPTER 71. DEPLOYING VDO	670
71.1. INTRODUCTION TO VDO	670
71.2. VDO DEPLOYMENT SCENARIOS	670
KVM	670
File systems	671
iSCSI target	671
LVM	671
Encryption	672
71.3. COMPONENTS OF A VDO VOLUME	672
71.4. THE PHYSICAL AND LOGICAL SIZE OF A VDO VOLUME	673
71.5. SLAB SIZE IN VDO	674

71.6. VDO REQUIREMENTS	675
71.6.1. Placement of VDO in the storage stack	675
71.6.2. VDO memory requirements	676
The VDO module	676
The Universal Deduplication Service (UDS) index	676
71.6.3. VDO storage space requirements	676
71.6.4. Examples of VDO requirements by physical volume size	677
Primary storage deployment	677
Backup storage deployment	677
71.7. INSTALLING VDO	678
71.8. CREATING A VDO VOLUME	678
71.9. MOUNTING A VDO VOLUME	680
71.10. ENABLING PERIODIC BLOCK DISCARD	680
71.11. MONITORING VDO	680
CHAPTER 72. MAINTAINING VDO	682
72.1. MANAGING FREE SPACE ON VDO VOLUMES	682
72.1.1. The physical and logical size of a VDO volume	682
72.1.2. Thin provisioning in VDO	683
72.1.3. Monitoring VDO	684
72.1.4. Reclaiming space for VDO on file systems	684
72.1.5. Reclaiming space for VDO without a file system	685
72.1.6. Reclaiming space for VDO on Fibre Channel or Ethernet network	685
72.2. STARTING OR STOPPING VDO VOLUMES	685
72.2.1. Started and activated VDO volumes	685
72.2.2. Starting a VDO volume	686
72.2.3. Stopping a VDO volume	686
72.2.4. Related information	687
72.3. AUTOMATICALLY STARTING VDO VOLUMES AT SYSTEM BOOT	687
72.3.1. Started and activated VDO volumes	687
72.3.2. Activating a VDO volume	687
72.3.3. Deactivating a VDO volume	688
72.4. SELECTING A VDO WRITE MODE	688
72.4.1. VDO write modes	688
72.4.2. The internal processing of VDO write modes	689
72.4.3. Checking the write mode on a VDO volume	690
72.4.4. Checking for a volatile cache	690
72.4.5. Setting a VDO write mode	691
72.5. RECOVERING A VDO VOLUME AFTER AN UNCLEAN SHUTDOWN	691
72.5.1. VDO write modes	691
72.5.2. VDO volume recovery	692
Automatic and manual recovery	692
72.5.3. VDO operating modes	693
72.5.4. Recovering a VDO volume online	694
72.5.5. Forcing an offline rebuild of a VDO volume metadata	694
72.5.6. Removing an unsuccessfully created VDO volume	695
72.6. OPTIMIZING THE UDS INDEX	695
72.6.1. Components of a VDO volume	695
72.6.2. The UDS index	696
72.6.3. Recommended UDS index configuration	697
72.7. ENABLING OR DISABLING DEDUPLICATION IN VDO	698
72.7.1. Deduplication in VDO	698
72.7.2. Enabling deduplication on a VDO volume	698

72.7.3. Disabling deduplication on a VDO volume	698
72.8. ENABLING OR DISABLING COMPRESSION IN VDO	699
72.8.1. Compression in VDO	699
72.8.2. Enabling compression on a VDO volume	699
72.8.3. Disabling compression on a VDO volume	699
72.9. INCREASING THE SIZE OF A VDO VOLUME	700
72.9.1. The physical and logical size of a VDO volume	700
72.9.2. Thin provisioning in VDO	701
72.9.3. Increasing the logical size of a VDO volume	702
72.9.4. Increasing the physical size of a VDO volume	702
72.10. REMOVING VDO VOLUMES	703
72.10.1. Removing a working VDO volume	703
72.10.2. Removing an unsuccessfully created VDO volume	703
72.11. RELATED INFORMATION	703
CHAPTER 73. DISCARDING UNUSED BLOCKS	705
73.1. BLOCK DISCARD OPERATIONS	705
Requirements	705
73.2. TYPES OF BLOCK DISCARD OPERATIONS	705
Recommendations	705
73.3. PERFORMING BATCH BLOCK DISCARD	705
73.4. ENABLING ONLINE BLOCK DISCARD	706
73.5. ENABLING ONLINE BLOCK DISCARD USING RHEL SYSTEM ROLES	706
73.5.1. Example Ansible playbook to enable online block discard	707
73.6. ENABLING PERIODIC BLOCK DISCARD	707
CHAPTER 74. USING THE WEB CONSOLE FOR MANAGING VIRTUAL DATA OPTIMIZER VOLUMES	708
74.1. VDO VOLUMES IN THE WEB CONSOLE	708
74.2. CREATING VDO VOLUMES IN THE WEB CONSOLE	709
74.3. FORMATTING VDO VOLUMES IN THE WEB CONSOLE	710
74.4. EXTENDING VDO VOLUMES IN THE WEB CONSOLE	712
PART V. DESIGN OF LOG FILE	715
CHAPTER 75. AUDITING THE SYSTEM	716
75.1. LINUX AUDIT	716
75.2. AUDIT SYSTEM ARCHITECTURE	717
75.3. CONFIGURING AUDITD FOR A SECURE ENVIRONMENT	718
75.4. STARTING AND CONTROLLING AUDITD	719
75.5. UNDERSTANDING AUDIT LOG FILES	720
75.6. USING AUDITCTL FOR DEFINING AND EXECUTING AUDIT RULES	724
75.7. DEFINING PERSISTENT AUDIT RULES	725
75.8. USING PRE-CONFIGURED RULES FILES	725
75.9. USING AUGENRULES TO DEFINE PERSISTENT RULES	726
75.10. RELATED INFORMATION	726
PART VI. DESIGN OF KERNEL	728
CHAPTER 76. THE LINUX KERNEL RPM	729
76.1. WHAT AN RPM IS	729
Types of RPM packages	729
76.2. THE LINUX KERNEL RPM PACKAGE OVERVIEW	729
76.3. DISPLAYING CONTENTS OF THE KERNEL PACKAGE	730
CHAPTER 77. UPDATING KERNEL WITH YUM	732

77.1. WHAT IS THE KERNEL	732
77.2. WHAT IS YUM	732
77.3. UPDATING THE KERNEL	732
77.4. INSTALLING THE KERNEL	733
CHAPTER 78. CONFIGURING KERNEL COMMAND-LINE PARAMETERS	734
78.1. UNDERSTANDING KERNEL COMMAND-LINE PARAMETERS	734
78.2. WHAT GRUBBY IS	734
78.3. WHAT BOOT ENTRIES ARE	735
78.4. SETTING KERNEL COMMAND-LINE PARAMETERS	735
78.4.1. Changing kernel command-line parameters for all boot entries	735
78.4.2. Changing kernel command-line parameters for a single boot entry	736
CHAPTER 79. CONFIGURING KERNEL PARAMETERS AT RUNTIME	738
79.1. WHAT ARE KERNEL PARAMETERS	738
79.2. SETTING KERNEL PARAMETERS AT RUNTIME	739
79.2.1. Configuring kernel parameters temporarily with sysctl	739
79.2.2. Configuring kernel parameters permanently with sysctl	740
79.2.3. Using configuration files in /etc/sysctl.d/ to adjust kernel parameters	740
79.2.4. Configuring kernel parameters temporarily through /proc/sys/	741
79.3. KEEPING KERNEL PANIC PARAMETERS DISABLED IN VIRTUALIZED ENVIRONMENTS	742
79.3.1. What is a soft lockup	742
79.3.2. Parameters controlling kernel panic	742
79.3.3. Spurious soft lockups in virtualized environments	743
79.4. ADJUSTING KERNEL PARAMETERS FOR DATABASE SERVERS	744
79.4.1. Introduction to database servers	744
79.4.2. Parameters affecting performance of database applications	744
CHAPTER 80. INSTALLING AND CONFIGURING KDUMP	747
80.1. WHAT IS KDUMP	747
80.2. INSTALLING KDUMP	747
80.3. CONFIGURING KDUMP ON THE COMMAND LINE	748
80.3.1. Configuring kdump memory usage	748
80.3.2. Configuring the kdump target	749
80.3.3. Configuring the core collector	752
80.3.4. Configuring the kdump default failure responses	752
80.3.5. Enabling and disabling the kdump service	753
80.4. CONFIGURING KDUMP IN THE WEB CONSOLE	754
80.4.1. Configuring kdump memory usage and target location in web console	754
80.5. SUPPORTED KDUMP CONFIGURATIONS AND TARGETS	756
80.5.1. Memory requirements for kdump	756
80.5.2. Minimum threshold for automatic memory reservation	757
80.5.3. Supported kdump targets	758
80.5.4. Supported kdump filtering levels	759
80.5.5. Supported default failure responses	760
80.5.6. Estimating kdump size	761
80.6. TESTING THE KDUMP CONFIGURATION	761
80.7. USING KEXEC TO REBOOT THE KERNEL	762
80.8. BLACKLISTING KERNEL DRIVERS FOR KDUMP	763
80.9. RUNNING KDUMP ON SYSTEMS WITH ENCRYPTED DISK	764
80.10. ANALYZING A CORE DUMP	764
80.10.1. Installing the crash utility	765
80.10.2. Running and exiting the crash utility	765
80.10.3. Displaying various indicators in the crash utility	766

80.10.4. Using Kernel Oops Analyzer	769
80.11. USING EARLY KDUMP TO CAPTURE BOOT TIME CRASHES	770
80.11.1. What is early kdump	770
80.11.2. Enabling early kdump	770
80.12. RELATED INFORMATION	771
PART VII. SETTING LIMITS FOR APPLICATIONS	773
CHAPTER 81. UNDERSTANDING CONTROL GROUPS	774
CHAPTER 82. WHAT KERNEL RESOURCE CONTROLLERS ARE	776
CHAPTER 83. USING CONTROL GROUPS THROUGH A VIRTUAL FILE SYSTEM	778
83.1. SETTING CPU LIMITS TO APPLICATIONS USING CGROUPS-V1	778
83.2. SETTING CPU LIMITS TO APPLICATIONS USING CGROUPS-V2	781
CHAPTER 84. ROLE OF SYSTEMD IN CONTROL GROUPS VERSION 1	786
CHAPTER 85. USING CONTROL GROUPS VERSION 1 WITH SYSTEMD	788
85.1. CREATING CONTROL GROUPS VERSION 1 WITH SYSTEMD	788
85.1.1. Creating transient control groups	788
85.1.2. Creating persistent control groups	789
85.2. MODIFYING CONTROL GROUPS VERSION 1 WITH SYSTEMD	789
85.2.1. Configuring memory resource control settings on the command-line	789
85.2.2. Configuring memory resource control settings with unit files	790
85.3. REMOVING CONTROL GROUPS VERSION 1 WITH SYSTEMD	791
85.3.1. Removing transient control groups	791
85.3.2. Removing persistent control groups	792
CHAPTER 86. OBTAINING INFORMATION ABOUT CONTROL GROUPS VERSION 1	794
86.1. LISTING SYSTEMD UNITS	794
86.2. VIEWING A CONTROL GROUP VERSION 1 HIERARCHY	795
86.3. VIEWING RESOURCE CONTROLLERS	797
86.4. MONITORING RESOURCE CONSUMPTION	797
CHAPTER 87. WHAT NAMESPACES ARE	799
CHAPTER 88. ANALYZING SYSTEM PERFORMANCE WITH BPF COMPILER COLLECTION	800
88.1. A BRIEF INTRODUCTION TO BCC	800
88.2. INSTALLING THE BCC-TOOLS PACKAGE	800
88.3. USING SELECTED BCC-TOOLS FOR PERFORMANCE ANALYSES	801
Using execsnoop to examine the system processes	801
Using opensnoop to track what files a command opens	802
Using biotop to examine the I/O operations on the disk	802
Using xfsslower to expose unexpectedly slow file system operations	803
PART VIII. DESIGN OF HIGH AVAILABILITY SYSTEM	805
CHAPTER 89. HIGH AVAILABILITY ADD-ON OVERVIEW	806
89.1. HIGH AVAILABILITY ADD-ON COMPONENTS	806
89.2. PACEMAKER OVERVIEW	806
89.2.1. Pacemaker architecture components	806
89.2.2. Configuration and management tools	807
89.2.3. The cluster and pacemaker configuration files	808
89.3. FENCING OVERVIEW	808
89.4. QUORUM OVERVIEW	808

89.5. RESOURCE OVERVIEW	809
89.6. LVM LOGICAL VOLUMES IN A RED HAT HIGH AVAILABILITY CLUSTER	809
89.6.1. Choosing HA-LVM or shared volumes	809
89.6.2. Configuring LVM volumes in a cluster	810
CHAPTER 90. GETTING STARTED WITH PACEMAKER	811
90.1. LEARNING TO USE PACEMAKER	811
90.2. LEARNING TO CONFIGURE FAILOVER	815
CHAPTER 91. THE PCS COMMAND LINE INTERFACE	820
91.1. PCS HELP DISPLAY	820
91.2. VIEWING THE RAW CLUSTER CONFIGURATION	820
91.3. SAVING A CONFIGURATION CHANGE TO A WORKING FILE	820
91.4. DISPLAYING CLUSTER STATUS	821
91.5. DISPLAYING THE FULL CLUSTER CONFIGURATION	821
CHAPTER 92. CREATING A RED HAT HIGH-AVAILABILITY CLUSTER WITH PACEMAKER	822
92.1. INSTALLING CLUSTER SOFTWARE	822
92.2. INSTALLING THE PCP-ZEROCONF PACKAGE (RECOMMENDED)	823
92.3. CREATING A HIGH AVAILABILITY CLUSTER	824
92.4. CREATING A HIGH AVAILABILITY CLUSTER WITH MULTIPLE LINKS	825
92.5. CONFIGURING FENCING	826
92.6. BACKING UP AND RESTORING A CLUSTER CONFIGURATION	827
92.7. ENABLING PORTS FOR THE HIGH AVAILABILITY ADD-ON	827
CHAPTER 93. CONFIGURING AN ACTIVE/PASSIVE APACHE HTTP SERVER IN A RED HAT HIGH AVAILABILITY CLUSTER	830
93.1. CONFIGURING AN LVM VOLUME WITH AN EXT4 FILE SYSTEM IN A PACEMAKER CLUSTER	831
93.2. CONFIGURING AN APACHE HTTP SERVER	832
93.3. CREATING THE RESOURCES AND RESOURCE GROUPS	833
93.4. TESTING THE RESOURCE CONFIGURATION	835
CHAPTER 94. CONFIGURING AN ACTIVE/PASSIVE NFS SERVER IN A RED HAT HIGH AVAILABILITY CLUSTER	837
94.1. PREREQUISITES	837
94.2. PROCEDURAL OVERVIEW	837
94.3. CONFIGURING AN LVM VOLUME WITH AN EXT4 FILE SYSTEM IN A PACEMAKER CLUSTER	837
94.4. CONFIGURING AN NFS SHARE	839
94.5. CONFIGURING THE RESOURCES AND RESOURCE GROUP FOR AN NFS SERVER IN A CLUSTER	839
94.6. TESTING THE NFS RESOURCE CONFIGURATION	843
94.6.1. Testing the NFS export	843
94.6.2. Testing for failover	843
CHAPTER 95. GFS2 FILE SYSTEMS IN A CLUSTER	845
95.1. CONFIGURING A GFS2 FILE SYSTEM IN A CLUSTER	845
95.2. MIGRATING A GFS2 FILE SYSTEM FROM RHEL7 TO RHEL8	849
CHAPTER 96. CONFIGURING FENCING IN A RED HAT HIGH AVAILABILITY CLUSTER	851
96.1. DISPLAYING AVAILABLE FENCE AGENTS AND THEIR OPTIONS	851
96.2. CREATING A FENCE DEVICE	852
96.3. GENERAL PROPERTIES OF FENCING DEVICES	852
96.4. ADVANCED FENCING CONFIGURATION OPTIONS	853
96.5. TESTING A FENCE DEVICE	860
96.6. CONFIGURING FENCING LEVELS	862
96.7. CONFIGURING FENCING FOR REDUNDANT POWER SUPPLIES	864

96.8. DISPLAYING CONFIGURED FENCE DEVICES	864
96.9. MODIFYING AND DELETING FENCE DEVICES	864
96.10. MANUALLY FENCING A CLUSTER NODE	865
96.11. DISABLING A FENCE DEVICE	865
96.12. PREVENTING A NODE FROM USING A FENCE DEVICE	865
96.13. CONFIGURING ACPI FOR USE WITH INTEGRATED FENCE DEVICES	865
96.13.1. Disabling ACPI Soft-Off with the BIOS	866
96.13.2. Disabling ACPI Soft-Off in the logind.conf file	867
96.13.3. Disabling ACPI completely in the GRUB 2 File	868
CHAPTER 97. CONFIGURING CLUSTER RESOURCES	869
Resource creation examples	869
Deleting a configured resource	869
97.1. RESOURCE AGENT IDENTIFIERS	869
97.2. DISPLAYING RESOURCE-SPECIFIC PARAMETERS	870
97.3. CONFIGURING RESOURCE META OPTIONS	871
97.3.1. Changing the default value of a resource option	873
97.3.2. Displaying currently configured resource defaults	873
97.3.3. Setting meta options on resource creation	873
97.4. CONFIGURING RESOURCE GROUPS	874
97.4.1. Creating a resource group	874
97.4.2. Removing a resource group	875
97.4.3. Displaying resource groups	875
97.4.4. Group options	875
97.4.5. Group stickiness	875
97.5. DETERMINING RESOURCE BEHAVIOR	876
CHAPTER 98. DETERMINING WHICH NODES A RESOURCE CAN RUN ON	877
98.1. CONFIGURING LOCATION CONSTRAINTS	877
98.2. LIMITING RESOURCE DISCOVERY TO A SUBSET OF NODES	878
98.3. CONFIGURING A LOCATION CONSTRAINT STRATEGY	880
98.3.1. Configuring an "Opt-In" Cluster	880
98.3.2. Configuring an "Opt-Out" Cluster	881
98.4. CONFIGURING A RESOURCE TO PREFER ITS CURRENT NODE	881
CHAPTER 99. DETERMINING THE ORDER IN WHICH CLUSTER RESOURCES ARE RUN	883
99.1. CONFIGURING MANDATORY ORDERING	884
99.2. CONFIGURING ADVISORY ORDERING	884
99.3. CONFIGURING ORDERED RESOURCE SETS	884
99.4. CONFIGURING STARTUP ORDER FOR RESOURCE DEPENDENCIES NOT MANAGED BY PACEMAKER	886
CHAPTER 100. COLOCATING CLUSTER RESOURCES	887
100.1. SPECIFYING MANDATORY PLACEMENT OF RESOURCES	887
100.2. SPECIFYING ADVISORY PLACEMENT OF RESOURCES	888
100.3. COLOCATING SETS OF RESOURCES	888
100.4. REMOVING COLOCATION CONSTRAINTS	889
CHAPTER 101. DISPLAYING RESOURCE CONSTRAINTS	890
101.1. DISPLAYING ALL CONFIGURED CONSTRAINTS	890
101.2. DISPLAYING LOCATION CONSTRAINTS	890
101.3. DISPLAYING ORDERING CONSTRAINTS	890
101.4. DISPLAYING COLOCATION CONSTRAINTS	890
101.5. DISPLAYING RESOURCE-SPECIFIC CONSTRAINTS	890

101.6. DISPLAYING RESOURCE DEPENDENCIES (RED HAT ENTERPRISE LINUX 8.2 AND LATER)	890
CHAPTER 102. DETERMINING RESOURCE LOCATION WITH RULES	893
102.1. PACEMAKER RULES	893
102.1.1. Node attribute expressions	893
102.1.2. Time/date based expressions	895
102.1.3. Date specifications	896
102.2. CONFIGURING A PACEMAKER LOCATION CONSTRAINT USING RULES	896
CHAPTER 103. MANAGING CLUSTER RESOURCES	898
103.1. DISPLAYING CONFIGURED RESOURCES	898
103.2. MODIFYING RESOURCE PARAMETERS	898
103.3. CLEARING FAILURE STATUS OF CLUSTER RESOURCES	899
103.4. MOVING RESOURCES IN A CLUSTER	899
103.4.1. Moving resources due to failure	899
103.4.2. Moving resources due to connectivity changes	900
103.5. DISABLING A MONITOR OPERATION	901
CHAPTER 104. CREATING CLUSTER RESOURCES THAT ARE ACTIVE ON MULTIPLE NODES (CLONED RESOURCES)	903
104.1. CREATING AND REMOVING A CLONED RESOURCE	903
104.2. CONFIGURING CLONE RESOURCE CONSTRAINTS	905
104.3. CREATING PROMOTABLE CLONE RESOURCES	906
104.3.1. Creating a promotable resource	906
104.3.2. Configuring promotable resource constraints	906
CHAPTER 105. MANAGING CLUSTER NODES	908
105.1. STOPPING CLUSTER SERVICES	908
105.2. ENABLING AND DISABLING CLUSTER SERVICES	908
105.3. ADDING CLUSTER NODES	908
105.4. REMOVING CLUSTER NODES	910
105.5. ADDING A NODE TO A CLUSTER WITH MULTIPLE LINKS	910
105.6. ADDING AND MODIFYING LINKS IN AN EXISTING CLUSTER (RHEL 8.1 AND LATER)	910
105.6.1. Adding and removing links in an existing cluster	910
105.6.2. Modifying a link in a cluster with multiple links	911
105.6.3. Modifying the link addresses in a cluster with a single link	911
105.6.4. Modifying the link options for a link in a cluster with a single link	912
105.6.5. Modifying a link when adding a new link is not possible	912
CHAPTER 106. PACEMAKER CLUSTER PROPERTIES	914
106.1. SUMMARY OF CLUSTER PROPERTIES AND OPTIONS	914
106.2. SETTING AND REMOVING CLUSTER PROPERTIES	916
106.3. QUERYING CLUSTER PROPERTY SETTINGS	917
CHAPTER 107. CONFIGURING A VIRTUAL DOMAIN AS A RESOURCE	918
107.1. VIRTUAL DOMAIN RESOURCE OPTIONS	918
107.2. CREATING THE VIRTUAL DOMAIN RESOURCE	920
CHAPTER 108. CLUSTER QUORUM	922
108.1. CONFIGURING QUORUM OPTIONS	922
108.2. MODIFYING QUORUM OPTIONS	923
108.3. DISPLAYING QUORUM CONFIGURATION AND STATUS	923
108.4. RUNNING INQUORATE CLUSTERS	924
108.5. QUORUM DEVICES	924
108.5.1. Installing quorum device packages	925

108.5.2. Configuring a quorum device	925
108.5.3. Managing the Quorum Device Service	929
108.5.4. Managing the quorum device settings in a cluster	930
108.5.4.1. Changing quorum device settings	930
108.5.4.2. Removing a quorum device	930
108.5.4.3. Destroying a quorum device	931
CHAPTER 109. INTEGRATING NON-COROSYNC NODES INTO A CLUSTER: THE PACEMAKER_REMOTE SERVICE	932
109.1. HOST AND GUEST AUTHENTICATION OF PACEMAKER_REMOTE NODES	933
109.2. CONFIGURING KVM GUEST NODES	933
109.2.1. Guest node resource options	933
109.2.2. Integrating a virtual machine as a guest node	934
109.3. CONFIGURING PACEMAKER REMOTE NODES	935
109.3.1. Remote node resource options	935
109.3.2. Remote node configuration overview	935
109.4. CHANGING THE DEFAULT PORT LOCATION	936
109.5. UPGRADING SYSTEMS WITH PACEMAKER_REMOTE NODES	937
CHAPTER 110. PERFORMING CLUSTER MAINTENANCE	938
110.1. PUTTING A NODE INTO STANDBY MODE	938
110.2. MANUALLY MOVING CLUSTER RESOURCES	939
110.2.1. Moving a resource from its current node	939
110.2.2. Moving a resource to its preferred node	940
110.3. DISABLING, ENABLING, AND BANNING CLUSTER RESOURCES	940
Disabling a cluster resource	941
Enabling a cluster resource	941
Preventing a resource from running on a particular node	941
Forcing a resource to start on the current node	942
110.4. SETTING A RESOURCE TO UNMANAGED MODE	942
110.5. PUTTING A CLUSTER IN MAINTENANCE MODE	942
110.6. UPDATING A RHEL HIGH AVAILABILITY CLUSTER	943
110.7. UPGRADING REMOTE NODES AND GUEST NODES	943
CHAPTER 111. CONFIGURING AND MANAGING LOGICAL VOLUMES	945
CHAPTER 112. LOGICAL VOLUMES	946
112.1. LVM ARCHITECTURE OVERVIEW	946
112.2. PHYSICAL VOLUMES	947
112.2.1. LVM physical volume layout	947
112.2.2. Multiple partitions on a disk	948
112.3. VOLUME GROUPS	948
112.4. LVM LOGICAL VOLUMES	949
112.4.1. Linear Volumes	949
112.4.2. Striped Logical Volumes	951
112.4.3. RAID logical volumes	952
112.4.4. Thinly-provisioned logical volumes (thin volumes)	953
112.4.5. Snapshot Volumes	953
112.4.6. Thinly-provisioned snapshot volumes	954
112.4.7. Cache Volumes	955
CHAPTER 113. CONFIGURING LVM LOGICAL VOLUMES	956
113.1. USING CLI COMMANDS	956
Specifying units in a command line argument	956

Specifying volume groups and logical volumes	956
Increasing output verbosity	956
Displaying help for LVM CLI commands	957
113.2. CREATING AN LVM LOGICAL VOLUME ON THREE DISKS	957
113.3. CREATING A RAID0 (STRIPED) LOGICAL VOLUME	958
113.4. RENAMING LVM LOGICAL VOLUMES	960
113.5. REMOVING A DISK FROM A LOGICAL VOLUME	961
113.5.1. Moving extents to existing physical volumes	961
113.5.2. Moving Extents to a New Disk	962
113.6. CONFIGURING PERSISTENT DEVICE NUMBERS	963
113.7. SPECIFYING LVM EXTENT SIZE	963
113.8. MANAGING LVM LOGICAL VOLUMES USING RHEL SYSTEM ROLES	963
113.8.1. Example Ansible playbook to manage logical volumes	963
113.8.2. Additional resources	964
113.9. REMOVING LVM LOGICAL VOLUMES	964
CHAPTER 114. MODIFYING THE SIZE OF A LOGICAL VOLUME	966
114.1. GROWING LOGICAL VOLUMES	966
114.2. GROWING A FILE SYSTEM ON A LOGICAL VOLUME	966
114.3. SHRINKING LOGICAL VOLUMES	967
114.4. EXTENDING A STRIPED LOGICAL VOLUME	968
CHAPTER 115. MANAGING LVM PHYSICAL VOLUMES	970
115.1. SCANNING FOR BLOCK DEVICES TO USE AS PHYSICAL VOLUMES	970
115.2. SETTING THE PARTITION TYPE FOR A PHYSICAL VOLUME	970
115.3. RESIZING AN LVM PHYSICAL VOLUME	971
115.4. REMOVING PHYSICAL VOLUMES	971
115.5. ADDING PHYSICAL VOLUMES TO A VOLUME GROUP	971
115.6. REMOVING PHYSICAL VOLUMES FROM A VOLUME GROUP	971
CHAPTER 116. DISPLAYING LVM COMPONENTS	973
116.1. DISPLAYING LVM INFORMATION WITH THE LVM COMMAND	973
116.2. DISPLAYING PHYSICAL VOLUMES	973
116.3. DISPLAYING VOLUME GROUPS	974
116.4. DISPLAYING LOGICAL VOLUMES	975
CHAPTER 117. CUSTOMIZED REPORTING FOR LVM	976
117.1. CONTROLLING THE FORMAT OF THE LVM DISPLAY	976
117.2. LVM OBJECT DISPLAY FIELDS	977
117.3. SORTING LVM REPORTS	986
117.4. SPECIFYING THE UNITS FOR AN LVM REPORT DISPLAY	986
117.5. DISPLAYING LVM COMMAND OUTPUT IN JSON FORMAT	987
117.6. DISPLAYING THE LVM COMMAND LOG	988
CHAPTER 118. CONFIGURING RAID LOGICAL VOLUMES	990
118.1. CREATING RAID LOGICAL VOLUMES	991
118.2. CREATING A RAID0 (STRIPED) LOGICAL VOLUME	992
118.3. CONTROLLING THE RATE AT WHICH RAID VOLUMES ARE INITIALIZED	994
118.4. CONVERTING A LINEAR DEVICE TO A RAID DEVICE	994
118.5. CONVERTING AN LVM RAID1 LOGICAL VOLUME TO AN LVM LINEAR LOGICAL VOLUME	995
118.6. CONVERTING A MIRRORED LVM DEVICE TO A RAID1 DEVICE	996
118.7. RESIZING A RAID LOGICAL VOLUME	996
118.8. CHANGING THE NUMBER OF IMAGES IN AN EXISTING RAID1 DEVICE	997
118.9. SPLITTING OFF A RAID IMAGE AS A SEPARATE LOGICAL VOLUME	999

118.10. SPLITTING AND MERGING A RAID IMAGE	1000
118.11. SETTING A RAID FAULT POLICY	1002
118.11.1. The allocate RAID Fault Policy	1002
118.11.2. The warn RAID Fault Policy	1003
118.12. REPLACING A RAID DEVICE IN A LOGICAL VOLUME	1003
118.12.1. Replacing a RAID device that has not failed	1003
118.12.2. Replacing a failed RAID device in a logical volume	1005
118.13. CHECKING DATA COHERENCY IN A RAID LOGICAL VOLUME (RAID SCRUBBING)	1006
118.14. CONVERTING A RAID LEVEL (RAID TAKEOVER)	1008
118.15. CHANGING ATTRIBUTES OF A RAID VOLUME (RAID RESHAPE)	1008
118.16. CONTROLLING I/O OPERATIONS ON A RAID1 LOGICAL VOLUME	1008
118.17. CHANGING THE REGION SIZE ON A RAID LOGICAL VOLUME	1008
CHAPTER 119. SNAPSHOT LOGICAL VOLUMES	1010
119.1. SNAPSHOT VOLUMES	1010
119.2. CREATING SNAPSHOT VOLUMES	1011
119.3. MERGING SNAPSHOT VOLUMES	1013
CHAPTER 120. CREATING AND MANAGING THINLY-PROVISIONED LOGICAL VOLUMES (THIN VOLUMES)	1014
120.1. THINLY-PROVISIONED LOGICAL VOLUMES (THIN VOLUMES)	1014
120.2. CREATING THINLY-PROVISIONED LOGICAL VOLUMES	1014
120.3. THINLY-PROVISIONED SNAPSHOT VOLUMES	1017
120.4. CREATING THINLY-PROVISIONED SNAPSHOT VOLUMES	1018
120.5. TRACKING AND DISPLAYING THIN SNAPSHOT VOLUMES THAT HAVE BEEN REMOVED	1020
CHAPTER 121. ENABLING CACHING TO IMPROVE LOGICAL VOLUME PERFORMANCE	1024
121.1. CACHING METHODS IN LVM	1024
121.2. LVM CACHING COMPONENTS	1024
121.3. ENABLING DM-CACHE CACHING FOR A LOGICAL VOLUME	1024
121.4. ENABLING DM-WRITECACHE CACHING FOR A LOGICAL VOLUME	1026
121.5. DISABLING CACHING FOR A LOGICAL VOLUME	1027
CHAPTER 122. LOGICAL VOLUME ACTIVATION	1029
122.1. CONTROLLING AUTOACTIVATION OF LOGICAL VOLUMES	1029
122.2. CONTROLLING LOGICAL VOLUME ACTIVATION	1030
122.3. ACTIVATING SHARED LOGICAL VOLUMES	1030
122.4. ACTIVATING A LOGICAL VOLUME WITH MISSING DEVICES	1031
CHAPTER 123. CONTROLLING LVM DEVICE SCANNING	1032
123.1. THE LVM DEVICE FILTER	1032
123.2. EXAMPLES OF LVM DEVICE FILTER CONFIGURATIONS	1032
123.3. APPLYING AN LVM DEVICE FILTER CONFIGURATION	1033
CHAPTER 124. CONTROLLING LVM ALLOCATION	1034
124.1. LVM ALLOCATION POLICIES	1034
124.2. PREVENTING ALLOCATION ON A PHYSICAL VOLUME	1035
124.3. EXTENDING A LOGICAL VOLUME WITH THE CLING ALLOCATION POLICY	1035
CHAPTER 125. TROUBLESHOOTING LVM	1037
125.1. GATHERING DIAGNOSTIC DATA ON LVM	1037
125.2. DISPLAYING INFORMATION ON FAILED LVM DEVICES	1038
125.3. REMOVING LOST LVM PHYSICAL VOLUMES FROM A VOLUME GROUP	1039
125.4. RECOVERING AN LVM PHYSICAL VOLUME WITH DAMAGED METADATA	1040
125.4.1. Discovering that an LVM volume has missing or corrupted metadata	1040

125.4.2. Finding the metadata of a missing LVM physical volume	1040
125.4.3. Restoring metadata on an LVM physical volume	1041
125.5. REPLACING A MISSING LVM PHYSICAL VOLUME	1043
125.5.1. Finding the metadata of a missing LVM physical volume	1043
125.5.2. Restoring metadata on an LVM physical volume	1043
125.6. TROUBLESHOOTING INSUFFICIENT FREE EXTENTS FOR A LOGICAL VOLUME	1045
125.6.1. Volume groups	1045
125.6.2. Rounding errors in LVM output	1045
125.6.3. Preventing the rounding error when creating an LVM volume	1046
125.7. TROUBLESHOOTING DUPLICATE PHYSICAL VOLUME WARNINGS FOR MULTIPATHED LVM DEVICES	1047
125.7.1. Root cause of duplicate PV warnings	1047
125.7.2. Cases of duplicate PV warnings	1047
125.7.3. The LVM device filter	1048
125.7.4. Example LVM device filters that prevent duplicate PV warnings	1048
125.7.5. Applying an LVM device filter configuration	1049
125.7.6. Additional resources	1049
PART IX. VIRTUALIZATION ON ARM 64 SYSTEMS	1051
CHAPTER 126. GETTING STARTED WITH VIRTUALIZATION ON ARM 64	1052
126.1. ENABLING VIRTUALIZATION ON ARM 64	1052
126.2. HOW VIRTUALIZATION ON ARM 64 DIFFERS FROM AMD64 AND INTEL 64	1053
126.3. RELATED INFORMATION	1054
CHAPTER 127. CONFIGURING THE SCALABLE VECTOR EXTENSION ON ARM 64 VIRTUAL MACHINES	1055
CHAPTER 128. SHARING FILES BETWEEN THE HOST AND ITS VIRTUAL MACHINES USING VIRTIOFS	1056

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your input on our documentation. Please let us know how we could make it better. To do so:

- For simple comments on specific passages:
 1. Make sure you are viewing the documentation in the *Multi-page HTML* format. In addition, ensure you see the **Feedback** button in the upper right corner of the document.
 2. Use your mouse cursor to highlight the part of text that you want to comment on.
 3. Click the **Add Feedback** pop-up that appears below the highlighted text.
 4. Follow the displayed instructions.
- For submitting more complex feedback, create a Bugzilla ticket:
 1. Go to the [Bugzilla](#) website.
 2. As the Component, use **Documentation**.
 3. Fill in the **Description** field with your suggestion for improvement. Include a link to the relevant part(s) of documentation.
 4. Click **Submit Bug**.

PART I. DESIGN OF INSTALLATION

CHAPTER 1. INTRODUCTION

Red Hat Enterprise Linux 8 delivers a stable, secure, consistent foundation across hybrid cloud deployments with the tools needed to deliver workloads faster with less effort. It can be deployed as a guest on supported hypervisors and Cloud provider environments as well as deployed on physical infrastructure, so your applications can take advantage of innovations in the leading hardware architecture platforms.

1.1. SUPPORTED ARCHITECTURES

Red Hat Enterprise Linux supports the following architectures:

- AMD and Intel 64-bit architectures
- The 64-bit ARM architecture
- IBM Power Systems, Little Endian
- IBM Z

1.2. INSTALLATION TERMINOLOGY

This section describes Red Hat Enterprise Linux installation terminology. Different terminology can be used for the same concepts, depending on its upstream or downstream origin.

Anaconda: The operating system installer used in Fedora, Red Hat Enterprise Linux, and their derivatives. Anaconda is a set of Python modules and scripts with additional files like Gtk widgets (written in C), systemd units, and dracut libraries. Together, they form a tool that allows users to set parameters of the resulting (target) system. In this document, the term **installation program** refers to the installation aspect of **Anaconda**.

CHAPTER 2. QUICK INSTALLATION

If you are familiar with Red Hat Enterprise Linux, review the following sections to determine the best installation method for your requirements. If you are a new Red Hat Enterprise Linux user, it is recommended that you review [Chapter 3, *Preparing for your installation*](#) before you select the installation method.

2.1. AVAILABLE INSTALLATION METHODS

The following GUI installation methods are available:

- **Install RHEL using an ISO image from the Customer Portal:** Install Red Hat Enterprise Linux by downloading the **Binary DVD ISO** image file from the Customer Portal. Registration is performed after the GUI installation completes. This installation method is also supported by Kickstart.
- **Register and install RHEL from the Content Delivery Network:** Register your system, attach subscriptions, and install Red Hat Enterprise Linux from the Content Delivery Network (CDN). This installation method is supported by the **Boot ISO** and **Binary DVD ISO** image files; however, it is recommended that you use the **Boot ISO** image file as the installation source defaults to CDN for the Boot ISO image file. Registration is performed before the installation packages are downloaded and installed from the CDN. This installation method is also supported by Kickstart.



IMPORTANT

You can customize the RHEL installation for your specific requirements using the GUI. You can select additional options for specific environment requirements, for example, Connect to Red Hat, software selection, partitioning, security, and many more. For more information, see [Chapter 5, *Customizing your installation using the GUI*](#).

Advanced installation methods are also available. They are:

- **Perform an automated RHEL installation using Kickstart:** Install Red Hat Enterprise Linux using Kickstart. Kickstart is an automated installation that allows you to execute unattended operating system installation tasks.
- **Perform a remote RHEL installation using VNC:** The RHEL installation program offers two VNC installation modes: Direct and Connect. Once a connection is established, the two modes do not differ. The mode you select depends on your environment.
- **Install RHEL from the network using PXE:** A network installation allows you to install Red Hat Enterprise Linux to a system that has access to an installation server. At a minimum, two systems are required for a network installation.

For more information about the advanced installation methods, see the [Performing an advanced RHEL installation](#) document.

2.2. INSTALLING RHEL USING AN ISO IMAGE FROM THE CUSTOMER PORTAL

Use this procedure to perform a graphical installation of RHEL using a Binary DVD ISO image that you downloaded from the Customer Portal.

Prerequisites

- You have downloaded the Binary DVD ISO image file from the Customer Portal.
- You have created bootable installation media.
- You have booted the installation program and the boot menu is displayed.

Procedure

1. From the boot menu, select **Install Red Hat Enterprise Linux 8.x**
2. Press the **Enter** key on your keyboard.
3. From the **Welcome to Red Hat Enterprise Linux 8.x** window, select your language and location.
4. Click **Continue** to proceed to the **Installation Summary** window.



NOTE

The **Installation Summary** window is the central hub to configure the Red Hat Enterprise Linux graphical user interface. The default settings assigned by the installation program are displayed under each category.

5. From the **Installation Summary** window, accept the default **Localization** and **Software** options.
6. Select **System > Installation Destination**
 - a. From the **Local Standard Disks** pane, select the target disk.
 - b. Click **Done** to accept the selection and the default setting of automatic partitioning, and return to the **Installation Summary** window.
7. Select **Network & Host Name**
 - a. Toggle the **Ethernet** switch to **ON** to enable network configuration.
 - i. Optional: Select a network device and click **Configure** to update the network interface configuration.
 - b. Click **Done** to accept the changes and return to the **Installation Summary** window.
8. Optional: Select **Connect to Red Hat**
 - a. Configure the **Connect to Red Hat** options to attach RHEL subscriptions, install RHEL from the CDN, configure System Purpose, and enable Red Hat Insights.
 - b. Click **Done** to accept the changes and return to the **Installation Summary** window.
9. Optional: Select **KDUMP**.
 - a. Change the default KDUMP settings for your requirements.
 - b. Click **Done** to accept the changes and return to the **Installation Summary** window.
10. Optional: Select **Security Policy**.

- a. Select the profile that you require, and click **Select profile**.
 - b. Click **Done** to accept the changes and return to the **Installation Summary** window.
11. Click **Begin Installation** to start the installation.
 12. From the **Configuration** window, configure a root password and create a user account.
 13. When the installation process is complete, click **Reboot** to restart the system.
 14. From the **Initial Setup** window, accept the licensing agreement and register your system.

Additional resources

- To learn more about how to prepare for your installation, see [Performing a standard RHEL installation](#) for more information.
- To learn more about how to customize your network, connect to Red Hat, system purpose, installation destination, KDUMP, and security policy, see [Performing a standard RHEL installation](#) for more information.
- To learn more about how to register your system, see [Performing a standard RHEL installation](#) for more information.

2.3. REGISTERING AND INSTALLING RHEL FROM THE CDN USING THE GUI

This section contains information about how to register your system, attach RHEL subscriptions, and install RHEL from the Red Hat Content Delivery Network (CDN) using the GUI.

2.3.1. What is the Content Delivery Network

The Red Hat Content Delivery Network (CDN), available from cdn.redhat.com, is a geographically distributed series of static web servers that contain content and errata that is consumed by systems. The content can be consumed directly, such as using a system registered to Red Hat Subscription Management. The CDN is protected by x.509 certificate authentication to ensure that only valid users have access. When a system is registered to Red Hat Subscription Management, the attached subscriptions govern which subset of the CDN the system can access.

Registering and installing RHEL from the CDN provides the following benefits:

- The CDN installation method supports the Boot ISO and the Binary DVD ISO image files. However, the use of the smaller Boot ISO image file is recommended as it consumes less space than the larger Binary DVD ISO image file.
- The CDN uses the latest packages resulting in a fully up-to-date system right after installation. There is no requirement to install package updates immediately after installation as is often the case when using the Binary DVD ISO image file.
- Integrated support for connecting to Red Hat Insights and enabling System Purpose.

Registering and installing RHEL from the CDN is supported by the GUI and Kickstart. For information about how to register and install RHEL using the GUI, see the [Performing a standard RHEL installation](#) document. For information about how to register and install RHEL using Kickstart, see the [Performing an advanced RHEL installation](#) document.

2.3.2. Registering and installing RHEL from the CDN

Use this procedure to register your system, attach RHEL subscriptions, and install RHEL from the Red Hat Content Delivery Network (CDN) using the GUI.



IMPORTANT

The CDN feature is supported by the **Boot ISO** and **Binary DVD ISO** image files. However, it is recommended that you use the **Boot ISO** image file as the installation source defaults to CDN for the Boot ISO image file.

Prerequisites

- Your system is connected to a network that can access the CDN.
- You have downloaded the **Boot ISO** image file from the Customer Portal.
- You have created bootable installation media.
- You have booted the installation program and the boot menu is displayed.



IMPORTANT

The installation repository used after system registration is dependent on how the system was booted.

Procedure

1. From the boot menu, select **Install Red Hat Enterprise Linux 8.x**
2. Press the **Enter** key on your keyboard.
3. From the **Welcome to Red Hat Enterprise Linux 8.x** window, select your language and location.
4. Click **Continue** to proceed to the **Installation Summary** window.



NOTE

The **Installation Summary** window is the central hub to configure a RHEL installation when using the GUI. The default settings assigned by the installation program are displayed under each category. The **Installation Source** defaults to *Red Hat CDN* and **Software Selection** displays a *Red Hat CDN requires registration* message.

5. From the **Installation Summary** window, the default **Localization** settings are displayed.
 - a. Optional: Change the default settings for your requirements.
6. Select **System > Network & Host Name**
 - a. Toggle the **Ethernet** switch to **ON** to enable network configuration.
 - i. Optional: Select a network device and click **Configure** to update the network interface configuration.

- b. Click **Done** to accept the changes and return to the **Installation Summary** window.
7. Select **System > Connect to Red Hat**



IMPORTANT

You can register to the CDN using either your Red Hat account or your activation key details.

- a. Click **Account**.
 - i. Enter your Red Hat Customer Portal username and password details.
- b. Optional: Click **Activation Key**.
 - i. Enter your organization ID and activation key. You can enter more than one activation key, separated by a comma, as long as the activation keys are registered to your subscription.
- c. Select the **Set System Purpose** check box.
 - i. Select the required **Role**, **SLA**, and **Usage** from the corresponding drop-down lists.
- d. The **Connect to Red Hat Insights** check box is enabled by default. Clear the check box if you do not want to connect to Red Hat Insights.



NOTE

Red Hat Insights is a Software-as-a-Service (SaaS) offering that provides continuous, in-depth analysis of registered Red Hat-based systems to proactively identify threats to security, performance and stability across physical, virtual and cloud environments, and container deployments.

- e. Optional: Expand **Options**.
 - i. Select the **Use HTTP proxy** check box if your network environment allows external Internet access only or access to content servers through an HTTP proxy. Clear the **Use HTTP proxy** check box if an HTTP proxy is not used.
 - ii. If you are running Satellite Server or performing internal testing, select the **Custom server URL** and **Custom base URL** check boxes and enter the required details.



IMPORTANT

- o The **Custom server URL** field does not require the HTTP protocol, for example **nameofhost.com**. The **Custom base URL** field does require the HTTP protocol, for example **http://nameofhost.com**.
- o To change the **Custom base URL** after registration, you must unregister, provide the new details, and then re-register.

- f. Click **Register** to register the system. When the system is successfully registered and subscriptions are attached, the **Connect to Red Hat** window displays the attached subscription details.



NOTE

Depending on the amount of subscriptions, the registration and attachment process might take up to a minute to complete.

- g. Click **Done** to return to the **Installation Summary** window.
 - i. A *Registered* message is displayed under **Connect to Red Hat**
8. Select **System > Software Selection**
 - a. Choose the required options from **Base Environment** and **Additional software for selected environment**. The **Server with GUI** base environment is the default base environment and it launches the **Initial Setup** application after the installation completes and you restart the system.
 - b. Click **Done** to accept the selection and return to the **Installation Summary** window.
9. Select **System > Installation Destination**
 - a. From the **Local Standard Disks** pane, select the target disk.
b. Click **Done** to accept the selection and the default setting of automatic partitioning and return to the **Installation Summary** window.
10. Optional: Select **System > KDUMP**.
 - a. Change the default KDUMP settings for your requirements.
b. Click **Done** to accept the changes and return to the **Installation Summary** window.
11. Optional: Select **System > Security Policy**.
 - a. Select the profile that you require and click **Select profile**.
b. Click **Done** to accept the changes and return to the **Installation Summary** window.
12. Click **Begin Installation** to start the installation. The required packages are downloaded from the CDN.
13. From the **Configuration** window, configure a root password and create a user account.
14. When the installation process is complete, click **Reboot** to restart the system.
15. From the **Initial Setup** window, accept the licensing agreement.

Additional resources

- To learn more about how to prepare for your installation, see *Performing a standard RHEL installation* for more information.

- To learn more about how to customize your network, connection to Red Hat, system purpose, installation destination, KDUMP, and security policy, see [Performing a standard RHEL installation](#) for more information.
- For information about Red Hat Insights, see the [Red Hat Insights product documentation](#).
- For information about Activation Keys, see the [Understanding Activation Keys](#) chapter of the [Using Red Hat Subscription Management](#) document.
- For information about how to set up an HTTP proxy for Subscription Manager, see the [Using an HTTP proxy](#) chapter of the [Using and Configuring Red Hat Subscription Manager](#) document.

2.3.2.1. Installation source repository after system registration

The installation source repository used after system registration is dependent on how the system was booted.

System booted from the Boot ISO or the Binary DVD ISO image file

If you booted the RHEL installation using either the **Boot ISO** or the **Binary DVD ISO** image file with the default boot parameters, the installation program automatically switches the installation source repository to the CDN after registration.

System booted with the `inst.repo=<URL>` boot parameter

If you booted the RHEL installation with the **inst.repo=<URL>** boot parameter, the installation program does not automatically switch the installation source repository to the CDN after registration. If you want to use the CDN to install RHEL, you must manually switch the installation source repository to the CDN by selecting the **Red Hat CDN** option in the **Installation Source** window of the graphical installation. If you do not manually switch to the CDN, the installation program installs the packages from the repository specified on the kernel command line.



IMPORTANT

- You can switch the installation source repository to the CDN using the **rhsm** Kickstart command only if you do not specify an installation source using **inst.repo=** on the kernel command line or the **url** command in the Kickstart file. You must use **inst.stage2=<URL>** on the kernel command line to fetch the installation image, but not specify the installation source.
- An installation source URL specified using a boot option or included in a Kickstart file takes precedence over the CDN, even if the Kickstart file contains the **rhsm** command with valid credentials. The system is registered, but it is installed from the URL installation source. This ensures that earlier installation processes operate as normal.

2.3.3. Verifying your system registration from the CDN

Use this procedure to verify that your system is registered to the CDN using the GUI.



WARNING

You can only verify your registration from the CDN if you have **not** clicked the **Begin Installation** button from the **Installation Summary** window. Once the **Begin Installation** button is clicked, you cannot return to the Installation Summary window to verify your registration.

Prerequisite

- You have completed the registration process as documented in the [Section 2.3.2, “Registering and installing RHEL from the CDN”](#) and *Registered* is displayed under **Connect to Red Hat** on the **Installation Summary** window.

Procedure

1. From the **Installation Summary** window, select **Connect to Red Hat**
2. The window opens and displays a registration summary:

Method

The registered account name or activation keys are displayed.

System Purpose

If set, the role, SLA, and usage details are displayed.

Insights

If enabled, the Insights details are displayed.

Number of subscriptions

The number of subscriptions attached are displayed.

3. Verify that the registration summary matches the details that were entered.

2.3.4. Unregistering your system from the CDN

Use this procedure to unregister your system from the CDN using the GUI.



WARNING

- You can unregister from the CDN if you have **not** clicked the **Begin Installation** button from the **Installation Summary** window. Once the **Begin Installation** button is clicked, you cannot return to the Installation Summary window to unregister your registration.
- When unregistering, the installation program switches to the first available repository, in the following order:
 - a. The URL used in the `inst.repo=<URL>` boot parameter on the kernel command line.
 - b. An automatically detected repository on the installation media (USB or DVD).

Prerequisite

- You have completed the registration process as documented in the [Section 2.3.2, “Registering and installing RHEL from the CDN”](#) and **Registered** is displayed under **Connect to Red Hat** on the **Installation Summary** window.

Procedure

1. From the **Installation Summary** window, select **Connect to Red Hat**
2. The **Connect to Red Hat** window opens and displays a registration summary:

Method

The registered account name or activation keys used are displayed.

System Purpose

If set, the role, SLA, and usage details are displayed.

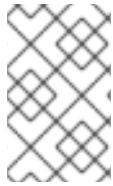
Insights

If enabled, the Insights details are displayed.

Number of subscriptions

The number of subscriptions attached are displayed.

3. Click **Unregister** to remove the registration from the CDN. The original registration details are displayed with a **Not registered** message displayed in the lower-middle part of the window.
4. Click **Done** to return to the **Installation Summary** window.
5. **Connect to Red Hat** displays a *Not registered* message, and **Software Selection** displays a *Red Hat CDN requires registration* message.

**NOTE**

After unregistering, it is possible to register your system again. Click **Connect to Red Hat**. The previously entered details are populated. Edit the original details, or update the fields based on the account, purpose, and connection. Click **Register** to complete.

CHAPTER 3. PREPARING FOR YOUR INSTALLATION

If you are a new Red Hat Enterprise Linux user, it is important that you prepare for your installation by reviewing the information about system requirements, installation boot media, and installation ISO images.

3.1. RECOMMENDED STEPS

Preparing for your RHEL installation consists of several steps.

Steps

1. Check system requirements.
2. Review the installation boot media options.
3. Download the required installation ISO image.
4. Create a bootable installation medium.
5. Prepare the installation source*

*Only required for the Boot ISO (minimal install) image if you are not using the Content Delivery Network (CDN) to download the required software packages.

3.2. SYSTEM REQUIREMENTS

If this is a first-time install of Red Hat Enterprise Linux it is recommended that you review the guidelines provided for system, hardware, security, memory, and RAID before installing.

Additional resources

For more information about securing Red Hat Enterprise Linux, see the [Security hardening](#) document.

3.3. INSTALLATION BOOT MEDIA OPTIONS

There are several options available to boot the Red Hat Enterprise Linux installation program.

Full installation DVD or USB flash drive

Create a full installation DVD or USB flash drive using the **Binary DVD ISO** image. The DVD or USB flash drive can be used as a boot device and as an installation source for installing software packages. Due to the size of the Binary DVD ISO image, a DVD or USB flash drive are the recommended media types.

Minimal installation DVD, CD, or USB flash drive

Create a minimal installation CD, DVD, or USB flash drive using the **Boot ISO** image, which contains only the minimum files necessary to boot the system and start the installation program.



IMPORTANT

If you are not using the Content Delivery Network (CDN) to download the required software packages, the **Boot ISO** image requires an installation source that contains the required software packages.

PXE Server

A *preboot execution environment* (PXE) server allows the installation program to boot over the network. After a system boot, you must complete the installation from a different installation source, such as a local hard drive or a network location.

Additional resources

- For more information about PXE servers, see the [Performing an advanced RHEL installation](#) document.

3.4. TYPES OF INSTALLATION ISO IMAGES

Two types of Red Hat Enterprise Linux 8 installation ISO images are available from the Red Hat Customer Portal.

Binary DVD ISO image file

A full installation program that contains the BaseOS and AppStream repositories and allows you to complete the installation without additional repositories.



IMPORTANT

You can use a Binary DVD for IBM Z to boot the installation program using a SCSI DVD drive, or as an installation source.

Boot ISO image file

The Boot ISO image is a minimal installation that can be used to install RHEL in two different ways:

- When registering and installing RHEL from the Content Delivery Network (CDN).
- As a minimal image that requires access to the BaseOS and AppStream repositories to install software packages. The repositories are part of the Binary DVD ISO image that is available for download from <https://access.redhat.com/home>. Download and unpack the Binary DVD ISO image to access the repositories.

The following table contains information about the images that are available for the supported architectures.

Table 3.1. Boot and installation images

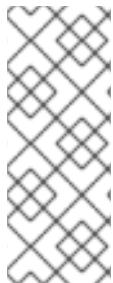
Architecture	Installation DVD	Boot DVD
AMD64 and Intel 64	x86_64 Binary DVD ISO image file	x86_64 Boot ISO image file
ARM 64	AArch64 Binary DVD ISO image file	AArch64 Boot ISO image file
IBM POWER	ppc64le Binary DVD ISO image file	ppc64le Boot ISO image file
IBM Z	s390x Binary DVD ISO image file	s390x Boot ISO image file

3.5. DOWNLOADING THE INSTALLATION ISO IMAGE

This section contains instructions about downloading a Red Hat Enterprise Linux installation image from the Red Hat Customer Portal or by using the **curl** command.

3.5.1. Downloading an ISO image from the Customer Portal

Follow this procedure to download a Red Hat Enterprise Linux 8 ISO image file from the Red Hat Customer Portal.



NOTE

- The Boot ISO image is a minimal image file that supports registering your system, attaching subscriptions, and installing RHEL from the Content Delivery Network (CDN).
- The Binary DVD ISO image file contains all repositories and software packages and does not require any additional configuration.

Prerequisites

- You have an active Red Hat subscription.
- You are logged in to the **Product Downloads** section of the Red Hat Customer Portal at <https://access.redhat.com/downloads>.

Procedure

1. From the **Product Downloads** page, select the **By Category** tab.
2. Click the **Red Hat Enterprise Linux 8** link.
The **Download Red Hat Enterprise Linux** web page opens.
3. From the **Product Variant** drop-down menu, select the variant that you require.
 - a. Optional: Select the **Packages** tab to view the packages contained in the selected variant.
For information on the packages available in Red Hat Enterprise Linux 8, see the [Package Manifest](#) document.
4. The **Version** drop-down menu defaults to the latest version for the selected variant.
5. The **Architecture** drop-down menu displays the supported architecture.
The **Product Software** tab displays the image files, which include:
 - **Red Hat Enterprise Linux Binary DVDimage**.
 - **Red Hat Enterprise Linux Boot ISOimage**.Additional images may be available, for example, preconfigured virtual machine images, but they are beyond the scope of this document.
6. Click **Download Now** beside the ISO image that you require.

3.5.2. Downloading an ISO image using curl

Use the **curl** command to download installation images directly from a specific URL.

Prerequisites

- Verify the curl package is installed:
 - If your distribution uses the **yum** package manager:

```
# yum install curl
```
 - If your distribution uses the **dnf** package manager:

```
# dnf install curl
```
 - If your distribution uses the **apt** package manager:

```
# apt update
# apt install curl
```
- If your Linux distribution does not use yum, dnf, or apt, or if you do not use Linux, download the most appropriate software package from the [curl web site](#).
- You have navigated to the **Product Downloads** section of the Red Hat Customer Portal at <https://access.redhat.com/downloads>, and selected the variant, version, and architecture that you require. You have right-clicked on the required ISO image file, and selected **Copy Link Location** to copy the URL of the ISO image file to your clipboard.

Procedure

1. On the command line, enter a suitable directory, and run the following command to download the file:

```
$ curl --output directory-path/filename.iso 'copied_link_location'
```

Replace *directory-path* with a path to the location where you want to save the file; replace *filename.iso* with the ISO image name as displayed in the Customer Portal; replace *copied_link_location* with the link that you have copied from the Customer Portal.

3.6. CREATING A BOOTABLE INSTALLATION MEDIUM

This section contains information about using the ISO image file that you downloaded in [Section 3.5, “Downloading the installation ISO image”](#) to create a bootable physical installation medium, such as a USB, DVD, or CD.



NOTE

By default, the **inst.stage2=** boot option is used on the installation medium and is set to a specific label, for example, **inst.stage2=hd:LABEL=RHEL8\x86_64**. If you modify the default label of the file system containing the runtime image, or if you use a customized procedure to boot the installation system, you must verify that the label is set to the correct value.

3.6.1. Creating a bootable DVD or CD

You can create a bootable installation DVD or CD using burning software and a CD/DVD burner. The exact steps to produce a DVD or CD from an ISO image file vary greatly, depending on the operating system and disc burning software installed. Consult your system's burning software documentation for the exact steps to burn a CD or DVD from an ISO image file.



WARNING

You can create a bootable DVD or CD using either the Binary DVD ISO image (full install) or the Boot ISO image (minimal install). However, the Binary DVD ISO image is larger than 4.7 GB, and as a result, it might not fit on a single or dual-layer DVD. Check the size of the Binary DVD ISO image file before you proceed. A USB key is recommended when using the Binary DVD ISO image to create bootable installation media.

3.6.2. Creating a bootable USB device on Linux

Follow this procedure to create a bootable USB device on a Linux system.



NOTE

This procedure is destructive and data on the USB flash drive is destroyed without a warning.

Prerequisites

- You have downloaded an installation ISO image as described in [Section 3.5, “Downloading the installation ISO image”](#).
- The **Binary DVD** ISO image is larger than 4.7 GB, so a USB flash drive that is large enough to hold the ISO image is required.

Procedure

1. Connect the USB flash drive to the system.
2. Open a terminal window and run the **dmesg** command:

```
$ dmesg|tail
```

The **dmesg** command returns a log that details all recent events. Messages resulting from the attached USB flash drive are displayed at the bottom of the log. Record the name of the connected device.

3. Switch to user root:

```
$ su -
```

4. Enter your root password when prompted.
5. Find the device node assigned to the drive. In this example, the drive name is **sdd**.

```
# dmesg|tail
[288954.686557] usb 2-1.8: New USB device strings: Mfr=0, Product=1, SerialNumber=2
[288954.686559] usb 2-1.8: Product: USB Storage
[288954.686562] usb 2-1.8: SerialNumber: 000000009225
[288954.712590] usb-storage 2-1.8:1.0: USB Mass Storage device detected
[288954.712687] scsi host6: usb-storage 2-1.8:1.0
[288954.712809] usbcore: registered new interface driver usb-storage
[288954.716682] usbcore: registered new interface driver uas
[288955.717140] scsi 6:0:0:0: Direct-Access Generic STORAGE DEVICE 9228 PQ: 0
ANSI: 0
[288955.717745] sd 6:0:0:0: Attached scsi generic sg4 type 0
[288961.876382] sd 6:0:0:0: sdd Attached SCSI removable disk
```

- Run the **dd** command to write the ISO image directly to the USB device.

```
# dd if=/image_directory/image.iso of=/dev/device
```

Replace */image_directory/image.iso* with the full path to the ISO image file that you downloaded, and replace *device* with the device name that you retrieved with the **dmesg** command. In this example, the full path to the ISO image is **/home/testuser/Downloads/rhel-8-x86_64-boot.iso**, and the device name is **sdd**:

```
# dd if=/home/testuser/Downloads/rhel-8-x86_64-boot.iso of=/dev/sdd
```



NOTE

Ensure that you use the correct device name, and not the name of a partition on the device. Partition names are usually device names with a numerical suffix. For example, **sdd** is a device name, and **sdd1** is the name of a partition on the device **sdd**.

- Wait for the **dd** command to finish writing the image to the device. The data transfer is complete when the **#** prompt appears. When the prompt is displayed, log out of the root account and unplug the USB drive. The USB drive is now ready to be used as a boot device.

3.6.3. Creating a bootable USB device on Windows

Follow the steps in this procedure to create a bootable USB device on a Windows system. The procedure varies depending on the tool. Red Hat recommends using Fedora Media Writer, available for download at <https://github.com/FedoraQt/MediaWriter/releases>.



NOTE

- Fedora Media Writer is a community product and is not supported by Red Hat. You can report any issues with the tool at <https://github.com/FedoraQt/MediaWriter/issues>.
- This procedure is destructive and data on the USB flash drive is destroyed without a warning.

Prerequisites

- You have downloaded an installation ISO image as described in [Section 3.5, “Downloading the installation ISO image”](#).
- The **Binary DVD** ISO image is larger than 4.7 GB, so a USB flash drive that is large enough to hold the ISO image is required.

Procedure

1. Download and install Fedora Media Writer from <https://github.com/FedoraQt/MediaWriter/releases>.



NOTE

To install Fedora Media Writer on Red Hat Enterprise Linux, use the pre-built Flatpak package. You can obtain the package from the official Flatpak repository Flathub.org at <https://flathub.org/apps/details/org.fedoraproject.MediaWriter>.

2. Connect the USB flash drive to the system.
3. Open Fedora Media Writer.
4. From the main window, click **Custom Image** and select the previously downloaded Red Hat Enterprise Linux ISO image.
5. From **Write Custom Image** window, select the drive that you want to use.
6. Click **Write to disk**. The boot media creation process starts. Do not unplug the drive until the operation completes. The operation may take several minutes, depending on the size of the ISO image, and the write speed of the USB drive.
7. When the operation completes, unmount the USB drive. The USB drive is now ready to be used as a boot device.

3.6.4. Creating a bootable USB device on Mac OS X

Follow the steps in this procedure to create a bootable USB device on a Mac OS X system.



NOTE

This procedure is destructive and data on the USB flash drive is destroyed without a warning.

Prerequisites

- You have downloaded an installation ISO image as described in [Section 3.5, “Downloading the installation ISO image”](#).
- The **Binary DVD** ISO image is larger than 4.7 GB, so a USB flash drive that is large enough to hold the ISO image is required.

Procedure

1. Connect the USB flash drive to the system.
2. Identify the device path with the **diskutil list** command. The device path has the format of

`/dev/disknumber`, where number is the number of the disk. The disks are numbered starting at zero (0). Typically, Disk 0 is the OS X recovery disk, and Disk 1 is the main OS X installation. In the following example, the USB device is **disk2**:

```
$ diskutil list
/dev/disk0
#:          TYPE NAME          SIZE  IDENTIFIER
0: GUID_partition_scheme          *500.3 GB  disk0
1:   EFI   EFI           209.7 MB  disk0s1
2: Apple_CoreStorage          400.0 GB  disk0s2
3:   Apple_Boot Recovery HD      650.0 MB  disk0s3
4: Apple_CoreStorage          98.8 GB   disk0s4
5:   Apple_Boot Recovery HD      650.0 MB  disk0s5
/dev/disk1
#:          TYPE NAME          SIZE  IDENTIFIER
0: Apple_HFS YosemiteHD        *399.6 GB  disk1
Logical Volume on disk0s1
8A142795-8036-48DF-9FC5-84506DFBB7B2
Unlocked Encrypted
/dev/disk2
#:          TYPE NAME          SIZE  IDENTIFIER
0: FDisk_partition_scheme          *8.1 GB   disk2
1:   Windows_NTFS SanDisk USB      8.1 GB   disk2s1
```

3. To identify your USB flash drive, compare the NAME, TYPE and SIZE columns to your flash drive. For example, the NAME should be the title of the flash drive icon in the **Finder** tool. You can also compare these values to those in the information panel of the flash drive.
4. Use the **diskutil unmountDisk** command to unmount the flash drive's filesystem volumes:

```
$ diskutil unmountDisk /dev/disknumber
Unmount of all volumes on disknumber was successful
```

When the command completes, the icon for the flash drive disappears from your desktop. If the icon does not disappear, you may have selected the wrong disk. Attempting to unmount the system disk accidentally returns a **failed to unmount** error.

5. Log in as root:

```
$ su -
```

6. Enter your root password when prompted.
7. Use the **dd** command as a parameter of the sudo command to write the ISO image to the flash drive:

```
# sudo dd if=/path/to/image.iso of=/dev/rdisknumber bs=1m>
```



NOTE

Mac OS X provides both a block (`/dev/disk*`) and character device (`/dev/rdisk*`) file for each storage device. Writing an image to the `/dev/rdisknumber` character device is faster than writing to the `/dev/disknumber` block device.

8. To write the `/Users/user_name/Downloads/rhel-8-x86_64-boot.iso` file to the `/dev/rdisk2` device, run the following command:

```
# sudo dd if=/Users/user_name/Downloads/rhel-8-x86_64-boot.iso of=/dev/rdisk2
```

9. Wait for the **dd** command to finish writing the image to the device. The data transfer is complete when the **#** prompt appears. When the prompt is displayed, log out of the root account and unplug the USB drive. The USB drive is now ready to be used as a boot device.

3.7. PREPARING AN INSTALLATION SOURCE

The Boot ISO image file does not include any repositories or software packages; it contains only the installation program and the tools required to boot the system and start the installation. This section contains information about creating an installation source for the Boot ISO image using the Binary DVD ISO image that contains the required repositories and software packages.



IMPORTANT

An installation source is required for the Boot ISO image file only if you decide not to register and install RHEL from the Content Delivery Network (CDN).

3.7.1. Types of installation source

You can use one of the following installation sources for minimal boot images:

- **DVD:** Burn the Binary DVD ISO image to a DVD. The installation program will automatically install the software packages from the DVD.
- **Hard drive or USB drive:** Copy the Binary DVD ISO image to the drive and configure the installation program to install the software packages from the drive. If you use a USB drive, verify that it is connected to the system before the installation begins. The installation program cannot detect media after the installation begins.
 - **Hard drive limitation:** The Binary DVD ISO image on the hard drive must be on a partition with a file system that the installation program can mount. The supported file systems are **xfs**, **ext2**, **ext3**, **ext4**, and **vfat (FAT32)**.



WARNING

On Microsoft Windows systems, the default file system used when formatting hard drives is NTFS. The exFAT file system is also available. However, neither of these file systems can be mounted during the installation. If you are creating a hard drive or a USB drive as an installation source on Microsoft Windows, verify that you formatted the drive as FAT32. Note that the FAT32 file system cannot store files larger than 4 GiB.

In Red Hat Enterprise Linux 8, you can enable installation from a directory on a local hard drive. To do so, you need to copy the contents of the DVD ISO image to a directory on a hard drive and then specify the directory as the installation source instead of the ISO image. For example:

inst.repo=hd:<device>:<path to the directory>

- **Network location:** Copy the Binary DVD ISO image or the installation tree (extracted contents of the Binary DVD ISO image) to a network location and perform the installation over the network using the following protocols:
 - **NFS:** The Binary DVD ISO image is in a Network File System (NFS) share.
 - **HTTPS, HTTP or FTP:** The installation tree is on a network location that is accessible over HTTP, HTTPS or FTP.

3.7.2. Specify the installation source

You can specify the installation source using any of the following methods:

- **Graphical installation:** Select the installation source in the **Installation Source** window of the graphical install.
- **Boot option:** Configure a custom boot option to specify the installation source.
- **Kickstart file:** Use the `install` command in a Kickstart file to specify the installation source. See the [Performing an advanced RHEL installation](#) document for more information.

3.7.3. Ports for network-based installation

The following table lists the ports that must be open on the server providing the files for each type of network-based installation.

Table 3.2. Ports for network-based installation

Protocol used	Ports to open
HTTP	80
HTTPS	443

Protocol used	Ports to open
FTP	21
NFS	2049, 111, 20048
TFTP	69

Additional resources

- See the [Securing networks](#) document for more information.

3.7.4. Creating an installation source on an NFS server

Follow the steps in this procedure to place the installation source on an NFS server. Use this installation method to install multiple systems from a single source, without having to connect to physical media.

Prerequisites

- You have administrator level access to a server with Red Hat Enterprise Linux 8, and this server is on the same network as the system to be installed.
- You have downloaded a Binary DVD image. See [Downloading the installation ISO image](#) from the *Performing a standard RHEL installation* document for more information.
- You have created a bootable CD, DVD, or USB device from the image file. See [Creating installation media](#) from the *Performing a standard RHEL installation* document for more information.
- You have verified that your firewall allows the system you are installing to access the remote installation source. See [Ports for network-based installation](#) from the *Performing a standard RHEL installation* document for more information.

Procedure

1. Install the **nfs-utils** package:

```
# yum install nfs-utils
```

2. Copy the Binary DVD ISO image to a directory on the NFS server.

3. Open the **/etc/exports** file using a text editor and add a line with the following syntax:

```
/exported_directory/ clients
```

4. Replace `/exported_directory/` with the full path to the directory with the ISO image. Replace `clients` with the host name or IP address of the target system, the subnetwork that all target systems can use to access the ISO image, or the asterisk sign (*) if you want to allow any system with network access to the NFS server to use the ISO image. See the **exports(5)** man page for detailed information about the format of this field.
A basic configuration that makes the `/rhel8-install/` directory available as read-only to all clients is:

```
/rhel8-install *
```

5. Save the **/etc/exports** file and exit the text editor.

6. Start the nfs service:

```
# systemctl start nfs-server.service
```

If the service was running before you changed the **/etc/exports** file, run the following command for the running NFS server to reload its configuration:

```
# systemctl reload nfs-server.service
```

The ISO image is now accessible over NFS and ready to be used as an installation source.



NOTE

When configuring the installation source, use **nfs**: as the protocol, the server host name or IP address, the colon sign (:), and the directory holding the ISO image. For example, if the server host name is **myserver.example.com** and you have saved the ISO image in **/rhel8-install/**, specify **nfs:myserver.example.com:/rhel8-install/** as the installation source.

3.7.5. Creating an installation source using HTTP or HTTPS

Follow the steps in this procedure to create an installation source for a network-based installation using an installation tree, which is a directory containing extracted contents of the Binary DVD ISO image and a valid **.treeinfo** file. The installation source is accessed over HTTP or HTTPS.

Prerequisites

- You have administrator level access to a server with Red Hat Enterprise Linux 8, and this server is on the same network as the system to be installed.
- You have downloaded a Binary DVD image. See [Downloading the installation ISO image](#) from the *Performing a standard RHEL installation* document for more information.
- You have created a bootable CD, DVD, or USB device from the image file. See [Creating installation media](#) from the *Performing a standard RHEL installation* document for more information.
- You have verified that your firewall allows the system you are installing to access the remote installation source. See [Ports for network-based installation](#) from the *Performing a standard RHEL installation* document for more information.

Procedure

1. Install the **httpd** package:

```
# yum install httpd
```



WARNING

If your Apache web server configuration enables SSL security, verify that you enable only the TLSv1 protocol, and disable SSLv2 and SSLv3. This is due to the POODLE SSL vulnerability (CVE-2014-3566). See <https://access.redhat.com/solutions/1232413> for details.



IMPORTANT

If you use an HTTPS server with a self-signed certificate, you must boot the installation program with the **noverifyssl** option.

2. Copy the Binary DVD ISO image to the HTTP(S) server.
3. Mount the Binary DVD ISO image, using the **mount** command, to a suitable directory:

```
# mkdir /mnt/rhel8-install/
# mount -o loop,ro -t iso9660 /image_directory/image.iso /mnt/rhel8-install/
```

Replace */image_directory/image.iso* with the path to the Binary DVD ISO image.

4. Copy the files from the mounted image to the HTTP(S) server root. This command creates the **/var/www/html/rhel8-install/** directory with the contents of the image.

```
# cp -r /mnt/rhel8-install/ /var/www/html/
```

This command creates the **/var/www/html/rhel8-install/** directory with the content of the image. Note that some copying methods can skip the **.treeinfo** file which is required for a valid installation source. Running the **cp** command for whole directories as shown in this procedure will copy **.treeinfo** correctly.

5. Start the **httpd** service:

```
# systemctl start httpd.service
```

The installation tree is now accessible and ready to be used as the installation source.



NOTE

When configuring the installation source, use **http://** or **https://** as the protocol, the server host name or IP address, and the directory that contains the files from the ISO image, relative to the HTTP server root. For example, if you are using HTTP, the server host name is **myserver.example.com**, and you have copied the files from the image to **/var/www/html/rhel8-install/**, specify <http://myserver.example.com/rhel8-install/> as the installation source.

Additional resources

- For more information about HTTP servers, see the [Deploying different types of servers](#) document.

3.7.6. Creating an installation source using FTP

Follow the steps in this procedure to create an installation source for a network-based installation using an installation tree, which is a directory containing extracted contents of the Binary DVD ISO image and a valid `.treeinfo` file. The installation source is accessed over FTP.

Prerequisites

- You have administrator level access to a server with Red Hat Enterprise Linux 8, and this server is on the same network as the system to be installed.
- You have downloaded a Binary DVD image. See [Downloading the installation ISO image](#) from the *Performing a standard RHEL installation* document for more information.
- You have created a bootable CD, DVD, or USB device from the image file. See [Creating installation media](#) from the *Performing a standard RHEL installation* document for more information.
- You have verified that your firewall allows the system you are installing to access the remote installation source. See [Ports for network-based installation](#) from the *Performing a standard RHEL installation* document for more information.

Procedure

1. Install the `vsftpd` package by running the following command as root:


```
# yum install vsftpd
```
2. Open and edit the `/etc/vsftpd/vsftpd.conf` configuration file in a text editor.
 - a. Change the line `anonymous_enable=NO` to `anonymous_enable=YES`
 - b. Change the line `write_enable=YES` to `write_enable=NO`.
 - c. Add lines `pasv_min_port=min_port` and `pasv_max_port=max_port`. Replace `min_port` and `max_port` with the port number range used by FTP server in passive mode, e. g. **10021** and **10031**.
This step can be necessary in network environments featuring various firewall/NAT setups.
 - d. Optionally, add custom changes to your configuration. For available options, see the `vsftpd.conf(5)` man page. This procedure assumes that default options are used.



WARNING

If you configured SSL/TLS security in your `vsftpd.conf` file, ensure that you enable only the TLSv1 protocol, and disable SSLv2 and SSLv3. This is due to the POODLE SSL vulnerability (CVE-2014-3566). See <https://access.redhat.com/solutions/1234773> for details.

3. Configure the server firewall.

- a. Enable the firewall:

```
# systemctl enable firewalld
# systemctl start firewalld
```

- b. Enable in your firewall the FTP port and port range from previous step:

```
# firewall-cmd --add-port min_port-max_port/tcp --permanent
# firewall-cmd --add-service ftp --permanent
# firewall-cmd --reload
```

Replace *min_port-max_port* with the port numbers you entered into the **/etc/vsftpd/vsftpd.conf** configuration file.

4. Copy the Binary DVD ISO image to the FTP server.
5. Mount the Binary DVD ISO image, using the mount command, to a suitable directory:

```
# mkdir /mnt/rhel8-install
# mount -o loop,ro -t iso9660 /image-directory/image.iso /mnt/rhel8-install
```

Replace */image-directory/image.iso* with the path to the Binary DVD ISO image.

6. Copy the files from the mounted image to the FTP server root:

```
# mkdir /var/ftp/rhel8-install
# cp -r /mnt/rhel8-install/ /var/ftp/
```

This command creates the **/var/ftp/rhel8-install/** directory with the content of the image. Note that some copying methods can skip the **.treeinfo** file which is required for a valid installation source. Running the **cp** command for whole directories as shown in this procedure will copy **.treeinfo** correctly.

7. Make sure that the correct SELinux context and access mode is set on the copied content:

```
# restorecon -r /var/ftp/rhel8-install
# find /var/ftp/rhel8-install -type f -exec chmod 444 {} \;
# find /var/ftp/rhel8-install -type d -exec chmod 755 {} \;
```

8. Start the **vsftpd** service:

```
# systemctl start vsftpd.service
```

If the service was running before you changed the **/etc/vsftpd/vsftpd.conf** file, restart the service to load the edited file:

```
# systemctl restart vsftpd.service
```

Enable the **vsftpd** service to start during the boot process:

```
# systemctl enable vsftpd
```

The installation tree is now accessible and ready to be used as the installation source.



NOTE

When configuring the installation source, use **ftp://** as the protocol, the server host name or IP address, and the directory in which you have stored the files from the ISO image, relative to the FTP server root. For example, if the server host name is **myserver.example.com** and you have copied the files from the image to **/var/ftp/rhel8-install/**, specify **ftp://myserver.example.com/rhel8-install/** as the installation source.

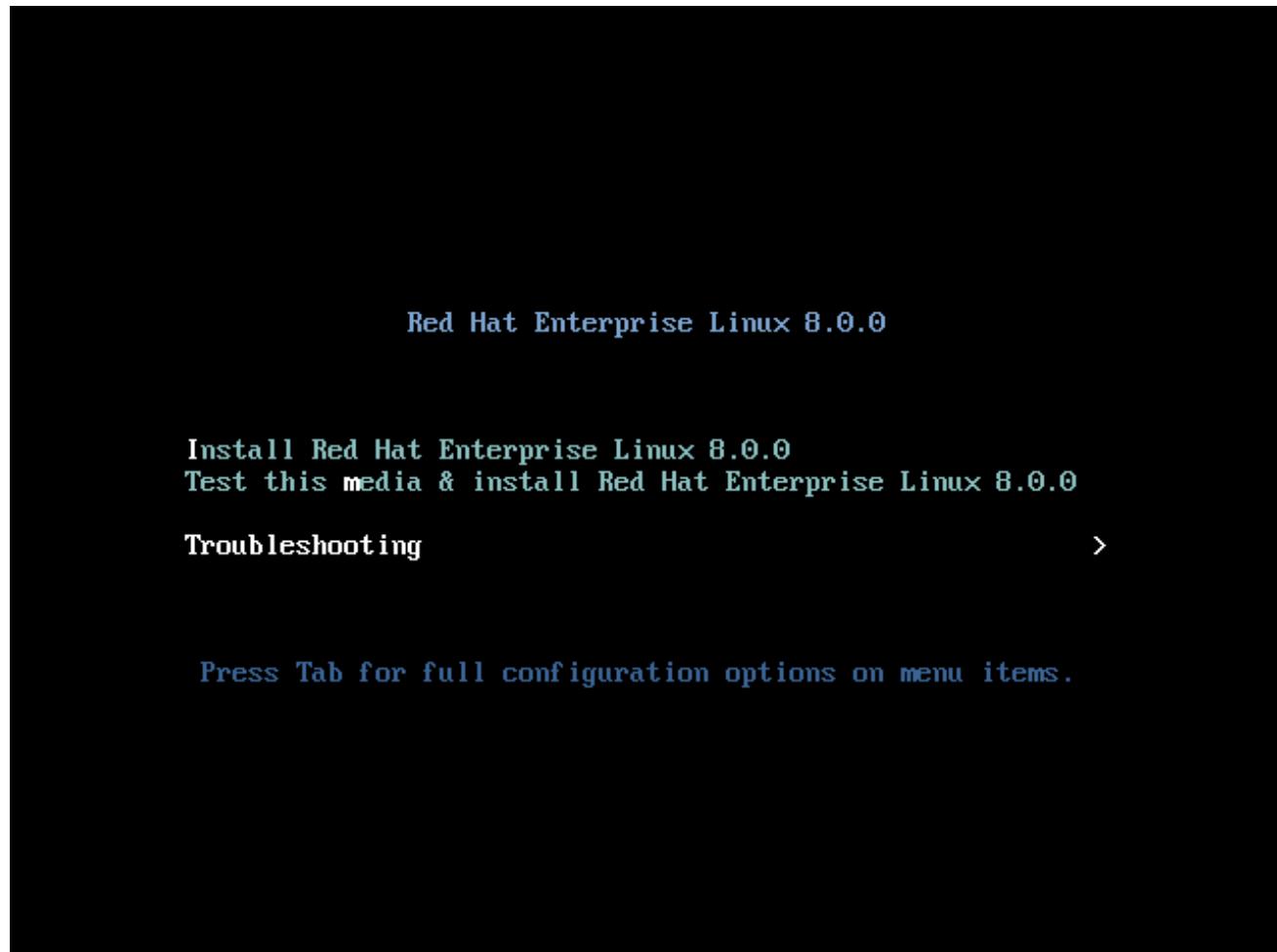
CHAPTER 4. BOOTING THE INSTALLATION

After you have created bootable media you are ready to boot the Red Hat Enterprise Linux installation.

4.1. BOOT MENU

The Red Hat Enterprise Linux boot menu is displayed using **GRand Unified Bootloader version 2** (GRUB2) when your system has completed loading the boot media.

Figure 4.1. Red Hat Enterprise Linux boot menu



The boot menu provides several options in addition to launching the installation program. If you do not make a selection within 60 seconds, the default boot option (highlighted in white) is run. To select a different option, use the arrow keys on your keyboard to make your selection and press **Enter**.

You can customize boot options for a particular menu entry:

- **On BIOS-based systems:** Press the **Tab** key and add custom boot options to the command line. You can also access the **boot:** prompt by pressing the **Esc** key but no required boot options are preset. In this scenario, you must always specify the Linux option before using any other boot options.
- **On UEFI-based systems:** Press the **e** key and add custom boot options to the command line. When ready press **Ctrl+X** to boot the modified option.

Table 4.1. Boot menu options

Boot menu option	Description
Install Red Hat Enterprise Linux 8.x	Use this option to install Red Hat Enterprise Linux using the graphical installation program. For more information, see Section 5.1, "Graphical installation workflow"
Test this media & install Red Hat Enterprise Linux 8.x	Use this option to check the integrity of the installation media. For more information, see Section A.1.4, "Verifying boot media"
Troubleshooting >	Use this option to resolve various installation issues. Press Enter to display its contents.

Table 4.2. Troubleshooting options

Troubleshooting option	Description
Troubleshooting > Install Red Hat Enterprise Linux 8.x in basic graphics mode	Use this option to install Red Hat Enterprise Linux in graphical mode even if the installation program is unable to load the correct driver for your video card. If your screen is distorted when using the Install Red Hat Enterprise Linux 8.x option, restart your system and use this option. For more information, see Section A.1.8, "Cannot boot into the graphical installation"
Troubleshooting > Rescue a Red Hat Enterprise Linux system	Use this option to repair any issues that prevent you from booting. For more information, see Section A.3.8, "Using rescue mode"
Troubleshooting > Run a memory test	Use this option to run a memory test on your system. Press Enter to display its contents. For more information, see Section A.1.3, "Detecting memory faults using the Memtest86 application"
Troubleshooting > Boot from local drive	Use this option to boot the system from the first installed disk. If you booted this disk accidentally, use this option to boot from the hard disk immediately without starting the installation program.

4.2. TYPES OF BOOT OPTIONS

There are two types of boot options; those with an equals "=" sign, and those without an equals "=" sign. Boot options are appended to the boot command line and multiple options must be separated by a single space. Boot options that are specific to the installation program always start with **inst**.

Options with an equals "=" sign

You must specify a value for boot options that use the **=** symbol. For example, the **inst.vncpassword=** option must contain a value, in this case, a password. The correct syntax for this example is **inst.vncpassword=password**.

Options without an equals "==" sign

This boot option does not accept any values or parameters. For example, the **rd.live.check** option forces the installation program to verify the installation media before starting the installation. If this boot option is present, the verification is performed; if the boot option is not present, the verification is skipped.

4.3. EDITING BOOT OPTIONS

This section contains information about the different ways that you can edit boot options from the boot menu. The boot menu opens after you boot the installation media.

Editing the boot: prompt in BIOS

When using the **boot:** prompt, the first option must always specify the installation program image file that you want to load. In most cases, you can specify the image using the keyword. You can specify additional options according to your requirements.

Prerequisites

- You have created bootable installation media (USB, CD or DVD).
- You have booted the installation from the media, and the installation boot menu is open.

Procedure

1. With the boot menu open, press the **Esc** key on your keyboard.
2. The **boot:** prompt is now accessible.
3. Press the **Tab** key on your keyboard to display the help commands.
4. Press the **Enter** key on your keyboard to start the installation with your options. To return from the **boot:** prompt to the boot menu, restart the system and boot from the installation media again.



NOTE

The **boot:** prompt also accepts **dracut** kernel options. A list of options is available in the **dracut.cmdline(7)** man page.

Editing the > prompt

You can use the **>** prompt to edit predefined boot options. For example, select **Test this media and install Red Hat Enterprise Linux 8.1** from the boot menu to display a full set of options.



NOTE

This procedure is for BIOS-based AMD64 and Intel 64 systems.

Prerequisites

- You have created bootable installation media (USB, CD or DVD).

- You have booted the installation from the media, and the installation boot menu is open.

Procedure

1. From the boot menu, select an option and press the **Tab** key on your keyboard. The **>** prompt is accessible and displays the available options.
2. Append the options that you require to the **>** prompt.
3. Press the **Enter** key on your keyboard to start the installation.
4. Press the **Esc** key on your keyboard to cancel editing and return to the boot menu.

Editing the GRUB2 menu

The GRUB2 menu is available on UEFI-based AMD64, Intel 64, and 64-bit ARM systems.

Prerequisites

- You have created bootable installation media (USB, CD or DVD).
- You have booted the installation from the media, and the installation boot menu is open.

Procedure

1. From the boot menu window, select the required option and press the **e** key on your keyboard.
2. Move the cursor to the kernel command line. On UEFI systems, the kernel command line starts with **linuxefi**.
3. Move the cursor to the end of the **linuxefi** kernel command line.
4. Edit the parameters as required. For example, to configure one or more network interfaces, add the **ip=** parameter at the end of the **linuxefi** kernel command line, followed by the required value.
5. When you finish editing, press **Ctrl+X** on your keyboard to start the installation using the specified options.

4.4. BOOTING THE INSTALLATION FROM A USB, CD, OR DVD

Follow the steps in this procedure to boot the Red Hat Enterprise Linux installation using a USB, CD, or DVD. The following steps are generic. Consult your hardware manufacturer's documentation for specific instructions.

Prerequisite

You have created bootable installation media (USB, CD or DVD). See [Section 3.6, "Creating a bootable installation medium"](#) for more information.

Procedure

1. Power off the system to which you are installing Red Hat Enterprise Linux.
2. Disconnect any drives from the system.
3. Power on the system.

4. Insert the bootable installation media (USB, DVD, or CD).
5. Power off the system but do not remove the boot media.
6. Power on the system.

**NOTE**

You might need to press a specific key or combination of keys to boot from the media or configure the Basic Input/Output System (BIOS) of your system to boot from the media. For more information, see the documentation that came with your system.

7. The **Red Hat Enterprise Linux boot** window opens and displays information about a variety of available boot options.
8. Use the arrow keys on your keyboard to select the boot option that you require, and press **Enter** to select the boot option. The **Welcome to Red Hat Enterprise Linux** window opens and you can install Red Hat Enterprise Linux using the graphical user interface.

**NOTE**

The installation program automatically begins if no action is performed in the boot window within 60 seconds.

9. Optional: For UEFI-based systems, press **E** to edit the available boot options. For BIOS-based systems, press the **Tab** key on your keyboard to edit the available boot options. The boot window enters edit mode and you can change the predefined command line to add or remove boot options.
 - a. Press **Enter** to confirm your choice.

4.5. BOOTING THE INSTALLATION FROM A NETWORK USING PXE

When installing Red Hat Enterprise Linux on a large number of systems simultaneously, the best approach is to boot from a PXE server and install from a source in a shared network location. Follow the steps in this procedure to boot the Red Hat Enterprise Linux installation from a network using PXE.

**NOTE**

To boot the installation process from a network using PXE, you must use a physical network connection, for example, Ethernet. You cannot boot the installation process with a wireless connection.

Prerequisites

- You have configured a TFTP server, and there is a network interface in your system that supports PXE. See **Additional resources** for more information.
- You have configured your system to boot from the network interface. This option is in the BIOS, and can be labeled **Network Boot** or **Boot Services**.
- You have verified that the BIOS is configured to boot from the specified network interface. Some BIOS systems specify the network interface as a possible boot device, but do not support

the PXE standard. See your hardware's documentation for more information. When you have properly enabled PXE booting, the system can boot the Red Hat Enterprise Linux installation program without any other media.

Procedure

1. Verify that the network cable is attached. The link indicator light on the network socket should be lit, even if the computer is not switched on.

2. Switch on the system.

Depending on your hardware, some network setup and diagnostic information can be displayed before your system connects to a PXE server. When connected, a menu is displayed according to the PXE server configuration.

3. Press the number key that corresponds to the option that you require.



NOTE

In some instances, boot options are not displayed. If this occurs, press the **Enter** key on your keyboard or wait until the boot window opens.

The **Red Hat Enterprise Linux boot** window opens and displays information about a variety of available boot options.

4. Use the arrow keys on your keyboard to select the boot option that you require, and press **Enter** to select the boot option. The **Welcome to Red Hat Enterprise Linux** window opens and you can install Red Hat Enterprise Linux using the graphical user interface.



NOTE

The installation program automatically begins if no action is performed in the boot window within 60 seconds.

5. Optional: For UEFI-based systems, press **E** to edit the available boot options. For BIOS-based systems, press the **Tab** key on your keyboard to edit the available boot options. The boot window enters edit mode and you can change the predefined command line to add or remove boot options.
 - a. Press **Enter** to confirm your choice.

Additional Resources

- For information about how to prepare to install Red Hat Enterprise Linux from the network using PXE, see the [*Performing an advanced RHEL installation*](#) document.
- Refer to the Boot Options Reference for more information about the list of available boot options you can use on the boot command line.

CHAPTER 5. CUSTOMIZING YOUR INSTALLATION USING THE GUI

This section contains information about customizing your Red Hat Enterprise Linux installation using the Graphical User Interface (GUI). The GUI is the preferred method of installing Red Hat Enterprise Linux when you boot the system from a CD, DVD, or USB flash drive, or from a network using PXE.



NOTE

There may be some variance between the online help and the content that is published on the Customer Portal. For the latest updates, see the installation content on the Customer Portal.

5.1. GRAPHICAL INSTALLATION WORKFLOW

Complete the following steps to install Red Hat Enterprise Linux using the graphical user interface.



NOTE

When installing from a network location or from CDN, you must configure the network before you can proceed.

Steps

1. Configure language and location settings. See [Section 5.2, “Configuring language and location settings”](#) for more information.
2. Configure localization settings. See [Section 5.4, “Configuring localization options”](#) for more information.
3. Configure network, connection to Red Hat, system purpose, installation destination, KDUMP, and security policy. See [Section 5.5, “Configuring system options”](#) for more information.
4. Select the installation source and software packages that you require. See [Section 5.6, “Configuring software options”](#) for more information.
5. Configure storage. See [Section 5.7, “Configuring storage devices”](#) for more information.
6. Start the installation and create a user account and password. See [Section 5.9, “Starting the installation program”](#) for more information.
7. Complete the graphical installation. See [Section 5.9.4, “Graphical installation complete”](#) for more information.

5.2. CONFIGURING LANGUAGE AND LOCATION SETTINGS

The installation program uses the language that you selected during installation.

Prerequisites

1. You created installation media.
2. You specified an installation source if you are using the Boot ISO image file.

3. You booted the installation.

Procedure

1. From the left-hand pane of the **Welcome to Red Hat Enterprise Linux** window, select a language. Alternatively, type your preferred language into the **Search** field.



NOTE

A language is pre-selected by default. If network access is configured, that is, if you booted from a network server instead of local media, the pre-selected language is determined by the automatic location detection feature of the **GeolP** module. If you used the **inst.lang=** option on the boot command line or in your PXE server configuration, then the language that you define with the boot option is selected.

2. From the right-hand pane of the **Welcome to Red Hat Enterprise Linux** window, select a location specific to your region.
3. Click **Continue** to proceed to the [Section 5.3, “The Installation Summary window”](#) window.



IMPORTANT

If you are installing a pre-release version of Red Hat Enterprise Linux, a warning message is displayed about the pre-release status of the installation media. Click **I want to proceed** to continue with the installation, or **I want to exit** to quit the installation and reboot the system.

Additional resources

For information about how to change language and location settings during the installation program, see [Section 5.4, “Configuring localization options”](#)

5.3. THE INSTALLATION SUMMARY WINDOW

The **Installation Summary** window is the central location for the Red Hat Enterprise Linux 8 installation program.

Installation summary

The **Installation Summary** window contains three categories:

- **LOCALIZATION:** You can configure Keyboard, Language Support, and Time and Date.
- **SOFTWARE:** You can configure Installation Source and Software Selection.
- **SYSTEM:** You can configure Installation Destination, Connect to Red Hat, System Purpose, Network & Host Name, KDUMP, Network and Host Name, and Security Policy.

A category has a different status depending on where it is in the installation program.

Table 5.1. Category status

Category status	Status	Description
Warning symbol type 1	Yellow triangle with an exclamation mark and red text	Requires attention before installation. For example, Network & Host Name requires attention before you can register and download from the Content Delivery Network (CDN).
Warning symbol type 2	Grayed out and with a warning symbol (yellow triangle with an exclamation mark)	The installation program is configuring a category and you must wait for it to finish before accessing the window.



NOTE

A warning message is displayed at the bottom of the **Installation Summary** window and the **Begin Installation** button is disabled until you configure all of the required categories.

5.4. CONFIGURING LOCALIZATION OPTIONS

This section contains information about configuring your keyboard, language support, and time and date settings.



IMPORTANT

If you use a layout that cannot accept Latin characters, such as **Russian**, add the **English (United States)** layout and configure a keyboard combination to switch between the two layouts. If you select a layout that does not have Latin characters, you might be unable to enter a valid **root** password and user credentials later in the installation process. This might prevent you from completing the installation.

5.4.1. Configuring keyboard, language, and time and date settings



NOTE

Keyboard, Language, and Time and Date Settings are configured by default as part of [Section 5.2, "Configuring language and location settings"](#). To change any of the settings, complete the following steps, otherwise proceed to [Section 5.6, "Configuring software options"](#).

Procedure: Configuring keyboard settings

1. From the **Installation Summary** window, click **Keyboard**. The default layout depends on the option selected in [Section 5.2, "Configuring language and location settings"](#).
 - a. Click **+** to open the **Add a Keyboard Layout** window and change to a different layout.
 - b. Select a layout by browsing the list or use the **Search** field.
 - c. Select the required layout and click **Add**. The new layout appears under the default layout.

- d. Click **Options** to optionally configure a keyboard switch that you can use to cycle between available layouts. The **Layout Switching Options** window opens.
- e. To configure key combinations for switching, select one or more key combinations and click **OK** to confirm your selection.



NOTE

When you select a layout, click the **Keyboard** button to open a new dialog box that displays a visual representation of the selected layout.

- f. Click **Done** to apply the settings and return to [Section 5.3, "The Installation Summary window"](#).

Procedure: Configuring language settings

1. From the **Installation Summary** window, click **Language Support**. The **Language Support** window opens. The left pane lists the available language groups. If at least one language from a group is configured, a check mark is displayed and the supported language is highlighted.
 - a. From the left pane, click a group to select additional languages, and from the right pane, select regional options. Repeat this process for languages that you require.
 - b. Click **Done** to apply the changes and return to [Section 5.3, "The Installation Summary window"](#).

Procedure: Configuring time and date settings

1. From the **Installation Summary** window, click **Time & Date**. The **Time & Date** window opens.



NOTE

The **Time & Date** settings are configured by default based on the settings you selected in [Section 5.2, "Configuring language and location settings"](#).

The list of cities and regions come from the Time Zone Database (**tzdata**) public domain that is maintained by the Internet Assigned Numbers Authority (IANA). Red Hat can not add cities or regions to this database. You can find more information at the [IANA official website](#).

- a. From the **Region** drop-down menu, select a region.



NOTE

Select **Etc** as your region to configure a time zone relative to Greenwich Mean Time (GMT) without setting your location to a specific region.

- b. From the **City** drop-down menu, select the city, or the city closest to your location in the same time zone.
- c. Toggle the **Network Time** switch to enable or disable network time synchronization using the Network Time Protocol (NTP).

**NOTE**

Enabling the Network Time switch keeps your system time correct as long as the system can access the internet. By default, one NTP pool is configured; you can add a new option, or disable or remove the default options by clicking the gear wheel button next to the **Network Time** switch.

- d. Click **Done** to apply the changes and return to [Section 5.3, “The Installation Summary window”](#).

**NOTE**

If you disable network time synchronization, the controls at the bottom of the window become active, allowing you to set the time and date manually.

5.5. CONFIGURING SYSTEM OPTIONS

This section contains information about configuring Installation Destination, KDUMP, Network and Host Name, Security Policy, and System Purpose.

5.5.1. Configuring installation destination

Use the **Installation Destination** window to configure the storage options, for example, the disks that you want to use as the installation target for your Red Hat Enterprise Linux installation. You must select at least one disk.

**WARNING**

Back up your data if you plan to use a disk that already contains data. For example, if you want to shrink an existing Microsoft Windows partition and install Red Hat Enterprise Linux as a second system, or if you are upgrading a previous release of Red Hat Enterprise Linux. Manipulating partitions always carries a risk. For example, if the process is interrupted or fails for any reason data on the disk can be lost.



IMPORTANT

Special cases

- Some BIOS types do not support booting from a RAID card. In these instances, the **/boot** partition must be created on a partition outside of the RAID array, such as on a separate hard drive. It is necessary to use an internal hard drive for partition creation with problematic RAID cards. A **/boot** partition is also necessary for software RAID setups. If you choose to partition your system automatically, you should manually edit your **/boot** partition.
- To configure the Red Hat Enterprise Linux boot loader to *chain load* from a different boot loader, you must specify the boot drive manually by clicking the **Full disk summary and bootloader** link from the **Installation Destination** window.
- When you install Red Hat Enterprise Linux on a system with both multipath and non-multipath storage devices, the automatic partitioning layout in the installation program creates volume groups that contain a mix of multipath and non-multipath devices. This defeats the purpose of multipath storage. It is recommended that you select either multipath or non-multipath devices on the **Installation Destination** window. Alternatively, proceed to manual partitioning.

Prerequisite

The **Installation Summary** window is open.

Procedure

1. From the **Installation Summary** window, click **Installation Destination**. The **Installation Destination** window opens.
 - a. From the **Local Standard Disks** section, select the storage device that you require; a white check mark indicates your selection. Disks without a white check mark are not used during the installation process; they are ignored if you choose automatic partitioning, and they are not available in manual partitioning.



NOTE

All locally available storage devices (SATA, IDE and SCSI hard drives, USB flash and external disks) are displayed under **Local Standard Disks**. Any storage devices connected after the installation program has started are not detected. If you use a removable drive to install Red Hat Enterprise Linux, your system is unusable if you remove the device.

- a. Optional: Click the **Refresh** link in the lower right-hand side of the window if you want to configure additional local storage devices to connect new hard drives. The **Rescan Disks** dialog box opens.



NOTE

All storage changes that you make during the installation are lost when you click **Rescan Disks**.

- i. Click **Rescan Disks** and wait until the scanning process completes.

- ii. Click **OK** to return to the **Installation Destination** window. All detected disks including any new ones are displayed under the **Local Standard Disks** section.
2. Optional: To add a specialized storage device, click **Add a disk...**. The **Storage Device Selection** window opens and lists all storage devices that the installation program has access to. For information about how to add a specialized disk, see [Section 5.7.3, "Using advanced storage options"](#).
3. Optional: Under **Storage Configuration**, select the **Automatic** radio button.



IMPORTANT

Automatic partitioning is the **recommended** method of partitioning your storage. You can also configure custom partitioning, for more details see [Section 5.8, "Configuring manual partitioning"](#)

- 4. Optional: To reclaim space from an existing partitioning layout, select the **I would like to make additional space available** check box. For example, if a disk you want to use already contains a different operating system and you want to make this system's partitions smaller to allow more room for Red Hat Enterprise Linux.
- 5. Optional: Select **Encrypt my data** to encrypt all partitions except the ones needed to boot the system (such as **/boot**) using *Linux Unified Key Setup* (LUKS). Encrypting your hard drive is recommended.
 - a. If you selected **Encrypt my data** the **Disk Encryption Passphrase** dialog box opens.
 - i. Type your passphrase in the **Passphrase** and **Confirm** fields.
 - ii. Click **Save Passphrase** to complete disk encryption.



WARNING

If you lose the LUKS passphrase, any encrypted partitions and their data is completely inaccessible. There is no way to recover a lost passphrase. However, if you perform a Kickstart installation, you can save encryption passphrases and create backup encryption passphrases during the installation. See the [Performing an advanced RHEL installation](#) document for information.

6. Optional: Click the **Full disk summary and bootloader** link in the lower left-hand side of the window to select which storage device contains the boot loader. For more information, see [Section 5.5.1.1, "Configuring boot loader"](#).



NOTE

In most cases it is sufficient to leave the boot loader in the default location. Some configurations, for example, systems that require chain loading from another boot loader require the boot drive to be specified manually.

7. Click **Done**.

- If you selected **automatic partitioning** and **I would like to make additional space available**, or if there is not enough free space on your selected hard drives to install Red Hat Enterprise Linux, the **Reclaim Disk Space** dialog box opens when you click **Done**, and lists all configured disk devices and all partitions on those devices. The dialog box displays information about how much space the system needs for a minimal installation and how much space you have reclaimed.



WARNING

If you **delete** a partition, all data on that partition is lost. If you want to preserve your data, use the **Shrink** option, not the **Delete** option.

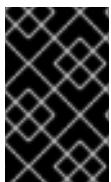
- Review the displayed list of available storage devices. The **Reclaimable Space** column shows how much space can be reclaimed from each entry.
- To reclaim space, select a disk or partition, and click either the **Delete** button to delete that partition, or all partitions on a selected disk, or click **Shrink** to use free space on a partition while preserving the existing data.



NOTE

Alternatively, you can click **Delete all**, this deletes all existing partitions on all disks and makes this space available to Red Hat Enterprise Linux. Existing data on all disks is lost.

- Click **Reclaim space** to apply the changes and return to [Section 5.3, “The Installation Summary window”](#).



IMPORTANT

No disk changes are made until you click **Begin Installation** on the **Installation Summary** window. The **Reclaim Space** dialog only marks partitions for resizing or deletion; no action is performed.

5.5.1.1. Configuring boot loader

Red Hat Enterprise Linux uses GRand Unified Bootloader version 2 (**GRUB2**) as the boot loader for AMD64 and Intel 64, IBM Power Systems, and ARM. For IBM Z, the **zipl** boot loader is used.

The boot loader is the first program that runs when the system starts and is responsible for loading and transferring control to an operating system. **GRUB2** can boot any compatible operating system (including Microsoft Windows) and can also use chain loading to transfer control to other boot loaders for unsupported operating systems.

**WARNING**

Installing **GRUB2** may overwrite your existing boot loader.

If an operating system is already installed, the Red Hat Enterprise Linux installation program attempts to automatically detect and configure the boot loader to start the other operating system. If the boot loader is not detected, you can manually configure any additional operating systems after you finish the installation.

If you are installing a Red Hat Enterprise Linux system with more than one disk, you might want to manually specify the disk where you want to install the boot loader.

Procedure

1. From the **Installation Destination** window, click the **Full disk summary and bootloader** link. The **Selected Disks** dialog box opens. The boot loader is installed on the device of your choice, or on a UEFI system; the **EFI system partition** is created on the target device during guided partitioning.
2. To change the boot device, select a device from the list and click **Set as Boot Device**. You can set only one device as the boot device.
3. To disable a new boot loader installation, select the device currently marked for boot and click **Do not install boot loader**. This ensures **GRUB2** is not installed on any device.

**WARNING**

If you choose not to install a boot loader, you cannot boot the system directly and you must use another boot method, such as a standalone commercial boot loader application. Use this option only if you have another way to boot your system.

The boot loader may also require a special partition to be created, depending on if your system uses BIOS or UEFI firmware, or if the boot drive has a *GUID Partition Table* (GPT) or a **Master Boot Record** (MBR, also known as **msdos**) label. If you use automatic partitioning, the installation program creates the partition.

5.5.2. Configuring Kdump

Kdump is a kernel crash-dumping mechanism. In the event of a system crash, **Kdump** captures the contents of the system memory at the moment of failure. This captured memory can be analyzed to find the cause of the crash. If **Kdump** is enabled, it must have a small portion of the system's memory (RAM) reserved to itself. This reserved memory is not accessible to the main kernel.

Procedure

1. From the **Installation Summary** window, click **Kdump**. The **Kdump** window opens.
2. Select the **Enable kdump** check box.
3. Select either the **Automatic** or **Manual** memory reservation setting.
 - a. If you select **Manual**, enter the amount of memory (in megabytes) that you want to reserve in the **Memory to be reserved** field using the + and - buttons. The **Usable System Memory** readout below the reservation input field shows how much memory is accessible to your main system after reserving the amount of RAM that you select.
4. Click **Done** to apply the settings and return to [Section 5.3, “The Installation Summary window”](#).



NOTE

The amount of memory that you reserve is determined by your system architecture (AMD64 and Intel 64 have different requirements than IBM Power) as well as the total amount of system memory. In most cases, automatic reservation is satisfactory.



IMPORTANT

Additional settings, such as the location where kernel crash dumps will be saved, can only be configured after the installation using either the **system-config-kdump** graphical interface, or manually in the **/etc/kdump.conf** configuration file.

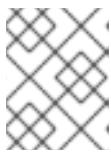
5.5.3. Configuring network and host name options

Use the **Network and Host name** window to configure network interfaces. Options that you select here are available both during the installation for tasks such as downloading packages from a remote location, and on the installed system.

Follow the steps in this procedure to configure your network and host name.

Procedure

1. From the **Installation Summary** window, click **Network and Host Name***.
2. From the list in the left-hand pane, select an interface. The details are displayed in the right-hand pane.
3. Toggle the **ON/OFF** switch to enable or disable the selected interface.



NOTE

The installation program automatically detects locally accessible interfaces, and you cannot add or remove them manually.

4. Click **+** to add a virtual network interface, which can be either: Team, Bond, Bridge, or VLAN.
5. Click **-** to remove a virtual interface.
6. Click **Configure** to change settings such as IP addresses, DNS servers, or routing configuration for an existing interface (both virtual and physical).
7. Type a host name for your system in the **Host Name** field.



NOTE

- There are several types of network device naming standards used to identify network devices with persistent names, for example, **em1** and **wl3sp0**. For information about these standards, see the [Configuring and managing networking](#) document.
- The host name can be either a fully-qualified domain name (FQDN) in the format *hostname.domainname*, or a short host name with no domain name. Many networks have a Dynamic Host Configuration Protocol (DHCP) service that automatically supplies connected systems with a domain name. To allow the DHCP service to assign the domain name to this machine, specify only the short host name. The value **localhost.localdomain** means that no specific static host name for the target system is configured, and the actual host name of the installed system is configured during the processing of the network configuration, for example, by **NetworkManager** using DHCP or DNS.

8. Click **Apply** to apply the host name to the environment.

5.5.3.1. Adding a virtual network interface

Follow the steps in this procedure to add a virtual network interface.

Procedure

1. From the **Network & Host name** window, click the **+** button to add a virtual network interface. The **Add a device** dialog opens.
2. Select one of the four available types of virtual interfaces:
 - **Bond**: NIC (*Network Interface Controller*) Bonding, a method to bind multiple physical network interfaces together into a single bonded channel.
 - **Bridge**: Represents NIC Bridging, a method to connect multiple separate networks into one aggregate network.
 - **Team**: NIC Teaming, a new implementation to aggregate links, designed to provide a small kernel driver to implement the fast handling of packet flows, and various applications to do everything else in user space.
 - **Vlan** (*Virtual LAN*): A method to create multiple distinct broadcast domains which are mutually isolated.
3. Select the interface type and click **Add**. An editing interface dialog box opens, allowing you to edit any available settings for your chosen interface type. For more information see [Section 5.5.3.2, “Editing network interface configuration”](#).
4. Click **Save** to confirm the virtual interface settings and return to the **Network & Host name** window.

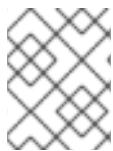


NOTE

If you need to change the settings of a virtual interface, select the interface and click **Configure**.

5.5.3.2. Editing network interface configuration

This section contains information about the most important settings for a typical wired connection used during installation. Configuration of other types of networks is broadly similar, although the specific configuration parameters might be different.



NOTE

On IBM Z, you cannot add a new connection as the network subchannels need to be grouped and set online beforehand, and this is currently done only in the booting phase.

Procedure

1. To configure a network connection manually, select the interface from the **Network and Host name** window and click **Configure**.
An editing dialog specific to the selected interface opens.



NOTE

The options present depend on the connection type - the available options are slightly different depending on whether the connection type is a physical interface (wired or wireless network interface controller) or a virtual interface (Bond, Bridge, Team, or Vlan) that was previously configured in [Section 5.5.3.1, “Adding a virtual network interface”](#).

5.5.3.3. Enabling or Disabling the Interface Connection

Follow the steps in this procedure to enable or disable an interface connection.

Procedure

1. Click the **General** tab.
2. Select the **Connect automatically with priority** check box to enable connection by default. Keep the default priority setting at **0**.



IMPORTANT

- When enabled on a wired connection, the system automatically connects during startup or reboot. On a wireless connection, the interface attempts to connect to any known wireless networks in range. For further information about NetworkManager, including the **nm-connection-editor** tool, see the [Configuring and managing networking](#) document.
- You can enable or disable all users on the system from connecting to this network using the **All users may connect to this network** option. If you disable this option, only **root** will be able to connect to this network.
- It is not possible to only allow a specific user other than **root** to use this interface, as no other users are created at this point during the installation. If you need a connection for a different user, you must configure it after the installation.

3. Click **Save** to apply the changes and return to the **Network and Host name** window.

5.5.3.4. Setting up Static IPv4 or IPv6 Settings

By default, both IPv4 and IPv6 are set to automatic configuration depending on current network settings. This means that addresses such as the local IP address, DNS address, and other settings are detected automatically when the interface connects to a network. In many cases, this is sufficient, but you can also provide static configuration in the **IPv4 Settings** and **IPv6 Settings** tabs. Complete the following steps to configure IPv4 or IPv6 settings:

Procedure

1. To set static network configuration, navigate to one of the IPv Settings tabs and from the **Method** drop-down menu, select a method other than **Automatic**, for example, **Manual**. The **Addresses** pane is enabled.



NOTE

In the **IPv6 Settings** tab, you can also set the method to **Ignore** to disable IPv6 on this interface.

2. Click **Add** and enter your address settings.
3. Type the IP addresses in the **Additional DNS servers** field; it accepts one or more IP addresses of DNS servers, for example, **10.0.0.1,10.0.0.8**.
4. Select the **Require IPvX addressing for this connection to complete** check box.



NOTE

Select this option in the **IPv4 Settings** or **IPv6 Settings** tabs to allow this connection only if IPv4 or IPv6 was successful. If this option remains disabled for both IPv4 and IPv6, the interface is able to connect if configuration succeeds on either IP protocol.

5. Click **Save** to apply the changes and return to the **Network & Host name** window.

5.5.3.5. Configuring Routes

Complete the following steps to configure routes.

Procedure

1. In the **IPv4 Settings** and **IPv6 Settings** tabs, click **Routes** to configure routing settings for a specific IP protocol on an interface. An editing routes dialog specific to the interface opens.
2. Click **Add** to add a route.
3. Select the **Ignore automatically obtained routes** check box to configure at least one static route and to disable all routes not specifically configured.
4. Select the **Use this connection only for resources on its network** check box to prevent the connection from becoming the default route.



NOTE

This option can be selected even if you did not configure any static routes. This route is used only to access certain resources, such as intranet pages that require a local or VPN connection. Another (default) route is used for publicly available resources. Unlike the additional routes configured, this setting is transferred to the installed system. This option is useful only when you configure more than one interface.

5. Click **OK** to save your settings and return to the editing routes dialog that is specific to the interface.
6. Click **Save** to apply the settings and return to the **Network and Host Name** window.

5.5.3.6. Additional resources

- To learn more about network configuration after installation, see the [Configuring and managing networking](#) document.

5.5.4. Configuring Connect to Red Hat

The Red Hat Content Delivery Network (CDN), available from cdn.redhat.com, is a geographically distributed series of static web servers that contain content and errata that is consumed by systems. The content can be consumed directly, such as using a system registered to Red Hat Subscription Management. The CDN is protected by x.509 certificate authentication to ensure that only valid users have access. When a system is registered to Red Hat Subscription Management, the attached subscriptions govern which subset of the CDN the system can access.

Registering and installing RHEL from the CDN provides the following benefits:

- The CDN installation method supports the Boot ISO and the Binary DVD ISO image files. However, the use of the smaller Boot ISO image file is recommended as it consumes less space than the larger Binary DVD ISO image file.
- The CDN uses the latest packages resulting in a fully up-to-date system right after installation. There is no requirement to install package updates immediately after installation as is often the case when using the Binary DVD ISO image file.
- Integrated support for connecting to Red Hat Insights and enabling System Purpose.

5.5.4.1. Introduction to System Purpose

System Purpose is an optional but recommended feature of the Red Hat Enterprise Linux installation. You use System Purpose to record the intended use of a Red Hat Enterprise Linux 8 system, and ensure that the entitlement server auto-attaches the most appropriate subscription to your system.

Benefits include:

- In-depth system-level information for system administrators and business operations.
- Reduced overhead when determining why a system was procured and its intended purpose.
- Improved customer experience of Subscription Manager auto-attach as well as automated discovery and reconciliation of system usage.

You can enter System Purpose data in one of the following ways:

- During image creation
- During a GUI installation when using **Connect to Red Hat** to register your system and attach your Red Hat subscription
- During a Kickstart installation when using Kickstart automation scripts
- After installation using the **syspurpose** command-line (CLI) tool

To record the intended purpose of your system, you can configure the following components of System Purpose. The selected values are used by the entitlement server upon registration to attach the most suitable subscription for your system.

- **Role**
 - Red Hat Enterprise Linux Server
 - Red Hat Enterprise Linux Workstation
 - Red Hat Enterprise Linux Compute Node
- **Service Level Agreement**
 - Premium
 - Standard
 - Self-Support
- **Usage**
 - Production
 - Development/Test
 - Disaster Recovery

Additional resources

- For more information about Image Builder, see the [Composing a customized RHEL system image](#) document.
- For more information about Kickstart, see the [Performing an advanced RHEL installation](#) document.
- For more information about Subscription Manager, see the [Using and Configuring Red Hat Subscription Manager](#) document.

5.5.4.2. Configuring Connect to Red Hat options

Use the following procedure to configure the Connect to Red Hat options in the GUI.



NOTE

You can register to the CDN using either your Red Hat account or your activation key details.

Procedure

1. Click **Account**.
 - a. Enter your Red Hat Customer Portal username and password details.
2. Optional: Click **Activation Key**.
 - a. Enter your organization ID and activation key. You can enter more than one activation key, separated by a comma, as long as the activation keys are registered to your subscription.
3. Select the **Set System Purpose** check box. System Purpose enables the entitlement server to determine and automatically attach the most appropriate subscription to satisfy the intended use of your RHEL 8 system.
 - a. Select the required **Role**, **SLA**, and **Usage** from the corresponding drop-down lists.
4. The **Connect to Red Hat Insights** check box is enabled by default. Clear the check box if you do not want to connect to Red Hat Insights.



NOTE

Red Hat Insights is a Software-as-a-Service (SaaS) offering that provides continuous, in-depth analysis of registered Red Hat-based systems to proactively identify threats to security, performance and stability across physical, virtual and cloud environments, and container deployments.

5. Optional: Expand **Options**.
 - a. Select the **Use HTTP proxy** check box if your network environment only allows external Internet access or access to content servers through an HTTP proxy. Clear the **Use HTTP proxy** check box if an HTTP proxy is not used.
 - b. If you are running Satellite Server or performing internal testing, select the **Custom server URL** and **Custom base URL** check boxes and enter the required details.



IMPORTANT

- The **Custom server URL** field does not require the HTTP protocol, for example **nameofhost.com**. The **Custom base URL** field does require the HTTP protocol, for example **http://nameofhost.com**.
- To change the **Custom base URL** after registration, you must unregister, provide the new details, and then re-register.

6. Click **Register** to register the system. When the system is successfully registered and subscriptions are attached, the **Connect to Red Hat** window displays the attached subscription details.



NOTE

Depending on the amount of subscriptions, the registration and attachment process might take up to a minute to complete.

7. Click **Done** to return to the **Installation Summary** window.

- A *Registered* message is displayed under **Connect to Red Hat**

5.5.4.3. Installation source repository after system registration

The installation source repository used after system registration is dependent on how the system was booted.

System booted from the Boot ISO or the Binary DVD ISO image file

If you booted the RHEL installation using either the **Boot ISO** or the **Binary DVD ISO** image file with the default boot parameters, the installation program automatically switches the installation source repository to the CDN after registration.

System booted with the **inst.repo=<URL>** boot parameter

If you booted the RHEL installation with the **inst.repo=<URL>** boot parameter, the installation program does not automatically switch the installation source repository to the CDN after registration. If you want to use the CDN to install RHEL, you must manually switch the installation source repository to the CDN by selecting the **Red Hat CDN** option in the **Installation Source** window of the graphical installation. If you do not manually switch to the CDN, the installation program installs the packages from the repository specified on the kernel command line.



IMPORTANT

- You can switch the installation source repository to the CDN using the **rhsm** Kickstart command only if you do not specify an installation source using **inst.repo=** on the kernel command line or the **url** command in the Kickstart file. You must use **inst.stage2=<URL>** on the kernel command line to fetch the installation image, but not specify the installation source.
- An installation source URL specified using a boot option or included in a Kickstart file takes precedence over the CDN, even if the Kickstart file contains the **rhsm** command with valid credentials. The system is registered, but it is installed from the URL installation source. This ensures that earlier installation processes operate as normal.

5.5.4.4. Verifying your system registration from the CDN

Use this procedure to verify that your system is registered to the CDN using the GUI.



WARNING

You can only verify your registration from the CDN if you have **not** clicked the **Begin Installation** button from the **Installation Summary** window. Once the **Begin Installation** button is clicked, you cannot return to the **Installation Summary** window to verify your registration.

Prerequisite

- You have completed the registration process as documented in the [Section 2.3.2, “Registering and installing RHEL from the CDN”](#) and *Registered* is displayed under **Connect to Red Hat** on the **Installation Summary** window.

Procedure

1. From the **Installation Summary** window, select **Connect to Red Hat**
2. The window opens and displays a registration summary:

Method

The registered account name or activation keys are displayed.

System Purpose

If set, the role, SLA, and usage details are displayed.

Insights

If enabled, the Insights details are displayed.

Number of subscriptions

The number of subscriptions attached are displayed.

3. Verify that the registration summary matches the details that were entered.

5.5.4.5. Unregistering your system from the CDN

Use this procedure to unregister your system from the CDN using the GUI.



WARNING

- You can unregister from the CDN if you have **not** clicked the **Begin Installation** button from the **Installation Summary** window. Once the **Begin Installation** button is clicked, you cannot return to the Installation Summary window to unregister your registration.
- When unregistering, the installation program switches to the first available repository, in the following order:
 - a. The URL used in the `inst.repo=<URL>` boot parameter on the kernel command line.
 - b. An automatically detected repository on the installation media (USB or DVD).

Prerequisite

- You have completed the registration process as documented in the [Section 2.3.2, “Registering and installing RHEL from the CDN”](#) and *Registered* is displayed under **Connect to Red Hat** on the **Installation Summary** window.

Procedure

1. From the **Installation Summary** window, select **Connect to Red Hat**
2. The **Connect to Red Hat** window opens and displays a registration summary:

Method

The registered account name or activation keys used are displayed.

System Purpose

If set, the role, SLA, and usage details are displayed.

Insights

If enabled, the Insights details are displayed.

Number of subscriptions

The number of subscriptions attached are displayed.

3. Click **Unregister** to remove the registration from the CDN. The original registration details are displayed with a **Not registered** message displayed in the lower-middle part of the window.
4. Click **Done** to return to the **Installation Summary** window.
5. **Connect to Red Hat** displays a *Not registered* message, and **Software Selection** displays a *Red Hat CDN requires registration* message.



NOTE

After unregistering, it is possible to register your system again. Click **Connect to Red Hat**. The previously entered details are populated. Edit the original details, or update the fields based on the account, purpose, and connection. Click **Register** to complete.

Related information

- For information about Red Hat Insights, see the [Red Hat Insights product documentation](#).
- For information about Activation Keys, see the [Understanding Activation Keys](#) chapter of the [Using Red Hat Subscription Management](#) document.
- For information about how to set up an HTTP proxy for Subscription Manager, see the [Using an HTTP proxy](#) chapter of the [Using and Configuring Red Hat Subscription Manager](#) document.

5.5.5. Configuring Security Policy

This section contains information about the Red Hat Enterprise Linux 8 security policy and how to configure it for use on your system.

5.5.5.1. About security policy

The Red Hat Enterprise Linux security policy adheres to restrictions and recommendations (compliance policies) defined by the Security Content Automation Protocol (SCAP) standard. The packages are automatically installed. However, by default, no policies are enforced and therefore no checks are performed during or after installation unless specifically configured.

Applying a security policy is not a mandatory feature of the installation program. If you apply a security policy to the system, it is installed using restrictions and recommendations defined in the profile that you selected. The **openscap-scanner** package is added to your package selection, providing a preinstalled tool for compliance and vulnerability scanning. After the installation finishes, the system is automatically scanned to verify compliance. The results of this scan are saved to the **/root/openscap_data** directory on the installed system. You can also load additional profiles from an HTTP, HTTPS, or FTP server.

5.5.5.2. Configuring a security policy

Complete the following steps to configure a security policy.

Prerequisite

The **Installation Summary** window is open.

Procedure

1. From the **Installation Summary** window, click **Security Policy**. The **Security Policy** window opens.
2. To enable security policies on the system, toggle the **Apply security policy** switch to **ON**.
3. Select one of the profiles listed in the top pane.
4. Click **Select profile**.

Profile changes that you must apply before installation appear in the bottom pane.



NOTE

The default profiles do not require changes before installation. However, loading a custom profile can require pre-installation tasks.

5. Click **Change content** to use a custom profile. A separate window opens allowing you to enter a URL for valid security content.
 - a. Click **Fetch** to retrieve the URL.
 - b. Click **Use SCAP Security Guide** to return to the **Security Policy** window.



NOTE

You can load custom profiles from an **HTTP**, **HTTPS**, or **FTP** server. Use the full address of the content including the protocol, such as **http://**. A network connection must be active before you can load a custom profile. The installation program detects the content type automatically.

6. Click **Done** to apply the settings and return to the **Installation Summary** window.

5.5.5.3. Related information

- **scap-security-guide(8)** – The manual page for the **scap-security-guide** project contains information about SCAP security profiles, including examples on how to utilize the provided benchmarks using the OpenSCAP utility.
- Red Hat Enterprise Linux security compliance information is available in the [Security hardening](#) document.

5.6. CONFIGURING SOFTWARE OPTIONS

This section contains information about configuring your installation source and software selection settings, and activating a repository.

5.6.1. Configuring installation source

Complete the steps in this procedure to configure an installation source from either auto-detected installation media, Red Hat CDN, or the network.



NOTE

When the **Installation Summary** window first opens, the installation program attempts to configure an installation source based on the type of media that was used to boot the system. The full Red Hat Enterprise Linux Server DVD configures the source as local media.

Prerequisites

- You have downloaded the full installation image.
- You have created a bootable physical media.
- The **Installation Summary** window is open.

Procedure

1. From the **Installation Summary** window, click **Installation Source**. The **Installation Source** window opens.
 - a. Review the **Auto-detected installation media** section to verify the details. This option is selected by default if you started the installation program from media containing an installation source, for example, a DVD.
 - b. Click **Verify** to check the media integrity.
 - c. Review the **Additional repositories** section and note that the **AppStream** checkbox is selected by default.



IMPORTANT

- **No additional configuration is necessary as the BaseOS and AppStream repositories are installed as part of the full installation image.**
- **Do not disable the AppStream repository check box if you want a full Red Hat Enterprise Linux 8 installation.**

2. Optional: Select the **Red Hat CDN** option to register your system, attach RHEL subscriptions, and install RHEL from the Red Hat Content Delivery Network (CDN). For more information, see the *Registering and installing RHEL from the CDN* section.
3. Optional: Select the **On the network** option to download and install packages from a network location instead of local media.



NOTE

- If you do not want to download and install additional repositories from a network location, proceed to [Section 5.6.2, "Configuring software selection"](#).
- This option is available only when a network connection is active. See [Section 5.5.3, "Configuring network and host name options"](#) for information about how to configure network connections in the GUI.

- a. Select the **On the network** drop-down menu to specify the protocol for downloading packages. This setting depends on the server that you want to use.



WARNING

The AppStream repository check box is disabled if you select **On the network** and then decide to revert to **Auto-detected installation**. You must select the AppStream check box to enable the AppStream repository.

- b. Type the server address (without the protocol) into the address field. If you choose NFS, a second input field opens where you can specify custom **NFS mount options**. This field accepts options listed in the **nfs(5)** man page.



IMPORTANT

When selecting an NFS installation source, you must specify the address with a colon (:) character separating the host name from the path. For example:

server.example.com:/path/to/directory



NOTE

The following steps are optional and are only required if you use a proxy for network access.

- c. Click **Proxy setup...** to configure a proxy for an HTTP or HTTPS source.
- d. Select the **Enable HTTP proxy** check box and type the URL into the **Proxy Host** field.
- e. Select the **Use Authentication** check box if the proxy server requires authentication.
- f. Type in your user name and password.
- g. Click **OK** to finish the configuration and exit the **Proxy Setup...** dialog box.

**NOTE**

If your HTTP or HTTPS URL refers to a repository mirror menu, select the required option from the **URL type** drop-down list. All environments and additional software packages are available for selection when you finish configuring the sources.

4. Click **+** to add a repository.
5. Click **-** to delete a repository.
6. Click the arrow icon to revert the current entries to the setting when you opened the **Installation Source** window.
7. To activate or deactivate a repository, click the check box in the **Enabled** column for each entry in the list.

**NOTE**

You can name and configure your additional repository in the same way as the primary repository on the network.

8. Click **Done** to apply the settings and return to the **Installation Summary** window.

5.6.2. Configuring software selection

Use the **Software Selection** window to select the software packages that you require. The packages are organized by Base Environment and Additional Software.

- **Base Environment** contains predefined packages. You can select only one base environment, and availability is dependent on the installation ISO image that is used as the installation source.
- **Additional Software for Selected Environment** contains additional software packages for the base environment. You can select multiple software packages.

Use a predefined environment and additional software to customize your system. However, in a standard installation, you cannot select individual packages to install. To view the packages contained in a specific environment, see the **repository/repodata/*-comps-repository.architecture.xml** file on your installation source media (DVD, CD, USB). The XML file contains details of the packages installed as part of a base environment. Available environments are marked by the **<environment>** tag, and additional software packages are marked by the **<group>** tag.

If you are unsure about which packages to install, Red Hat recommends that you select the **Minimal Install** base environment. Minimal install installs a basic version of Red Hat Enterprise Linux with only a minimal amount of additional software. After the system finishes installing and you log in for the first time, you can use the **Yum package manager** to install additional software. For more information about Yum package manager, see the [Configuring basic system settings](#) document.



NOTE

- The **yum group list** command lists all package groups from yum repositories. See the [Configuring basic system settings](#) document for more information.
- If you need to control which packages are installed, you can use a Kickstart file and define the packages in the **%packages** section. See the [Performing an advanced RHEL installation](#) document for information about installing Red Hat Enterprise Linux using Kickstart.

Prerequisites

- You have configured the installation source.
- The installation program downloaded package metadata.
- The **Installation Summary** window is open.

Procedure

1. From the **Installation Summary** window, click **Software Selection**. The **Software Selection** window opens.
2. From the **Base Environment** pane, select a base environment. You can select only one base environment.



NOTE

The **Server with GUI** base environment is the default base environment and it launches the **Initial Setup** application after the installation completes and you restart the system.

3. From the **Additional Software for Selected Environment** pane, select one or more options.
4. Click **Done** to apply the settings and return to [Section 5.3, “The Installation Summary window”](#).

5.7. CONFIGURING STORAGE DEVICES

You can install Red Hat Enterprise Linux on a large variety of storage devices. You can configure basic, locally accessible, storage devices in the **Installation Destination** window. Basic storage devices directly connected to the local system, such as hard disk drives and solid-state drives, are displayed in the **Local Standard Disks** section of the window. On IBM Z, this section contains activated Direct Access Storage Devices (DASDs).

**WARNING**

A known issue prevents DASDs configured as HyperPAV aliases from being automatically attached to the system after the installation is complete. These storage devices are available during the installation, but are not immediately accessible after you finish installing and reboot. To attach HyperPAV alias devices, add them manually to the **/etc/dasd.conf** configuration file of the system.

5.7.1. Storage device selection

The storage device selection window lists all storage devices that the installation program can access. Depending on your system and available hardware, some tabs might not be displayed. The devices are grouped under the following tabs:

Multipath Devices

Storage devices accessible through more than one path, such as through multiple SCSI controllers or Fiber Channel ports on the same system.

**IMPORTANT**

The installation program only detects multipath storage devices with serial numbers that are 16 or 32 characters long.

Other SAN Devices

Devices available on a Storage Area Network (SAN).

Firmware RAID

Storage devices attached to a firmware RAID controller.

NVDIMM Devices

Under specific circumstances, Red Hat Enterprise Linux 8 can boot and run from (NVDIMM) devices in sector mode on the Intel 64 and AMD64 architectures.

System z Devices

Storage devices, or Logical Units (LUNs), attached through the zSeries Linux FCP (Fiber Channel Protocol) driver.

5.7.2. Filtering storage devices

In the storage device selection window you can filter storage devices either by their World Wide Identifier (WWID) or by the port, target, or logical unit number (LUN).

Prerequisite

The **Installation Summary** window is open.

Procedure

1. From the **Installation Summary** window, click **Installation Destination**. The **Installation Destination** window opens, listing all available drives.

2. Under the **Specialized & Network Disks** section, click **Add a disk...** The storage devices selection window opens.
3. Click the **Search by** tab to search by port, target, LUN, or WWID.
Searching by WWID or LUN requires additional values in the corresponding input text fields.
4. Select the option that you require from the **Search** drop-down menu.
5. Click **Find** to start the search. Each device is presented on a separate row with a corresponding check box.
6. Select the check box to enable the device that you require during the installation process.
Later in the installation process you can choose to install Red Hat Enterprise Linux on any of the selected devices, and you can choose to mount any of the other selected devices as part of the installed system automatically.



NOTE

- Selected devices are not automatically erased by the installation process and selecting a device does not put the data stored on the device at risk.
- You can add devices to the system after installation by modifying the **/etc/fstab** file.

7. Click **Done** to return to the **Installation Destination** window.



IMPORTANT

Any storage devices that you do not select are hidden from the installation program entirely. To chain load the boot loader from a different boot loader, select all the devices present.

5.7.3. Using advanced storage options

To use an advanced storage device, you can configure an iSCSI (SCSI over TCP/IP) target or FCoE (Fibre Channel over Ethernet) SAN (Storage Area Network).

To use iSCSI storage devices for the installation, the installation program must be able to discover them as iSCSI targets and be able to create an iSCSI session to access them. Each of these steps might require a user name and password for Challenge Handshake Authentication Protocol (CHAP) authentication. Additionally, you can configure an iSCSI target to authenticate the iSCSI initiator on the system to which the target is attached (reverse CHAP), both for discovery and for the session. Used together, CHAP and reverse CHAP are called mutual CHAP or two-way CHAP. Mutual CHAP provides the greatest level of security for iSCSI connections, particularly if the user name and password are different for CHAP authentication and reverse CHAP authentication.



NOTE

Repeat the iSCSI discovery and iSCSI login steps to add all required iSCSI storage. You cannot change the name of the iSCSI initiator after you attempt discovery for the first time. To change the iSCSI initiator name, you must restart the installation.

5.7.3.1. Discovering and starting an iSCSI session

Complete the following steps to discover and start an iSCSI session.

Prerequisites

- The **Installation Summary** window is open.

Procedure

1. From the **Installation Summary** window, click **Installation Destination**. The **Installation Destination** window opens, listing all available drives.
2. Under the **Specialized & Network Disks** section, click **Add a disk...** The storage devices selection window opens.
3. Click **Add iSCSI target...** The **Add iSCSI Storage Target** window opens.



IMPORTANT

You cannot place the **/boot** partition on iSCSI targets that you have manually added using this method – an iSCSI target containing a **/boot** partition must be configured for use with iBFT. However, in instances where the installed system is expected to boot from iSCSI with iBFT configuration provided by a method other than firmware iBFT, for example using iPXE, you can remove the **/boot** partition restriction using the **inst.nonibftiscsiboot** installer boot option.

4. Enter the IP address of the iSCSI target in the **Target IP Address** field.
5. Type a name in the **iSCSI Initiator Name** field for the iSCSI initiator in iSCSI qualified name (IQN) format. A valid IQN entry contains the following information:
 - The string **iqn**. (note the period).
 - A date code that specifies the year and month in which your organization's Internet domain or subdomain name was registered, represented as four digits for the year, a dash, and two digits for the month, followed by a period. For example, represent September 2010 as **2010-09**.
 - Your organization's Internet domain or subdomain name, presented in reverse order with the top-level domain first. For example, represent the subdomain **storage.example.com** as **com.example.storage**.
 - A colon followed by a string that uniquely identifies this particular iSCSI initiator within your domain or subdomain. For example, **:diskarrays-sn-a8675309**.
 A complete IQN is as follows: **iqn.2010-09.storage.example.com:diskarrays-sn-a8675309**. The installation program prepopulates the **iSCSI Initiator Name** field with a name in this format to help you with the structure. For more information about IQNs, see 3.2.6. *iSCSI Names in RFC 3720 - Internet Small Computer Systems Interface (iSCSI)* available from tools.ietf.org and 1. *iSCSI Names and Addresses in RFC 3721 - Internet Small Computer Systems Interface (iSCSI) Naming and Discovery* available from tools.ietf.org.
6. Select the **Discovery Authentication Type** drop-down menu to specify the type of authentication to use for iSCSI discovery. The following options are available:
 - No credentials
 - CHAP pair

- CHAP pair and a reverse pair
7. a. If you selected **CHAP pair** as the authentication type, enter the user name and password for the iSCSI target in the **CHAP Username** and **CHAP Password** fields.
 - b. If you selected **CHAP pair and a reverse pair** as the authentication type, enter the user name and password for the iSCSI target in the **CHAP Username** and **CHAP Password** field, and the user name and password for the iSCSI initiator in the **Reverse CHAP Username** and **Reverse CHAP Password** fields.
 8. Optionally, select the **Bind targets to network interfaces** check box.
 9. Click **Start Discovery**.
- The installation program attempts to discover an iSCSI target based on the information provided. If discovery succeeds, the **Add iSCSI Storage Target** window displays a list of all iSCSI nodes discovered on the target.
10. Select the check boxes for the node that you want to use for installation.



NOTE

The **Node login authentication type** menu contains the same options as the **Discovery Authentication Type** menu. However, if you need credentials for discovery authentication, use the same credentials to log in to a discovered node.

11. Click the additional **Use the credentials from discovery** drop-down menu. When you provide the proper credentials, the **Log In** button becomes available.
12. Click **Log In** to initiate an iSCSI session.

5.7.3.2. Configuring FCoE parameters

Complete the following steps to configure FCoE parameters.

Prerequisite

The **Installation Summary** window is open.

Procedure

1. From the **Installation Summary** window, click **Installation Destination**. The **Installation Destination** window opens, listing all available drives.
2. Under the **Specialized & Network Disks** section, click **Add a disk...**. The storage devices selection window opens.
3. Click **dd FCoE SAN...**. A dialog box opens for you to configure network interfaces for discovering FCoE storage devices.
4. Select a network interface that is connected to an FCoE switch in the **NIC** drop-down menu.
5. Click **Add FCoE disk(s)** to scan the network for SAN devices.
6. Select the required check boxes:
 - **Use DCB**: *Data Center Bridging* (DCB) is a set of enhancements to the Ethernet protocols

designed to increase the efficiency of Ethernet connections in storage networks and clusters. Select the check box to enable or disable the installation program's awareness of DCB. Enable this option only for network interfaces that require a host-based DCBX client. For configurations on interfaces that use a hardware DCBX client, disable the check box.

- **Use auto vlan:** Auto VLAN is enabled by default and indicates whether VLAN discovery should be performed. If this check box is enabled, then the FIP (FCoE Initiation Protocol) VLAN discovery protocol runs on the Ethernet interface when the link configuration has been validated. If they are not already configured, network interfaces for any discovered FCoE VLANs are automatically created and FCoE instances are created on the VLAN interfaces.
7. Discovered FCoE devices are displayed under the **Other SAN Devices** tab in the **Installation Destination** window.

5.7.3.3. Configuring DASD storage devices

Complete the following steps to configure DASD storage devices.

Prerequisite

The **Installation Summary** window is open.

Procedure

1. From the **Installation Summary** window, click **Installation Destination**. The **Installation Destination** window opens, listing all available drives.
2. Under the **Specialized & Network Disks** section, click **Add a disk...** The storage devices selection window opens.
3. Click **Add DASD**. The **Add DASD Storage Target** dialog box opens and prompts you to specify a device number, such as **0.0.0204**, and attach additional DASDs that were not detected when the installation started.
4. Type the device number of the DASD that you want to attach in the **Device number** field.
5. Click **Start Discovery**.



NOTE

- If a DASD with the specified device number is found and if it is not already attached, the dialog box closes and the newly-discovered drives appear in the list of drives. You can then select the check boxes for the required devices and click **Done**. The new DASDs are available for selection (marked as **DASD device 0.0.xxxx**) in the **Local Standard Disks** section of the **Installation Destination** window.
- If you entered an invalid device number, or if the DASD with the specified device number is already attached to the system, an error message appears in the dialog box, explaining the error and prompting you to try again with a different device number.

5.7.3.4. Configuring FCP devices

FCP devices enable IBM Z to use SCSI devices rather than, or in addition to, Direct Access Storage Device (DASD) devices. FCP devices provide a switched fabric topology that enables IBM Z systems to use SCSI LUNs as disk devices in addition to traditional DASD devices.

Prerequisites

- The **Installation Summary** window is open.
- For an FCP-only installation, remove the **DASD=** option from the CMS configuration file or the **rd.dasd=** option from the parameter file to indicate that no DASD is present.

Procedure

1. From the **Installation Summary** window, click **Installation Destination**. The **Installation Destination** window opens, listing all available drives.
2. Under the **Specialized & Network Disks** section, click **Add a disk....** The storage devices selection window opens.
3. Click **Add ZFCP LUN**. The **Add zFCP Storage Target** dialog box opens allowing you to add a FCP (Fibre Channel Protocol) storage device.
IBM Z requires that you enter any FCP device manually so that the installation program can activate FCP LUNs. You can enter FCP devices either in the graphical installation, or as a unique parameter entry in the parameter or CMS configuration file. The values that you enter must be unique to each site that you configure.
4. Type the 4 digit hexadecimal device number in the **Device number** field.
5. Type the 16 digit hexadecimal World Wide Port Number (WWPN) in the **WWPN** field.
6. Type the 16 digit hexadecimal FCP LUN identifier in the **LUN** field.
7. Click **Start Discovery** to connect to the FCP device.

The newly-added devices are displayed in the **System z Devices** tab of the **Installation Destination** window.



NOTE

- Interactive creation of an FCP device is only possible in graphical mode. It is not possible to configure an FCP device interactively in text mode installation.
- Use only lower-case letters in hex values. If you enter an incorrect value and click **Start Discovery**, the installation program displays a warning. You can edit the configuration information and retry the discovery attempt.
- For more information about these values, consult the hardware documentation and check with your system administrator.

5.7.4. Installing to an NVDIMM device

Non-Volatile Dual In-line Memory Module (NVDIMM) devices combine the performance of RAM with disk-like data persistence when no power is supplied. Under specific circumstances, Red Hat Enterprise Linux 8 can boot and run from NVDIMM devices.

5.7.4.1. Criteria for using an NVDIMM device as an installation target

You can install Red Hat Enterprise Linux 8 to Non-Volatile Dual In-line Memory Module (NVDIMM) devices in sector mode on the Intel 64 and AMD64 architectures, supported by the **nd_pmem** driver.

Conditions for using an NVDIMM device as storage

To use an NVDIMM device as storage, the following conditions must be satisfied:

- The architecture of the system is Intel 64 or AMD64.
- The NVDIMM device is configured to sector mode. The installation program can reconfigure NVDIMM devices to this mode.
- The NVDIMM device must be supported by the **nd_pmem** driver.

Conditions for booting from an NVDIMM Device

Booting from an NVDIMM device is possible under the following conditions:

- All conditions for using the NVDIMM device as storage are satisfied.
- The system uses UEFI.
- The NVDIMM device must be supported by firmware available on the system, or by an UEFI driver. The UEFI driver may be loaded from an option ROM of the device itself.
- The NVDIMM device must be made available under a namespace.

Utilize the high performance of NVDIMM devices during booting, place the **/boot** and **/boot/efi** directories on the device. The Execute-in-place (XIP) feature of NVDIMM devices is not supported during booting and the kernel is loaded into conventional memory.

5.7.4.2. Configuring an NVDIMM device using the graphical installation mode

A Non-Volatile Dual In-line Memory Module (NVDIMM) device must be properly configured for use by Red Hat Enterprise Linux 8 using the graphical installation.



WARNING

Reconfiguration of a NVDIMM device process destroys any data stored on the device.

Prerequisites

- A NVDIMM device is present on the system and satisfies all the other conditions for usage as an installation target.
- The installation has booted and the **Installation Summary** window is open.

Procedure

1. From the **Installation Summary** window, click **Installation Destination**. The **Installation Destination** window opens, listing all available drives.
2. Under the **Specialized & Network Disks** section, click **Add a disk...** The storage devices selection window opens.
3. Click the **NVDIMM Devices** tab.
4. To reconfigure a device, select it from the list.
If a device is not listed, it is not in sector mode.
5. Click **Reconfigure NVDIMM...** A reconfiguration dialog opens.
6. Enter the sector size that you require and click **Start Reconfiguration**.
The supported sector sizes are 512 and 4096 bytes.
7. When reconfiguration completes click **OK**.
8. Select the device check box.
9. Click **Done** to return to the **Installation Destination** window.
The NVDIMM device that you reconfigured is displayed in the **Specialized & Network Disks** section.
10. Click **Done** to return to the **Installation Summary** window.

The NVDIMM device is now available for you to select as an installation target. Additionally, if the device meets the requirements for booting, you can set the device as a boot device.

5.8. CONFIGURING MANUAL PARTITIONING

You can use manual partitioning to configure your disk partitions and mount points and define the file system that Red Hat Enterprise Linux is installed on.



NOTE

Before installation, you should consider whether you want to use partitioned or unpartitioned disk devices. For more information, see the Knowledgebase article at <https://access.redhat.com/solutions/163853>.

An installation of Red Hat Enterprise Linux requires a minimum of one partition but Red Hat recommends using at least the following partitions or volumes: **PReP**, **/**, **/home**, **/boot**, and **swap**. You can also create additional partitions and volumes as you require.



NOTE

An installation of Red Hat Enterprise Linux on IBM Power Systems servers requires a **PReP** boot partition.



WARNING

To prevent data loss it is recommended that you back up your data before proceeding. If you are upgrading or creating a dual-boot system, you should back up any data you want to keep on your storage devices.

5.8.1. Starting manual partitioning

Prerequisites

- The **Installation Summary** screen is currently displayed.
- All disks are available to the installation program.

Procedure

1. Select disks for installation:
 - a. Click **Installation Destination** to open the **Installation Destination** window.
 - b. Select the disks that you require for installation by clicking the corresponding icon. A selected disk has a check-mark displayed on it.
 - c. Under **Storage Configuration**, select the **Custom** radio-button.
 - d. Optional: To enable storage encryption with LUKS, select the **Encrypt my data** check box.
 - e. Click **Done**.
2. If you selected to encrypt the storage, a dialog box for entering a disk encryption passphrase opens. Type in the LUKS passphrase:
 - a. Enter the passphrase in the two text fields. To switch keyboard layout, use the keyboard icon.



WARNING

In the dialog box for entering the passphrase, you cannot change the keyboard layout. Select the English keyboard layout to enter the passphrase in the installation program.

- b. Click **Save Passphrase**. The **Manual Partitioning** window opens.
3. Deleted Mount points are listed in the left-hand pane. The mount points are organized by detected operating system installations. As a result, some file systems may be displayed multiple times if a partition is shared among several installations.

- a. Select the mount points in the left pane; the options that can be customized are displayed in the right pane.



NOTE

- If your system contains existing file systems, ensure that enough space is available for the installation. To remove any partitions, select them in the list and click the **-** button.
The dialog has a check box that you can use to remove all other partitions used by the system to which the deleted partition belongs.
- If there are no existing partitions and you want to create the recommended set of partitions as a starting point, select your preferred partitioning scheme from the left pane (default for Red Hat Enterprise Linux is LVM) and click the **Click here to create them automatically** link.
A **/boot** partition, a **/** (root) volume, and a **swap** volume proportionate to the size of the available storage are created and listed in the left pane. These are the recommended file systems for a typical installation, but you can add additional file systems and mount points.

- b. Click **Done** to confirm any changes and return to the **Installation Summary** window.

5.8.2. Adding a mount point file system

Complete the following steps to add multiple mount point file systems.

Prerequisites

- Plan for your partitions:
 - To avoid problems with space allocation, first create small partitions with known fixed sizes, such as **/boot**, and then create the remaining partitions, letting the installation program allocate the remaining capacity to them.
 - If you want to install the system on multiple disks, or if your disks differ in size and a particular partition must be created on the first disk detected by BIOS, then create these partitions first.

Procedure

1. Click **+** to create a new mount point file system. The **Add a New Mount Point** dialog opens.
2. Select one of the preset paths from the **Mount Point** drop-down menu or type your own; for example, select **/** for the root partition or **/boot** for the boot partition.
3. Enter the size of the file system in to the **Desired Capacity** field; for example, **2GiB**.

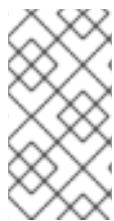
**WARNING**

If you do not specify a value in the Desired Capacity field, or if you specify a size bigger than available space, then all remaining free space is used.

4. Click **Add mount point** to create the partition and return to the **Manual Partitioning** window.

5.8.3. Configuring a mount point file system

This procedure describes how to set the partitioning scheme for each mount point that was created manually. The available options are **Standard Partition**, **LVM**, and **LVM Thin Provisioning**.

**NOTE**

- Btrfs support has been removed in Red Hat Enterprise Linux 8.
- The **/boot** partition is always located on a standard partition, regardless of the value selected.

Procedure

1. To change the devices that a single non-LVM mount point should be located on, select the required mount point from the left-hand pane.
2. Under the **Device(s)** heading, click **Modify...**. The **Configure Mount Point** dialog opens.
3. Select one or more devices and click **Select** to confirm your selection and return to the **Manual Partitioning** window.
4. Click **Update Settings** to apply the changes.

**NOTE**

Click the **Rescan** button (circular arrow button) to refresh all local disks and partitions; this is only required after performing advanced partition configuration outside the installation program. Clicking the **Rescan Disks** button resets all configuration changes made in the installation program.

5. In the lower left-hand side of the **Manual Partitioning** window, click the **storage device selected** link to open the **Selected Disks** dialog and review disk information.

5.8.4. Customizing a partition or volume

You can customize a partition or volume if you want to set specific settings.



IMPORTANT

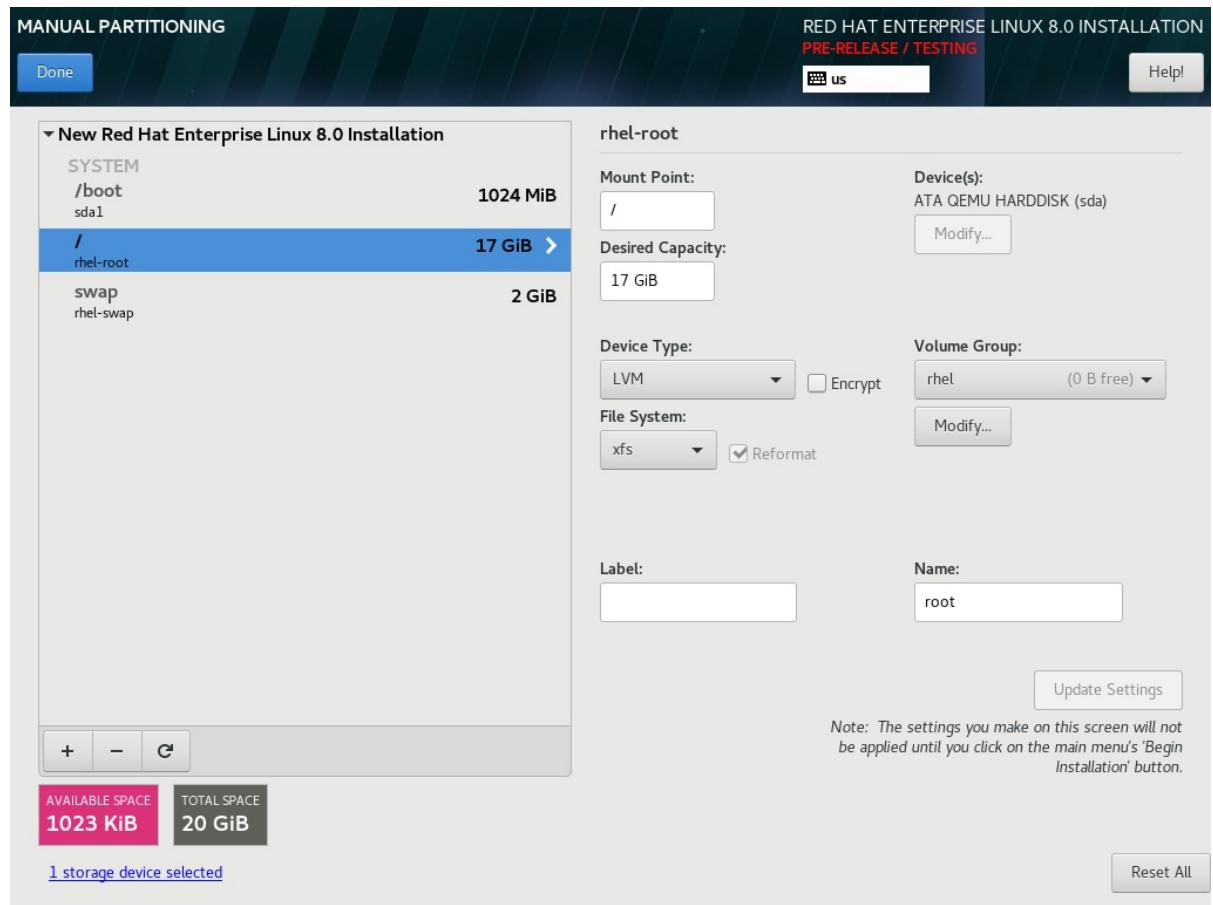
If **/usr** or **/var** is partitioned separately from the rest of the root volume, the boot process becomes much more complex as these directories contain critical components. In some situations, such as when these directories are placed on an iSCSI drive or an FCoE location, the system is unable to boot, or hangs with a **Device is busy** error when powering off or rebooting.

This limitation only applies to **/usr** or **/var**, not to directories below them. For example, a separate partition for **/var/www** works successfully.

Procedure

1. From the left pane, select the mount point.

Figure 5.1. Customizing Partitions



2. From the right-hand pane, you can customize the following options:

- a. Enter the file system mount point into the **Mount Point** field. For example, if a file system is the root file system, enter **/**; enter **/boot** for the **/boot** file system, and so on. For a swap file system, do not set the mount point as setting the file system type to **swap** is sufficient.
- b. Enter the size of the file system in the **Desired Capacity** field. You can use common size units such as KiB or GiB. The default is MiB if you do not set any other unit.
- c. Select the device type that you require from the drop-down **Device Type** menu: **Standard Partition**, **LVM**, or **LVM Thin Provisioning**.

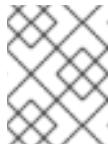
**WARNING**

The installation program does not support overprovisioned LVM thin pools.

**NOTE**

RAID is available only if two or more disks are selected for partitioning. If you choose **RAID**, you can also set the **RAID Level**. Similarly, if you select **LVM**, you can specify the **Volume Group**.

- d. Select the **Encrypt** check box to encrypt the partition or volume. You must set a password later in the installation program. The **LUKS Version** drop-down menu is displayed.
- e. Select the LUKS version that you require from the drop-down menu.
- f. Select the appropriate file system type for this partition or volume from the **File system** drop-down menu.

**NOTE**

Support for **VFAT** file system is not available for Linux system partitions. For example, **/**, **/var**, **/usr**, and so on.

- g. Select the **Reformat** check box to format an existing partition, or deselect the **Reformat** check box to retain your data. The newly-created partitions and volumes must be reformatted, and the check box cannot be deselected.
- h. Type a label for the partition in the **Label** field. Use labels to easily recognize and address individual partitions.
- i. Type a name in the **Name** field.

**NOTE**

Note that standard partitions are named automatically when they are created and you cannot edit the names of standard partitions. For example, you cannot edit the **/boot** name **sda1**.

3. Click **Update Settings** to apply your changes and if required, select another partition to customize. Changes are not applied until you click **Begin Installation** from the **Installation Summary** window.

**NOTE**

Click **Reset All** to discard your partition changes.

4. Click **Done** when you have created and customized all file systems and mount points. If you choose to encrypt a file system, you are prompted to create a passphrase.

A **Summary of Changes** dialog box opens, displaying a summary of all storage actions for the installation program.

5. Click **Accept Changes** to apply the changes and return to the **Installation Summary** window.

5.8.5. Preserving the **/home** directory

In a RHEL 8 graphical installation, you can preserve the **/home** directory that was used on your RHEL 7 system.



WARNING

Preserving **/home** is only possible if the **/home** directory is located on a separate **/home** partition on your RHEL 7 system.

Preserving the **/home** directory that includes various configuration settings, makes it possible that the GNOME Shell environment on the new RHEL 8 system is set in the same way as it was on your RHEL 7 system. Note that this applies only for users on RHEL 8 with the same user name and ID as on the previous RHEL 7 system.

Complete this procedure to preserve the **/home** directory from your RHEL 7 system.

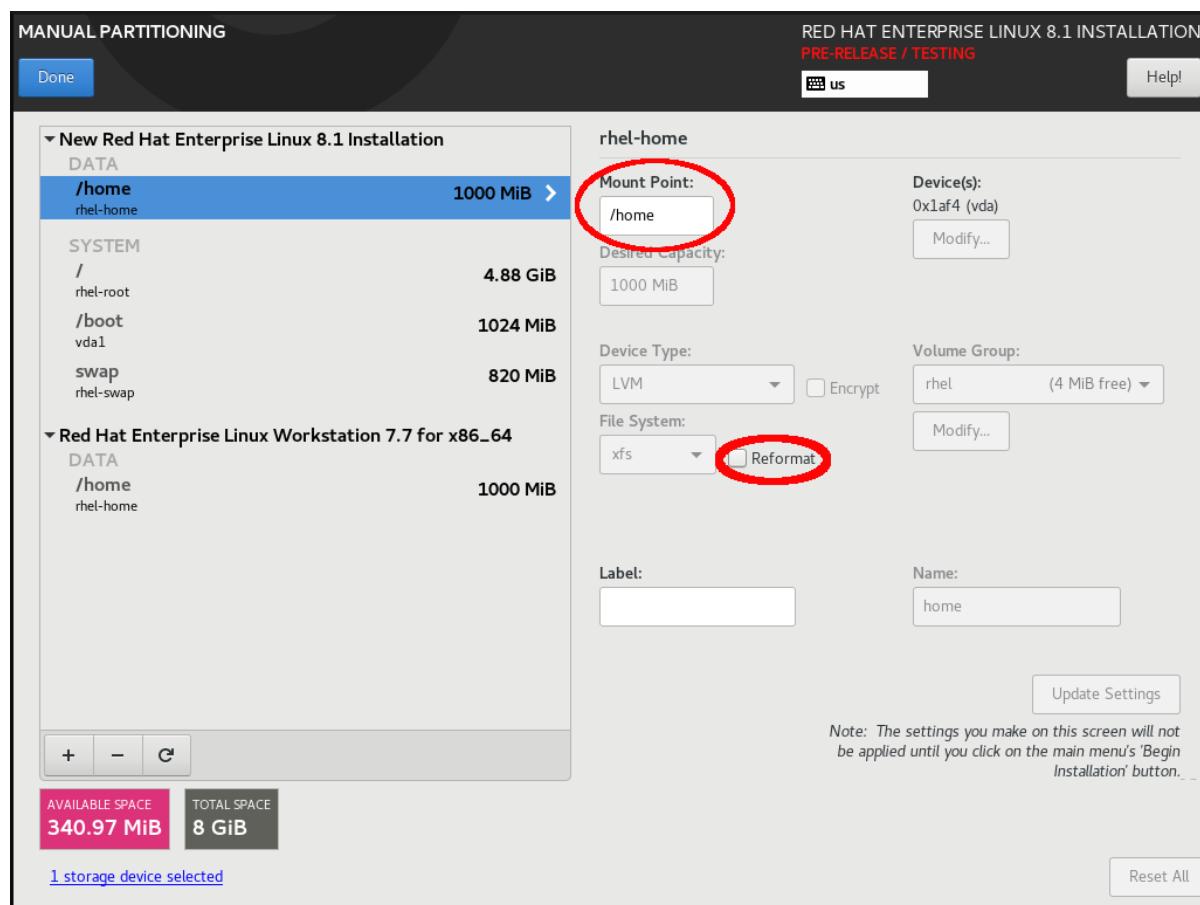
Prerequisites

- RHEL 7 system is installed on your computer.
- The **/home** directory is located on a separate **/home** partition on your RHEL 7 system.
- The RHEL 8 **Installation Summary** window is currently displayed.

Procedure

1. Click **Installation Destination** to open the **Installation Destination** window.
2. Under **Storage Configuration**, select the **Custom** radio button. Click **Done**.
3. Click **Done**, the **Manual Partitioning** window opens.
4. Choose the **/home** partition, fill in **/home** under **Mount Point:** and clear the **Reformat** check box.

Figure 5.2. Ensuring that /home is not reformatted



5. Optional: You can also customize various aspects of the **/home** partition required for your RHEL 8 system as described in [Section 5.8.4, "Customizing a partition or volume"](#). However, to preserve **/home** from your RHEL 7 system, it is necessary to clear the **Reformat** check box.
6. After you customized all partitions according to your requirements, click **Done**. The **Summary of changes** dialog box opens.
7. Verify that the **Summary of changes** dialog box does not show any change for **/home**. This means that the **/home** partition is preserved.
8. Click **Accept Changes** to apply the changes, and return to the **Installation Summary** window.

5.8.6. Creating software RAID

Follow the steps in this procedure to create a Redundant Arrays of Independent Disks (RAID) device. RAID devices are constructed from multiple storage devices that are arranged to provide increased performance and, in some configurations, greater fault tolerance.

A RAID device is created in one step and disks are added or removed as necessary. You can configure one RAID partition for each physical disk in your system, so the number of disks available to the installation program determines the levels of RAID device available. For example, if your system has two hard drives, you cannot create a RAID 10 device, as it requires a minimum of three separate disks.



NOTE

On IBM Z, the storage subsystem uses RAID transparently. You do not have to configure software RAID manually.

Prerequisites

- You have selected two or more disks for installation before RAID configuration options are visible. At least two disks are required to create a RAID device.
- You have created a mount point. By configuring a mount point, you configure the RAID device.
- You have selected the **Custom** radio button on the **Installation Destination** window.

Procedure

1. From the left pane of the **Manual Partitioning** window, select the required partition.
2. Under the **Device(s)** section, click **Modify**. The **Configure Mount Point** dialog box opens.
3. Select the disks that you want to include in the RAID device and click **Select**.
4. Click the **Device Type** drop-down menu and select **RAID**.
5. Click the **File System** drop-down menu and select your preferred file system type.
6. Click the **RAID Level** drop-down menu and select your preferred level of RAID.
7. Click **Update Settings** to save your changes.
8. Click **Done** to apply the settings and return to the **Installation Summary** window.

A message is displayed at the bottom of the window if the specified RAID level requires more disks.

5.8.7. Creating an LVM logical volume

Logical Volume Management (LVM) presents a simple logical view of underlying physical storage space, such as hard drives or LUNs. Partitions on physical storage are represented as physical volumes that you can group together into volume groups. You can divide each volume group into multiple logical volumes, each of which is analogous to a standard disk partition. Therefore, LVM logical volumes function as partitions that can span multiple physical disks.



NOTE

LVM configuration is available only in the graphical installation program.



IMPORTANT

During text-mode installation, LVM configuration is not available. To create an LVM configuration, press **Ctrl+Alt+F2** to use a different virtual console, and run the **lvm** command. To return to the text-mode installation, press **Ctrl+Alt+F1**.

Procedure

1. From the left-hand pane of the **Manual Partitioning** window, select the mount point.
2. Click the **Device Type** drop-down menu and select **LVM**. The **Volume Group** drop-down menu is displayed with the newly-created volume group name.



NOTE

You cannot specify the size of the volume group's physical extents in the configuration dialog. The size is always set to the default value of 4 MiB. If you want to create a volume group with different physical extents, you must create it manually by switching to an interactive shell and using the **vgcreate** command, or use a Kickstart file with the **volgroup --pesize=size** command. See the [Performing an advanced RHEL installation](#) document for more information about Kickstart.

Additional resources

- For more information about LVM, see the [Configuring and managing logical volumes](#) document.

5.8.8. Configuring an LVM logical volume

Follow the steps in this procedure to configure a newly-created LVM logical volume.



WARNING

Placing the **/boot** partition on an LVM volume is not supported.

Procedure

- From the left-hand pane of the **Manual Partitioning** window, select the mount point.
- Click the **Device Type** drop-down menu and select **LVM**. The **Volume Group** drop-down menu is displayed with the newly-created volume group name.
- Click **Modify** to configure the newly-created volume group.
The **Configure Volume Group** dialog box opens.



NOTE

You cannot specify the size of the volume group's physical extents in the configuration dialog. The size is always set to the default value of 4 MiB. If you want to create a volume group with different physical extents, you must create it manually by switching to an interactive shell and using the **vgcreate** command, or use a Kickstart file with the **volgroup --pesize=size** command. See the [Performing an advanced RHEL installation](#) document for more information about Kickstart.

- From the **RAID Level** drop-down menu, select the RAID level that you require.
The available RAID levels are the same as with actual RAID devices.
- Select the **Encrypt** check box to mark the volume group for encryption.
- From the **Size policy** drop-down menu, select the size policy for the volume group.
The available policy options are:

- **Automatic:** The size of the volume group is set automatically so that it is large enough to contain the configured logical volumes. This is optimal if you do not need free space within the volume group.
- **As large as possible** The volume group is created with maximum size, regardless of the size of the configured logical volumes it contains. This is optimal if you plan to keep most of your data on LVM and later need to increase the size of some existing logical volumes, or if you need to create additional logical volumes within this group.
- **Fixed:** You can set an exact size of the volume group. Any configured logical volumes must then fit within this fixed size. This is useful if you know exactly how large you need the volume group to be.

7. Click **Save** to apply the settings and return to the **Manual Partitioning** window.

8. Click **Update Settings** to save your changes

9. Click **Done** to return to the **Installation Summary** window.

5.9. STARTING THE INSTALLATION PROGRAM

Before you start the installation program, you must configure your root password and user settings.

5.9.1. Beginning installation

When the installation process has started, it is not possible to return to the **Installation Summary** window and change any settings. To change settings, you must wait for the installation process to finish, reboot your system, log in, and change your settings on the installed system.

Prerequisites

- You have completed all configuration steps in [Section 5.3, “The Installation Summary window”](#).
- The **Installation Summary** window is open.

Procedure

1. From the **Installation Summary** window, click **Begin Installation**. The **Configuration** window opens and the installation process starts.

Two user setting options, **Root Password** (mandatory) and **User Creation** (optional) are available.



IMPORTANT

Before you finish the installation and reboot, either remove the media (CD, DVD, or a USB drive) used to start the installation, or verify that your system tries to boot from the hard drive before attempting removable media. Otherwise, your system starts the installation program again, instead of the installed system.

5.9.2. Configuring a root password

You must configure a **root** password to finish the installation process and to log in to the administrator (also known as superuser or root) account that is used for system administration tasks. These tasks include installing and updating software packages and changing system-wide configuration such as

network and firewall settings, storage options, and adding or modifying users, groups and file permissions.



IMPORTANT

- Use one or both of the following ways to gain root privileges to the installed system:
 - Use a root account
 - Create a user account with administrative privileges (member of the wheel group). The **root** account is always created during the installation. Switch to the administrator account only when you need to perform a task that requires administrator access.



WARNING

The **root** account has complete control over the system. If unauthorized personnel gain access to the account, they can access or delete users' personal files.

Procedure

1. From the **Configuration** window, click **Root Password**. The **Root Password** window opens.
2. Type your password in the **Root Password** field.
The requirements and recommendations for creating a strong root password are:
 - *Must* be at least eight characters long
 - May contain numbers, letters (upper and lower case) and symbols
 - Is case-sensitive
3. Type the same password in the **Confirm** field.
4. Click **Done** to confirm your root password and return to [Section 5.9.1, "Beginning installation"](#).



NOTE

If you proceeded with a weak password, you must click **Done** twice.

5.9.3. Creating a user account

It is recommended that you create a user account to finish the installation. If you do not create a user account, you must log in to the system as **root** directly, which is **not** recommended.

Procedure

1. From the **Configuration** window, click **User Creation**. The **Create User** window opens.

2. Type the user account name in to the **Full name** field, for example: John Smith.
3. Type the username in to the **User name** field, for example: jsmith.



NOTE

The **User name** is used to log in from a command line; if you install a graphical environment, then your graphical login manager uses the **Full name**.

4. Select the **Make this user administrator** check box if the user requires administrative rights (the installation program adds the user to the **wheel** group).



IMPORTANT

An administrator user can use the **sudo** command to perform tasks that are only available to **root** using the user password, instead of the **root** password. This may be more convenient, but it can also cause a security risk.

5. Select the **Require a password to use this account** check box.



WARNING

If you give administrator privileges to a user, verify that the account is password protected. Never give a user administrator privileges without assigning a password to the account.

6. Type a password into the **Password** field.
7. Type the same password into the **Confirm password** field.
8. **Save Changes** to apply the changes and return to the **Configuration** window.
9. When the installation process is complete, click **Reboot** to reboot and log in to your Red Hat Enterprise Linux 8 system.

5.9.3.1. Editing advanced user settings

Follow the steps in this procedure to edit the default settings for the user account in the **Advanced User Configuration** dialog box.

Procedure

1. Edit the details in the **Home directory** field, if required. The field is populated by default with **/home/username** .
2. In the **User and Groups IDs** section you can:
 - a. Select the **Specify a user ID manually** check box and use the **+** or **-** to enter the required value.

**NOTE**

The default value is 1000. User IDs (UIDs) 0–999 are reserved by the system so they cannot be assigned to a user.

- b. Select the **Specify a group ID manually** check box and use the + or - to enter the required value.

**NOTE**

The default group name is the same as the user name, and the default Group ID (GID) is 1000. GIDs 0–999 are reserved by the system so they can not be assigned to a user group.

3. Specify additional groups as a comma-separated list in the **Group Membership** field. Groups that do not already exist are created; you can specify custom GIDs for additional groups in parentheses. If you do not specify a custom GID for a new group, the new group receives a GID automatically.

**NOTE**

The user account created always has one default group membership (the user's default group with an ID set in the **Specify a group ID manually** field).

4. Click **Save Changes** to apply the updates and return to the **Configuration** window.

5.9.4. Graphical installation complete

Remove any installation media if it is not ejected automatically upon reboot.

Red Hat Enterprise Linux 8 starts after your system's normal power-up sequence is complete. If your system was installed on a workstation with the X Window System, applications to configure your system are launched. These applications guide you through initial configuration and you can set your system time and date, register your system with Red Hat, and more. If the X Window System is not installed, a **login:** prompt is displayed.

To learn how to complete initial setup, register, and secure your system, see the [Completing post-installation tasks](#) section of the *Performing a standard RHEL installation* document.

CHAPTER 6. COMPLETING POST-INSTALLATION TASKS

This section describes how to complete the following post-installation tasks:

- Completing initial setup
- Registering your system



NOTE

Depending on your requirements, there are several methods to register your system. Most of these methods are completed as part of post-installation tasks. However, the Red Hat Content Delivery Network (CDN) registers your system and attaches RHEL subscriptions **before** the installation process starts. See [Section 2.3.2, “Registering and installing RHEL from the CDN”](#) for more information.

- Securing your system

6.1. COMPLETING INITIAL SETUP

This section contains information about how to complete initial setup on a Red Hat Enterprise Linux 8 system.



IMPORTANT

- If you selected the **Server with GUI** base environment during installation, the **Initial Setup** window opens the first time you reboot your system after the installation process is complete.
- If you registered and installed RHEL from the CDN, the **Subscription Manager** option displays a note that all installed products are covered by valid entitlements.

The information displayed in the **Initial Setup** window might vary depending on what was configured during installation. At a minimum, the **Licensing** and **Subscription Manager** options are displayed.

Prerequisites

- You have completed the graphical installation according to the recommended workflow described on [Section 5.1, “Graphical installation workflow”](#).
- You have an active, non-evaluation Red Hat Enterprise Linux subscription.

Procedure

1. From the **Initial Setup** window, select **Licensing Information**.
The **License Agreement** window opens and displays the licensing terms for Red Hat Enterprise Linux.
2. Review the license agreement and select the **I accept the license agreement** checkbox.

**NOTE**

You must accept the license agreement. Exiting **Initial Setup** without completing this step causes a system restart. When the restart process is complete, you are prompted to accept the license agreement again.

3. Click **Done** to apply the settings and return to the **Initial Setup** window.

**NOTE**

If you did not configure network settings, you cannot register your system immediately. In this case, click **Finish Configuration**. Red Hat Enterprise Linux 8 starts and you can login, activate access to the network, and register your system. See [Section 6.3, “Registering your system using the Subscription Manager User Interface”](#) for more information. If you configured network settings, as described in [Section 5.5.3, “Configuring network and host name options”](#), you can register your system immediately, as shown in the following steps:

4. From the **Initial Setup** window, select **Subscription Manager**.

**IMPORTANT**

If you registered and installed RHEL from the CDN, the Subscription Manager option displays a note that all installed products are covered by valid entitlements.

5. The **Subscription Manager** graphical interface opens and displays the option you are going to register, which is: `subscription.rhsm.redhat.com`.
6. Click **Next**.
7. Enter your **Login** and **Password** details and click **Register**.
8. Confirm the Subscription details and click **Attach**. You must receive the following confirmation message: **Registration with Red Hat Subscription Management is Done!**
9. Click **Done**. The **Initial Setup** window opens.
10. Click **Finish Configuration**. The login window opens.
11. Configure your system. See the [Configuring basic system settings](#) document for more information.

Additional resources

Depending on your requirements, there are five methods to register your system:

- Using the Red Hat Content Delivery Network (CDN) to register your system, attach RHEL subscriptions, and install Red Hat Enterprise Linux. See [Section 2.3.2, “Registering and installing RHEL from the CDN”](#) for more information.
- During installation using **Initial Setup**.
- After installation using the command line. See [Section 6.2, “Registering your system using the command line”](#) for more information.

- After installation using the Subscription Manager user interface. See [Section 6.3, “Registering your system using the Subscription Manager User Interface”](#) for more information.
- After installation using Registration Assistant. Registration Assistant is designed to help you choose the most suitable registration option for your Red Hat Enterprise Linux environment. See <https://access.redhat.com/labs/registrationassistant/> for more information.

6.2. REGISTERING YOUR SYSTEM USING THE COMMAND LINE

This section contains information about how to register your Red Hat Enterprise Linux 8 system using the command line.



NOTE

When auto-attaching a system, the subscription service checks if the system is physical or virtual, as well as how many sockets are on the system. A physical system usually consumes two entitlements, a virtual system usually consumes one. One entitlement is consumed per two sockets on a system.

Prerequisites

- You have an active, non-evaluation Red Hat Enterprise Linux subscription.
- Your Red Hat subscription status is verified.
- You have not previously received a Red Hat Enterprise Linux 8 subscription.
- You have activated your subscription before attempting to download entitlements from the Customer Portal. You need an entitlement for each instance that you plan to use. Red Hat Customer Service is available if you need help activating your subscription.
- You have successfully installed Red Hat Enterprise Linux 8 and logged into the system.

Procedure

1. Open a terminal window and register a subscription using your Red Hat Customer Portal username and password:

```
# subscription-manager register --username [username] --password [password]
```

2. When the subscription is successfully registered, an output similar to the following is displayed:

```
# The system has been registered with ID: 123456abcdef
# The registered system name is: localhost.localdomain
```

3. Set the role for the system, for example:

```
# subscription-manager role --set="Red Hat Enterprise Linux Server"
```



NOTE

Available roles depend on the subscriptions that have been purchased by the organization and the architecture of the RHEL 8 system. You can set one of the following roles: **Red Hat Enterprise Linux Server**, **Red Hat Enterprise Linux Workstation**, or **Red Hat Enterprise Linux Compute Node**.

4. Set the service level for the system, for example:

```
# subscription-manager service-level --set="Premium"
```

5. Set the usage for the system, for example:

```
# subscription-manager usage --set="Production"
```

6. Attach the system to an entitlement that matches the host system architecture:

```
# subscription-manager attach
```

7. When the subscription is successfully attached, an output similar to the following is displayed:

```
Installed Product Current Status:  
Product Name: Red Hat Enterprise Linux for x86_64  
Status: Subscribed
```



NOTE

You can also register Red Hat Enterprise Linux 8 by logging in to the system as a **root** user and using the Subscription Manager graphical user interface.

6.3. REGISTERING YOUR SYSTEM USING THE SUBSCRIPTION MANAGER USER INTERFACE

This section contains information about how to register your Red Hat Enterprise Linux 8 system using the Subscription Manager User Interface to receive updates and access package repositories.

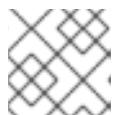
Prerequisites

- You have completed the graphical installation as per the recommended workflow described on [Section 5.1, “Graphical installation workflow”](#).
- You have an active, non-evaluation Red Hat Enterprise Linux subscription.
- Your Red Hat subscription status is verified.

Procedure

1. Log in to your system.
2. From the top left-hand side of the window, click **Activities**.
3. From the menu options, click the **Show Applications** icon.

4. Click the **Red Hat Subscription Manager** icon, or enter **Red Hat Subscription Manager** in the search.
5. Enter your administrator password in the **Authentication Required** dialog box.

**NOTE**

Authentication is required to perform privileged tasks on the system.

6. The **Subscriptions** window opens, displaying the current status of Subscriptions, System Purpose, and installed products. Unregistered products display a red X.
7. Click the **Register** button.
8. The **Register System** dialog box opens. Enter your **Customer Portal** credentials and click the **Register** button.

The **Register** button in the **Subscriptions** window changes to **Unregister** and installed products display a green X. You can troubleshoot an unsuccessful registration using the **subscription-manager status** command.

Additional resources

- For more information about Subscription Manager, see the [Using and Configuring Red Hat Subscription Manager](#) document.

6.4. REGISTRATION ASSISTANT

Registration Assistant is designed to help you choose the most suitable registration option for your Red Hat Enterprise Linux environment. See <https://access.redhat.com/labs/registrationassistant/> for more information.

6.5. CONFIGURING SYSTEM PURPOSE USING THE SYSPURPOSE COMMAND-LINE TOOL

System Purpose is an optional but recommended feature of the Red Hat Enterprise Linux installation. You use System Purpose to record the intended use of a Red Hat Enterprise Linux 8 system, and ensure that the entitlement server auto-attaches the most appropriate subscription to your system. The **syspurpose** command-line tool is part of the **python3_syspurpose.rpm** package. If System Purpose was not configured during the installation process, you can use the **syspurpose** command-line tool after installation to set the required attributes.

Prerequisites

- You installed and registered your Red Hat Enterprise Linux 8 system, but System Purpose is not configured.
- You are logged in as a **root** user.
- The **python3_syspurpose.rpm** package is available on your system.



NOTE

If your system is registered but has subscriptions that do not satisfy the required purpose, you can run the **subscription-manager remove --all** command to remove attached subscriptions. You can then use the **syspurpose** command-line tool to set the required purpose attributes, and run **subscription-manager attach --auto** to entitle the system with the updated attributes.

Procedure

Complete the steps in this procedure to configure System Purpose after installation using the **syspurpose** command-line tool. The selected values are used by the entitlement server to attach the most suitable subscription to your system.

1. From a terminal window, run the following command to set the intended role of the system:

```
# syspurpose set-role "VALUE"
```

Replace **VALUE** with the role that you want to assign:

- **Red Hat Enterprise Linux Server**
- **Red Hat Enterprise Linux Workstation**
- **Red Hat Enterprise Linux Compute Node**

For example:

```
# syspurpose set-role "Red Hat Enterprise Linux Server"
```

- a. Optional: Run the following command to unset the role:

```
# syspurpose unset-role
```

2. Run the following command to set the intended Service Level Agreement (SLA) of the system:

```
# syspurpose set-sla "VALUE"
```

Replace **VALUE** with the SLA that you want to assign:

- **Premium**
- **Standard**
- **Self-Support**

For example:

```
# syspurpose set-sla "Standard"
```

- a. Optional: Run the following command to unset the SLA:

```
# syspurpose unset-sla
```

3. Run the following command to set the intended usage of the system:

```
# syspurpose set-usage "VALUE"
```

Replace **VALUE** with the usage that you want to assign:

- **Production**
- **Disaster Recovery**
- **Development/Test**

For example:

```
# syspurpose set-usage "Production"
```

- a. Optional: Run the following command to unset the usage:

```
# syspurpose unset-usage
```

4. Run the following command to show the current system purpose properties:

```
# syspurpose show
```

- a. Optional: Run the following command to access the **syspurpose** man page:

```
# man syspurpose
```

6.6. SECURING YOUR SYSTEM

Complete the following security-related steps immediately after you install Red Hat Enterprise Linux.

Prerequisites

- You have completed the graphical installation.

Procedure

1. To update your system, run the following command as root:

```
# yum update
```

2. Even though the firewall service, **firewalld**, is automatically enabled with the installation of Red Hat Enterprise Linux, there are scenarios where it might be explicitly disabled, for example in a Kickstart configuration. In that scenario, it is recommended that you re-enable the firewall.

To start **firewalld**, run the following commands as root:

```
# systemctl start firewalld
# systemctl enable firewalld
```

3. To enhance security, disable services that you do not need. For example, if your system has no printers installed, disable the cups service using the following command:

```
# systemctl mask cups
```

To review active services, run the following command:

```
$ systemctl list-units | grep service
```

6.7. DEPLOYING SYSTEMS THAT ARE COMPLIANT WITH A SECURITY PROFILE RIGHT AFTER AN INSTALLATION

Administrators can use the OpenSCAP suite to deploy RHEL systems that are compliant with a security profile, such as OSPP or PCI-DSS, right after the installation process. Administrators that use this deployment method can apply specific rules, for example, a rule for password strength, that cannot be applied later using remediation scripts.

6.7.1. Deploying OSPP-compliant RHEL systems using the graphical installation

Use this procedure to deploy a RHEL system that is aligned with Protection Profile for General Purpose Operating System (OSPP).

Prerequisites

- You have booted into the graphical installation program.
- You have accessed the **Installation Summary** window.

Procedure

1. From the **Installation Summary** window, click **Software Selection**. The **Software Selection** window opens.
2. From the **Base Environment** pane, select the **Server** environment. You can select only one base environment.



WARNING

Server with GUI is the default base environment. GNOME packages installed by the **Server with GUI** option require the **nfs-utils** package and this package is not OSPP-compliant. If you do not change the default base environment to **Server**, the installation process stops after you select OSPP.

3. Click **Done** to apply the setting and return to the **Installation Summary** window.
4. Click **Security Policy**. The **Security Policy** window opens.
5. To enable security policies on the system, toggle the **Apply security policy** switch to **ON**.
6. Select **Protection Profile for General Purpose Operating Systems** from the profile pane.

7. Click **Select Profile** to confirm the selection.
8. Confirm the changes in the **Changes that were done or need to be done** pane that is displayed at the bottom of the window. Complete any remaining manual changes.
9. Because OSPP has strict partitioning requirements that must be met, create separate partitions for **/boot**, **/home**, **/var**, **/var/log**, **/var/tmp**, and **/var/log/audit**.
10. Complete the graphical installation process.



NOTE

The graphical installation program automatically creates a corresponding Kickstart file after a successful installation. You can use the **/root/anaconda-ks.cfg** file to automatically install OSPP-compliant systems.

Verification steps

1. The report of the hardening process is in the **/root/openscap_data/eval_remediate_report.html** file. Because **oscap** creates the report in a **chroot** environment, it can contain also false positives, for example, all service-related rules are shown as errors.
2. To check the current status of the system properly, scan it after it restarts once the installation is complete:

```
# oscap xccdf eval --profile ospp --report eval_postinstall_report.html
/usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

Additional resources

- For more details on partitioning, see [Configuring manual partitioning](#).

6.7.2. Deploying OSPP-compliant RHEL systems using Kickstart

Use this procedure to deploy RHEL systems that are aligned with Protection Profile for General Purpose Operating System (OSPP).

Prerequisites

- The **scap-security-guide** package is installed on your RHEL 8 system.

Procedure

1. Open the **/usr/share/scap-security-guide/kickstarts/ssg-rhel8-ospp-ks.cfg** Kickstart file in an editor of your choice.
2. Update the partitioning scheme to fit your configuration requirements. For OSPP compliance, the separate partitions for **/boot**, **/home**, **/var**, **/var/log**, **/var/tmp**, and **/var/log/audit** must be preserved, and you can only change the size of the partitions.



WARNING

Because the **OSCAP Anaconda Addon** plugin does not support text-only installation, do not use the **text** option in your Kickstart file. For more information, see [RHBZ#1674001](#).

3. Start a Kickstart installation as described in [Performing an automated installation using Kickstart](#).



IMPORTANT

Passwords in the hash form cannot be checked for OSPP requirements.

Verification steps

1. The report of the hardening process is in the `/root/openscap_data/eval_remediate_report.html` file. Because `oscap` creates the report in a `chroot` environment, it can contain also false positives, for example, all service-related rules are shown as errors.
2. To check the current status of the system properly, scan it after it restarts once the installation is complete:

```
# oscap xccdf eval --profile ospp --report eval_postinstall_report.html
/usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

Additional resources

- For more details, see the [OSCAP Anaconda Addon](#) project page.

APPENDIX A. TROUBLESHOOTING

The following sections cover various troubleshooting information that might be helpful when diagnosing issues during different stages of the installation process.

A.1. TROUBLESHOOTING AT THE START OF THE INSTALLATION PROCESS

The troubleshooting information in the following sections might be helpful when diagnosing issues at the start of the installation process. The following sections are for all supported architectures. However, if an issue is for a particular architecture, it is specified at the start of the section.

A.1.1. Dracut

Dracut is a tool that manages the **initramfs** image during the Linux operating system boot process. The **dracut** emergency shell is an interactive mode that can be initiated while the **initramfs** image is loaded. You can run basic troubleshooting commands from the **dracut** emergency shell. For more information, see the **Troubleshooting** section of the **dracut** man page.

A.1.2. Using installation log files

For debugging purposes, the installation program logs installation actions in files that are located in the **/tmp** directory. These log files are listed in the following table.

Table A.1. Log files generated during the installation

Log file	Contents
/tmp/anaconda.log	General messages.
/tmp/program.log	All external programs run during the installation.
/tmp/storage.log	Extensive storage module information.
/tmp/packaging.log	yum and rpm package installation messages.
/tmp/dbus.log	Information about the dbus session that is used for installation program modules.
/tmp/ifcfg.log	Information about networking scripts.
/tmp/sensitive-info.log	Configuration information that is not part of other logs and not copied to the installed system.
/tmp/syslog	Hardware-related system messages.

If the installation fails, the messages are consolidated into **/tmp/anaconda-tb-identifier**, where identifier is a random string. After a successful installation, these files are copied to the installed system under the directory **/var/log/anaconda/**. However, if the installation is unsuccessful, or if the **inst.nosave=all** or **inst.nosave=logs** options are used when booting the installation system, these logs only exist in the

installation program's RAM disk. This means that the logs are not saved permanently and are lost when the system is powered down. To store them permanently, copy the files to another system on the network or copy them to a mounted storage device such as a USB flash drive.

A.1.2.1. Creating pre-installation log files

Use this procedure to set the **inst.debug** option to create log files before the installation process starts. These log files contain, for example, the current storage configuration.

Prerequisites

- The Red Hat Enterprise Linux boot menu is displayed.

Procedure

1. Select the **Install Red Hat Enterprise Linux** option from the boot menu.
2. Press the **Tab** key on BIOS-based systems or the **e** key on UEFI-based systems to edit the selected boot options.
3. Append **inst.debug** to the options. For example:

```
vmlinuz ... inst.debug
```
4. Press the **Enter** key on your keyboard. The system stores the pre-installation log files in the **/tmp/pre-anaconda-logs/** directory before the installation program starts.
5. To access the log files, switch to the console.
6. Change to the **/tmp/pre-anaconda-logs/** directory:

```
# cd /tmp/pre-anaconda-logs/
```

A.1.2.2. Transferring installation log files to a USB drive

Use this procedure to transfer installation log files to a USB drive.

Prerequisites

- Back up any data on the USB drive before using this procedure.
- You are logged into a root account and you have access to the installation program's temporary file system.

Procedure

1. Press **Ctrl+Alt+F2** to access a shell prompt on the system you are installing.
2. Connect a USB flash drive to the system and run the **dmesg** command:

```
# dmesg
```

A log detailing all recent events is displayed. At the end of this log, a set of messages is displayed. For example:

```
[ 170.171135] sd 5:0:0:0: [sdb] Attached SCSI removable disk
```

3. Note the name of the connected device. In the above example, it is **sdb**.
4. Navigate to the **/mnt** directory and create a new directory that serves as the mount target for the USB drive. This example uses the name **usb**:

```
# mkdir usb
```

5. Mount the USB flash drive onto the newly created directory. In most cases, you do not want to mount the whole drive, but a partition on it. Do not use the name **sdb**, use the name of the partition you want to write the log files to. In this example, the name **sdb1** is used:

```
# mount /dev/sdb1 /mnt/usb
```

6. Verify that you mounted the correct device and partition by accessing it and listing its contents:

```
# cd /mnt/usb
```

```
# ls
```

7. Copy the log files to the mounted device.

```
# cp /tmp/*log /mnt/usb
```

8. Unmount the USB flash drive. If you receive an error message that the target is busy, change your working directory to outside the mount (for example, **/**).

```
# umount /mnt/usb
```

A.1.2.3. Transferring installation log files over the network

Use this procedure to transfer installation log files over the network.

Prerequisites

- You are logged into a root account and you have access to the installation program's temporary file system.

Procedure

1. Press **Ctrl+Alt+F2** to access a shell prompt on the system you are installing.
2. Switch to the **/tmp** directory where the log files are located:

```
# cd /tmp
```

3. Copy the log files onto another system on the network using the **scp** command:

```
# scp *log user@address:path
```

- a. Replace **user** with a valid user name on the target system, **address** with the target system's

address or host name, and **path** with the path to the directory where you want to save the log files. For example, if you want to log in as **john** on a system with an IP address of 192.168.0.122 and place the log files into the **/home/john/logs/** directory on that system, the command is as follows:

```
# scp *log john@192.168.0.122:/home/john/logs/
```

When connecting to the target system for the first time, the SSH client asks you to confirm that the fingerprint of the remote system is correct and that you want to continue:

The authenticity of host '192.168.0.122 (192.168.0.122)' can't be established.
ECDSA key fingerprint is a4:60:76:eb:b2:d0:aa:23:af:3d:59:5c:de:bb:c4:42.
Are you sure you want to continue connecting (yes/no)?

- b. Type **yes** and press **Enter** to continue. Provide a valid password when prompted. The files are transferred to the specified directory on the target system.

A.1.3. Detecting memory faults using the Memtest86 application

Faults in memory (RAM) modules can cause your system to fail unpredictably. In certain situations, memory faults might only cause errors with particular combinations of software. For this reason, you should test your system's memory before you install Red Hat Enterprise Linux.



NOTE

Red Hat Enterprise Linux includes the **Memtest86+** memory testing application for BIOS systems only. Support for UEFI systems is currently unavailable.

A.1.3.1. Running Memtest86

Use this procedure to run the **Memtest86** application to test your system's memory for faults before you install Red Hat Enterprise Linux.

Prerequisites

- You have accessed the Red Hat Enterprise Linux boot menu.

Procedure

1. From the Red Hat Enterprise Linux boot menu, select **Troubleshooting > Run a memory test**. The **Memtest86** application window is displayed and testing begins immediately. By default, **Memtest86** performs ten tests in every pass. After the first pass is complete, a message is displayed in the lower part of the window informing you of the current status. Another pass starts automatically. If **Memtest86+** detects an error, the error is displayed in the central pane of the window and is highlighted in red. The message includes detailed information such as which test detected a problem, the memory location that is failing, and others. In most cases, a single successful pass of all 10 tests is sufficient to verify that your RAM is in good condition. In rare circumstances, however, errors that went undetected during the first pass might appear on subsequent passes. To perform a thorough test on important systems, run the tests overnight or for a few days to complete multiple passes.



NOTE

The amount of time it takes to complete a single full pass of **Memtest86+** varies depending on your system's configuration, notably the RAM size and speed. For example, on a system with 2 GiB of DDR2 memory at 667 MHz, a single pass takes 20 minutes to complete.

2. Optional: Follow the on-screen instructions to access the **Configuration** window and specify a different configuration.
3. To halt the tests and reboot your computer, press the **Esc** key at any time.

Additional resources

- For more information about using **Memtest86**, see the official website at <http://www.memtest.org/>.

A.1.4. Verifying boot media

Verifying ISO images helps to avoid problems that are sometimes encountered during installation. These sources include DVD and ISO images stored on a hard drive or NFS server. Use this procedure to test the integrity of an ISO-based installation source before using it to install Red Hat Enterprise Linux.

Prerequisites

- You have accessed the Red Hat Enterprise Linux boot menu.

Procedure

1. From the boot menu, select **Test this media & install Red Hat Enterprise Linux 8.1** to test the boot media.
2. The boot process tests the media and highlights any issues.
3. Optional: You can start the verification process by appending **rd.live.check** to the boot command line.

A.1.5. Consoles and logging during installation

The Red Hat Enterprise Linux installer uses the **tmux** terminal multiplexer to display and control several windows in addition to the main interface. Each of these windows serve a different purpose; they display several different logs, which can be used to troubleshoot issues during the installation process. One of the windows provides an interactive shell prompt with **root** privileges, unless this prompt was specifically disabled using a boot option or a Kickstart command.



NOTE

In general, there is no reason to leave the default graphical installation environment unless you need to diagnose an installation problem.

The terminal multiplexer is running in virtual console 1. To switch from the actual installation environment to **tmux**, press **Ctrl+Alt+F1**. To go back to the main installation interface which runs in virtual console 6, press **Ctrl+Alt+F6**.



NOTE

If you choose text mode installation, you will start in virtual console 1 (**tmux**), and switching to console 6 will open a shell prompt instead of a graphical interface.

The console running **tmux** has five available windows; their contents are described in the following table, along with keyboard shortcuts. Note that the keyboard shortcuts are two-part: first press **Ctrl+b**, then release both keys, and press the number key for the window you want to use.

You can also use **Ctrl+b n**, **Alt+ Tab**, and **Ctrl+b p** to switch to the next or previous **tmux** window, respectively.

Table A.2. Available tmux windows

Shortcut	Contents
Ctrl+b 1	Main installation program window. Contains text-based prompts (during text mode installation or if you use VNC direct mode), and also some debugging information.
Ctrl+b 2	Interactive shell prompt with root privileges.
Ctrl+b 3	Installation log; displays messages stored in /tmp/anaconda.log .
Ctrl+b 4	Storage log; displays messages related to storage devices and configuration, stored in /tmp/storage.log .
Ctrl+b 5	Program log; displays messages from utilities executed during the installation process, stored in /tmp/program.log .

A.1.6. Saving screenshots

You can press **Shift+Print Screen** at any time during the graphical installation to capture the current screen. The screenshots are saved to **/tmp/anaconda-screenshots**.

A.1.7. Resuming an interrupted download attempt

You can resume an interrupted download using the **curl** command.

Prerequisite

- You have navigated to the **Product Downloads** section of the Red Hat Customer Portal at <https://access.redhat.com/downloads>, and selected the required variant, version, and architecture.
- You have right-clicked on the required ISO file, and selected **Copy Link Location** to copy the URL of the ISO image file to your clipboard.

Procedure

Procedure

1. Download the ISO image from the new link. Add the **--continue-at** - option to automatically resume the download:

```
$ curl --output directory-path/filename.iso 'new_copied_link_location' --continue-at -
```

2. Use a checksum utility such as **sha256sum** to verify the integrity of the image file after the download finishes:

```
$ sha256sum rhel-8.1-x86_64-dvd.iso
`85a...46c rhel-8.1-x86_64-dvd.iso'
```

Compare the output with reference checksums provided on the Red Hat Enterprise Linux **Product Download** web page.

Example A.1. Resuming an interrupted download attempt

The following is an example of a **curl** command for a partially downloaded ISO image:

```
$ curl --output _rhel-8.1-x86_64-dvd.iso
'https://access.cdn.redhat.com/content/origin/files/sha256/85/85a...46c/rhel-8.1-x86_64-dvd.iso?
_auth=141...963' --continue-at -
```

A.1.8. Cannot boot into the graphical installation

Some video cards have trouble booting into the Red Hat Enterprise Linux graphical installation program. If the installation program does not run using its default settings, it attempts to run in a lower resolution mode. If that fails, the installation program attempts to run in text mode. There are several possible solutions to resolve display issues, most of which involve specifying custom boot options. For more information, see [Section D.3, “Console boot options”](#).

Table A.3. Solutions

Solution	Description
Use the basic graphics mode	You can attempt to perform the installation using the basic graphics driver. To do this, either select Troubleshooting > Install Red Hat Enterprise Linux 8.1 in basic graphics mode from the boot menu, or edit the installation program’s boot options and append inst.xdriver=vesa at the end of the command line.
Specify the display resolution manually	If the installation program fails to detect your screen resolution, you can override the automatic detection and specify it manually. To do this, append the inst.resolution=x option at the boot menu, where x is your display’s resolution, for example, 1024x768.

Solution	Description
Use an alternate video driver	You can attempt to specify a custom video driver, overriding the installation program's automatic detection. To specify a driver, use the <code>inst.xdriver=x</code> option, where x is the device driver you want to use (for example, <code>nouveau</code>)*.
Perform the installation using VNC	If the above options fail, you can use a separate system to access the graphical installation over the network, using the Virtual Network Computing (VNC) protocol. For details on installing using VNC, see the Performing a remote RHEL installation using VNC section of the Performing an advanced RHEL installation document.

*If specifying a custom video driver solves your problem, you should report it as a bug at <https://bugzilla.redhat.com> under the **anaconda** component. The installation program should be able to detect your hardware automatically and use the appropriate driver without intervention.

A.2. TROUBLESHOOTING DURING THE INSTALLATION

The troubleshooting information in the following sections might be helpful when diagnosing issues during the installation process. The following sections are for all supported architectures. However, if an issue is for a particular architecture, it is specified at the start of the section.

A.2.1. Disks are not detected

If the installation program cannot find a writable storage device to install to, it returns the following error message in the **Installation Destination** window: **No disks detected. Please shut down the computer, connect at least one disk, and restart to complete installation.**

Check the following items:

- Your system has at least one storage device attached.
- If your system uses a hardware RAID controller; verify that the controller is properly configured and working as expected. See your controller's documentation for instructions.
- If you are installing into one or more iSCSI devices and there is no local storage present on the system, verify that all required LUNs are presented to the appropriate Host Bus Adapter (HBA).

If the error message is still displayed after rebooting the system and starting the installation process, the installation program failed to detect the storage. In many cases the error message is a result of attempting to install on an iSCSI device that is not recognized by the installation program.

In this scenario, you must perform a driver update before starting the installation. Check your hardware vendor's website to determine if a driver update is available. For more general information on driver updates, see the [Updating drivers during installation](#) section of the [Performing an advanced RHEL installation](#) document.

You can also consult the Red Hat Hardware Compatibility List, available at <https://access.redhat.com/ecosystem/search/#/category/Server>.

A.2.2. Reporting error messages to Red Hat Customer Support

If the graphical installation encounters an error, it displays the **unknown error** dialog box. You can send information about the error to Red Hat Customer Support. To send a report, you must enter your Customer Portal credentials. If you do not have a Customer Portal account, you can register at <https://www.redhat.com/wapps/ugc/register.html>. Automated error reporting requires a network connection.

Prerequisite

The graphical installation program encountered an error and displayed the **unknown error** dialog box.

Procedure

1. From the **unknown error** dialog box, click **Report Bug** to report the problem, or **Quit** to exit the installation.
 - a. Optionally, click **More Info...** to display a detailed output that might help determine the cause of the error. If you are familiar with debugging, click **Debug**. This displays the virtual terminal **tty1**, where you can request additional information. To return to the graphical interface from **tty1**, use the **continue** command.
2. Click **Report a bug to Red Hat Customer Support**
3. The **Red Hat Customer Support - Reporting Configuration** dialog box is displayed. From the **Basic** tab, enter your Customer Portal user name and password. If your network settings require you to use an HTTP or HTTPS proxy, you can configure it by selecting the **Advanced** tab and entering the address of the proxy server.
4. Complete all fields and click **OK**.
5. A text box is displayed. Explain each step that was taken before the **unknown error** dialog box was displayed.
6. Select an option from the **How reproducible is this problem** drop-down menu and provide additional information in the text box.
7. Click **Forward**.
8. Verify that all the information you provided is in the **Comment** tab. The other tabs include information such as your system's host name and other details about your installation environment. You can remove any of the information that you do not want to send to Red Hat, but be aware that providing less detail might affect the investigation of the issue.
9. Click **Forward** when you have finished reviewing all tabs.
10. A dialog box displays all the files that will be sent to Red Hat. Clear the check boxes beside the files that you do not want to send to Red Hat. To add a file, click **Attach a file**.
11. Select the check box **I have reviewed the data and agree with submitting it**.
12. Click **Forward** to send the report and attachments to Red Hat.
13. Click **Show log** to view the details of the reporting process or click **Close** to return to the **unknown error** dialog box.
14. Click **Quit** to exit the installation.

A.2.3. Partitioning issues for IBM Power Systems



NOTE

This issue is for IBM Power Systems.

If you manually created partitions, but cannot move forward in the installation process, you might not have created all the partitions that are necessary for the installation to proceed. At a minimum, you must have the following partitions:

- **/ (root)** partition
- **PReP** boot partition
- **/boot** partition (only if the root partition is an LVM logical volume)

See [Section C.4, “Recommended partitioning scheme”](#) for more information.

A.3. TROUBLESHOOTING AFTER INSTALLATION

The troubleshooting information in the following sections might be helpful when diagnosing issues after the installation process. The following sections are for all supported architectures. However, if an issue is for a particular architecture, it is specified at the start of the section.

A.3.1. Cannot boot with a RAID card

If you cannot boot your system after the installation, you might need to reinstall and repartition your system’s storage. Some BIOS types do not support booting from RAID cards. After you finish the installation and reboot the system for the first time, a text-based screen displays the boot loader prompt (for example, **grub>**) and a flashing cursor might be displayed. If this is the case, you must repartition your system and move your **/boot** partition and the boot loader outside of the RAID array. The **/boot** partition and the boot loader must be on the same drive. Once these changes have been made, you should be able to finish your installation and boot the system properly.

A.3.2. Graphical boot sequence is not responding

When rebooting your system for the first time after installation, the system might be unresponsive during the graphical boot sequence. If this occurs, a reset is required. In this scenario, the boot loader menu is displayed successfully, but selecting any entry and attempting to boot the system results in a halt. This usually indicates that there is a problem with the graphical boot sequence. To resolve the issue, you must disable the graphical boot by temporarily altering the setting at boot time before changing it permanently.

Procedure: Disabling the graphical boot temporarily

1. Start your system and wait until the boot loader menu is displayed. If you set your boot timeout period to **0**, press the **Esc** key to access it.
2. From the boot loader menu, use your cursor keys to highlight the entry you want to boot. Press the **Tab** key on BIOS-based systems or the **e** key on UEFI-based systems to edit the selected entry options.
3. In the list of options, find the kernel line – that is, the line beginning with the keyword **linux**. On this line, locate and delete **rhgb**.

4. Press **F10** or **Ctrl+X** to boot your system with the edited options.

If the system started successfully, you can log in normally. However, if you do not disable graphical boot permanently, you must perform this procedure every time the system boots.

Procedure: Disabling the graphical boot permanently

1. Log in to the root account on your system.
2. Use the `grubby` tool to find the default GRUB2 kernel:

```
# grubby --default-kernel
/boot/vmlinuz-4.18.0-94.el8.x86_64
```

3. Use the `grubby` tool to remove the **rhgb** boot option from the default kernel in your GRUB2 configuration. For example:

```
# grubby --remove-args="rhgb" --update-kernel /boot/vmlinuz-4.18.0-94.el8.x86_64
```

4. Reboot the system. The graphical boot sequence is no longer used. If you want to enable the graphical boot sequence, follow the same procedure, replacing the **--remove-args="rhgb"** parameter with the **--args="rhgb"** parameter. This restores the **rhgb** boot option to the default kernel in your GRUB2 configuration.

A.3.3. X server fails after log in

An X server is a program in the X Window System that runs on local machines, that is, the computers used directly by users. X server handles all access to the graphics cards, display screens and input devices, typically a keyboard and mouse on those computers. The X Window System, often referred to as X, is a complete, cross-platform and free client-server system for managing GUIs on single computers and on networks of computers. The client-server model is an architecture that divides the work between two separate but linked applications, referred to as clients and servers.*

If X server crashes after login, one or more of the file systems might be full. To troubleshoot the issue, execute the following command:

```
$ df -h
```

The output verifies which partition is full – in most cases, the problem is on the **/home** partition. The following is a sample output of the `df` command:

Filesystem	Size	Used	Avail	Use%	Mounted on
devtmpfs	396M	0	396M	0%	/dev
tmpfs	411M	0	411M	0%	/dev/shm
tmpfs	411M	6.7M	405M	2%	/run
tmpfs	411M	0	411M	0%	/sys/fs/cgroup
/dev/mapper/rhel-root	17G	4.1G	13G	25%	/
/dev/sda1	1014M	173M	842M	17%	/boot
tmpfs	83M	20K	83M	1%	/run/user/42
tmpfs	83M	84K	83M	1%	/run/user/1000
/dev/dm-4	90G	90G	0	100%	/home

In the example, you can see that the **/home** partition is full, which causes the failure. Remove any unwanted files. After you free up some disk space, start X using the `startx` command. For additional

information about **df** and an explanation of the options available, such as the **-h** option used in this example, see the **df(1)** man page.

*Source: http://www.linfo.org/x_server.html

A.3.4. RAM is not recognized

In some scenarios, the kernel does not recognize all memory (RAM), which causes the system to use less memory than is installed. You can find out how much RAM is being utilized using the free **-m** command. If the total amount of memory does not match your expectations, it is likely that at least one of your memory modules is faulty. On BIOS-based systems, you can use the **Memtest86+** utility to test your system's memory.

Some hardware configurations have part of the system's RAM reserved, and as a result, it is unavailable to the system. Some laptop computers with integrated graphics cards reserve a portion of memory for the GPU. For example, a laptop with 4 GiB of RAM and an integrated Intel graphics card shows roughly 3.7 GiB of available memory. Additionally, the kdump crash kernel dumping mechanism, which is enabled by default on most Red Hat Enterprise Linux systems, reserves some memory for the secondary kernel used in case of a primary kernel failure. This reserved memory is not displayed as available when using the free command.

Procedure: Manually configuring the memory

Use this procedure to manually set the amount of memory using the **mem=** kernel option.

1. Start your system and wait until the boot loader menu is displayed. If you set your boot timeout period to **0**, press the **Esc** key to access it.
2. From the boot loader menu, use your cursor keys to highlight the entry you want to boot, and press the **Tab** key on BIOS-based systems or the **e** key on UEFI-based systems to edit the selected entry options.
3. In the list of options, find the kernel line – that is, the line beginning with the keyword **linux**. Append the following option to the end of this line:


```
mem=xxM
```
4. Replace **xx** with the amount of RAM you have in MiB.
5. Press **F10** or **Ctrl+X** to boot your system with the edited options.
6. Wait for the system to boot and then log in.
7. Open a command line and execute the free **-m** command again. If the total amount of RAM displayed by the command matches your expectations, append the following to the line beginning with **GRUB_CMDLINE_LINUX** in the **/etc/default/grub** file to make the change permanent:


```
# grub2-mkconfig --output=/boot/grub2/grub.cfg
```

A.3.5. System is displaying signal 11 errors

A signal 11 error, commonly known as a segmentation fault means that a program accessed a memory location that it was not assigned. A signal 11 error can occur due to a bug in one of the software programs that are installed, or faulty hardware. If you receive a signal 11 error during the installation process, verify

that you are using the most recent installation images and prompt the installation program to verify them to ensure they are not corrupt. For more information, see [Section A.1.4, “Verifying boot media”](#).

Faulty installation media (such as an improperly burned or scratched optical disk) are a common cause of signal 11 errors. Verifying the integrity of the installation media is recommended before every installation. For information about obtaining the most recent installation media, see [Section 3.5, “Downloading the installation ISO image”](#).

To perform a media check before the installation starts, append the **rd.live.check** boot option at the boot menu. If you performed a media check without any errors and you still have issues with segmentation faults, it usually indicates that your system encountered a hardware error. In this scenario, the problem is most likely in the system’s memory (RAM). This can be a problem even if you previously used a different operating system on the same computer without any errors.

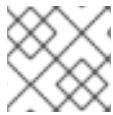


NOTE

For AMD and Intel 64-bit and 64-bit ARM architectures: On BIOS-based systems, you can use the **Memtest86+** memory testing module included on the installation media to perform a thorough test of your system’s memory. For more information, see [Section A.1.3, “Detecting memory faults using the Memtest86 application”](#).

Other possible causes are beyond this document’s scope. Consult your hardware manufacturer’s documentation and also see the Red Hat Hardware Compatibility List, available online at <https://access.redhat.com/ecosystem/search/#/category/Server>.

A.3.6. Unable to IPL from network storage space



NOTE

This issue is for IBM Power Systems.

If you experience difficulties when trying to IPL from Network Storage Space (*NWSSTG), it is most likely due to a missing PReP partition. In this scenario, you must reinstall the system and create this partition during the partitioning phase or in the Kickstart file.

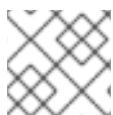
A.3.7. Using XDMCP

There are scenarios where you have installed the X Window System and want to log in to your Red Hat Enterprise Linux system using a graphical login manager. Use this procedure to enable the X Display Manager Control Protocol (XDMCP) and remotely log in to a desktop environment from any X-compatible client, such as a network-connected workstation or X11 terminal.



NOTE

XDMCP is not supported by the Wayland protocol. For more information, see the [Using the desktop environment in RHEL 8](#) document.



NOTE

This issue is for IBM Z.

Procedure

1. Open the **/etc/gdm/custom.conf** configuration file in a plain text editor such as **vi** or **nano**.
2. In the **custom.conf** file, locate the section starting with **[xdmcp]**. In this section, add the following line:

```
Enable=true
```

3. Save the file and exit the text editor.
4. Restart the X Window System. To do this, either reboot the system, or restart the GNOME Display Manager using the following command as root:

```
# systemctl restart gdm.service
```

5. Wait for the login prompt and log in using your user name and password. The X Window System is now configured for XDMCP. You can connect to it from another workstation (client) by starting a remote X session using the X command on the client workstation. For example:

```
$ X :1 -query address
```

6. Replace **address** with the host name of the remote X11 server. The command connects to the remote X11 server using XDMCP and displays the remote graphical login screen on display :1 of the X11 server system (usually accessible by pressing **Ctrl-Alt-F8**). You can also access remote desktop sessions using a nested X11 server, which opens the remote desktop as a window in your current X11 session. You can use Xnest to open a remote desktop nested in a local X11 session. For example, run Xnest using the following command, replacing address with the host name of the remote X11 server:

```
$ Xnest :1 -query address
```

For more information about XDMCP, see the X Window System documentation at <http://www.x.org/releases/X11R7.6/doc/libXdmcp/xdmcp.html>.

A.3.8. Using rescue mode

The installation program's rescue mode is a minimal Linux environment that can be booted from the Red Hat Enterprise Linux DVD or other boot media. It contains command-line utilities for repairing a wide variety of issues. Rescue mode can be accessed from the **Troubleshooting** menu of the boot menu. In this mode, you can mount file systems as read-only, blacklist or add a driver provided on a driver disc, install or upgrade system packages, or manage partitions.



NOTE

The installation program's rescue mode is different from rescue mode (an equivalent to single-user mode) and emergency mode, which are provided as parts of the **systemd** system and service manager.

To boot into rescue mode, you must be able to boot the system using one of the Red Hat Enterprise Linux boot media, such as a minimal boot disc or USB drive, or a full installation DVD.



IMPORTANT

Advanced storage, such as iSCSI or zFCP devices, must be configured either using **dracut** boot options such as **rd.zfcp=** or **root=iscsi: options**, or in the CMS configuration file on IBM Z. It is not possible to configure these storage devices interactively after booting into rescue mode. For information about **dracut** boot options, see the **dracut.cmdline(7)** man page.

A.3.8.1. Booting into rescue mode

Use this procedure to boot into rescue mode.

Procedure

1. Boot the system from either minimal boot media, or a full installation DVD or USB drive, and wait for the boot menu to be displayed.
2. From the boot menu, either select **Troubleshooting > Rescue a Red Hat Enterprise Linux system** option, or append the **inst.rescue** option to the boot command line. To enter the boot command line, press the **Tab** key on BIOS-based systems or the **e** key on UEFI-based systems.
3. Optional: If your system requires a third-party driver provided on a driver disc to boot, append the **inst.dd=driver_name** to the boot command line:

inst.rescue inst.dd=driver_name
4. Optional: If a driver that is part of the Red Hat Enterprise Linux distribution prevents the system from booting, append the **modprobe.blacklist=** option to the boot command line:

inst.rescue modprobe.blacklist=driver_name
5. Press **Enter** (BIOS-based systems) or **Ctrl+X** (UEFI-based systems) to boot the modified option. Wait until the following message is displayed:

The rescue environment will now attempt to find your Linux installation and mount it under the directory: **/mnt/sysimage/**. You can then make any changes required to your system. Choose 1 to proceed with this step. You can choose to mount your file systems read-only instead of read-write by choosing 2. If for some reason this process does not work choose 3 to skip directly to a shell.

- 1) Continue
 - 2) Read-only mount
 - 3) Skip to shell
 - 4) Quit (Reboot)

If you select 1, the installation program attempts to mount your file system under the directory **/mnt/sysimage/**. You are notified if it fails to mount a partition. If you select 2, it attempts to mount your file system under the directory **/mnt/sysimage/**, but in read-only mode. If you select 3, your file system is not mounted.

6. Select 1 to continue. Once your system is in rescue mode, a prompt appears on VC (virtual console) 1 and VC 2. Use the **Ctrl+Alt+F1** key combination to access VC 1 and **Ctrl+Alt+F2** to access VC 2:

sh-4.2#

- Even if your file system is mounted, the default root partition while in rescue mode is a temporary root partition, not the root partition of the file system used during normal user mode (**multi-user.target** or **graphical.target**). If you selected to mount your file system and it mounted successfully, you can change the root partition of the rescue mode environment to the root partition of your file system by executing the following command:

```
sh-4.2# chroot /mnt/sysimage
```

This is useful if you need to run commands, such as **rpm**, that require your root partition to be mounted as `/`. To exit the chroot environment, type **exit** to return to the prompt.

- If you selected **3**, you can still try to mount a partition or LVM2 logical volume manually inside rescue mode by creating a directory, such as `/directory`, and typing the following command:

```
sh-4.2# mount -t xfs /dev/mapper/VolGroup00-LogVol02 /directory
```

In the above command, `/directory` is the directory that you created and `/dev/mapper/VolGroup00-LogVol02` is the LVM2 logical volume you want to mount. If the partition is a different type than XFS, replace the `xfs` string with the correct type (such as `ext4`).

- If you do not know the names of all physical partitions, use the following command to list them:

```
sh-4.2# fdisk -l
```

If you do not know the names of all LVM2 physical volumes, volume groups, or logical volumes, use the **pvdisplay**, **vgdisplay** or **lvdisplay** commands.

A.3.8.2. Using an SOS report in rescue mode

The **sosreport** command-line utility collects configuration and diagnostic information, such as the running kernel version, loaded modules, and system and service configuration files from the system. The utility output is stored in a tar archive in the `/var/tmp` directory. The **sosreport** utility is useful for analyzing system errors and troubleshooting. Use this procedure to capture an **sosreport** output in rescue mode.

Prerequisites

- You have booted into rescue mode.
- You have mounted the installed system `/` (**root**) partition in read-write mode.
- You have contacted Red Hat Support about your case and received a case number.

Procedure

- Change the root directory to the `/mnt/sysimage` directory:

```
sh-4.2# chroot /mnt/sysimage/
```

- Execute **sosreport** to generate an archive with system configuration and diagnostic information:

```
sh-4.2# sosreport
```



IMPORTANT

sosreport prompts you to enter your name and the case number you received from Red Hat Support. Use only letters and numbers because adding any of the following characters or spaces could render the report unusable:

```
# % & { } \<>> * ? / $ ~ ' " : @ + ` | =
```

3. Optional: If you want to transfer the generated archive to a new location using the network, it is necessary to have a network interface configured. In this scenario, use the dynamic IP addressing as no other steps required. However, when using static addressing, enter the following command to assign an IP address (for example 10.13.153.64/23) to a network interface, for example dev eth0:

```
bash-4.2# ip addr add 10.13.153.64/23 dev eth0
```

4. Exit the chroot environment:

```
sh-4.2# exit
```

5. Store the generated archive in a new location, from where it can be easily accessible:

```
sh-4.2# cp /mnt/sysimage/var/tmp/sosreport new_location
```

6. For transferring the archive through the network, use the **scp** utility:

```
sh-4.2# scp /mnt/sysimage/var/tmp/sosreport username@hostname:sosreport
```

Additional resources

- For general information about sosreport, see [What is an sosreport and how to create one in Red Hat Enterprise Linux?](#)
- For information about using sosreport in rescue mode, see [How to generate sosreport from the rescue environment](#).
- For information about generating an sosreport to a different location than /tmp, see [How do I make sosreport write to an alternative location?](#)
- For information about collecting an sosreport manually, see [Sosreport fails. What data should I provide in its place?](#)

A.3.8.3. Reinstalling the GRUB2 boot loader

In some scenarios, the GRUB2 boot loader is mistakenly deleted, corrupted, or replaced by other operating systems. Use this procedure to reinstall GRUB2 on the master boot record.

Prerequisites

- You have booted into rescue mode.
- You have mounted the installed system / (**root**) partition in read-write mode.

Procedure

1. Change the root partition:

```
sh-4.2# chroot /mnt/sysimage/
```

2. Reinstall the GRUB2 boot loader, where **install_device** is the boot device, typically, **/dev/sda**:

```
sh-4.2# /sbin/grub2-install install_device
```

3. Reboot the system.

A.3.8.4. Using RPM to add or remove a driver

Missing or malfunctioning drivers cause problems when booting the system. Rescue mode provides an environment in which you can add or remove a driver even when the system fails to boot. Wherever possible, it is recommended that you use the RPM package manager to remove malfunctioning drivers or to add updated or missing drivers. Use the following procedures to add or remove a driver.



IMPORTANT

When you install a driver from a driver disc, the driver disc updates all **initramfs** images on the system to use this driver. If a problem with a driver prevents a system from booting, you cannot rely on booting the system from another **initramfs** image.

Procedure: Adding a driver using RPM

Use this procedure to add a driver.

Prerequisites

- You have booted into rescue mode.
 - You have mounted the installed system in read-write mode.
1. Make the RPM package that contains the driver available. For example, mount a CD or USB flash drive and copy the RPM package to a location of your choice under **/mnt/sysimage/**, for example: **/mnt/sysimage/root/drivers/**.

2. Change the root directory to **/mnt/sysimage/**:

```
sh-4.2# chroot /mnt/sysimage/
```

3. Use the **rpm -ivh** command to install the driver package. For example, run the following command to install the **xorg-x11-drv-wacom** driver package from **/root/drivers/**:

```
sh-4.2# rpm -ivh /root/drivers/xorg-x11-drv-wacom-0.23.0-6.el7.x86_64.rpm
```



NOTE

The **/root/drivers/** directory in this chroot environment is the **/mnt/sysimage/root/drivers/** directory in the original rescue environment.

4. Exit the chroot environment:

```
sh-4.2# exit
```

Procedure: Removing a driver using RPM

Use this procedure to remove a driver.

Prerequisites

- You have booted into rescue mode.
 - You have mounted the installed system in read-write mode.
1. Change the root directory to the **/mnt/sysimage** directory:

```
sh-4.2# chroot /mnt/sysimage/
```

2. Use the **rpm -e** command to remove the driver package. For example, to remove the **xorg-x11-drv-wacom** driver package, run:

```
sh-4.2# rpm -e xorg-x11-drv-wacom
```

3. Exit the chroot environment:

```
sh-4.2# exit
```

If you cannot remove a malfunctioning driver for some reason, you can instead blacklist the driver so that it does not load at boot time.

4. When you have finished adding and removing drivers, reboot the system.

A.3.9. ip= boot option returns an error

Using the **ip=** boot option format **ip=[ip address]** for example, **ip=192.168.1.1** returns the error message **Fatal for argument 'ip=[insert ip here]'\n sorry, unknown value [ip address] refusing to continue.**

In previous releases of Red Hat Enterprise Linux, the boot option format was:

```
--ip=192.168.1.15 --netmask=255.255.255.0 --gateway=192.168.1.254 --nameserver=192.168.1.250  
--hostname=myhost1
```

However, in Red Hat Enterprise Linux 8, the boot option format is:

```
ip=192.168.1.15::192.168.1.254:255.255.255.0:myhost1::none: nameserver=192.168.1.250
```

To resolve the issue, use the format: **ip=ip::gateway:netmask:hostname:interface:none** where:

- **ip** specifies the client ip address. You can specify IPv6 addresses in square brackets, for example, **[2001:DB8::1]**.
- **gateway** is the default gateway. IPv6 addresses are also accepted.

- **netmask** is the netmask to be used. This can be either a full netmask, for example, 255.255.255.0, or a prefix, for example, **64**.
- **hostname** is the host name of the client system. This parameter is optional.

For more information, see [Section D.2, “Network boot options”](#).

APPENDIX B. SYSTEM REQUIREMENTS REFERENCE

This section provides information and guidelines for hardware, installation target, system, memory, and RAID when installing Red Hat Enterprise Linux.

B.1. HARDWARE COMPATIBILITY

Red Hat works closely with hardware vendors on supported hardware.

- To verify that your hardware is supported, see the Red Hat Hardware Compatibility List, available at <https://access.redhat.com/ecosystem/search/#/category/Server>.
- To view supported memory sizes or CPU counts, see <https://access.redhat.com/articles/rhel-limits> for information.

B.2. SUPPORTED INSTALLATION TARGETS

An installation target is a storage device that stores Red Hat Enterprise Linux and boots the system. Red Hat Enterprise Linux supports the following installation targets for AMD64, Intel 64, and 64-bit ARM systems:

- Storage connected by a standard internal interface, such as SCSI, SATA, or SAS
- BIOS/firmware RAID devices
- NVDIMM devices in sector mode on the Intel64 and AMD64 architectures, supported by the `nd_pmem` driver.
- Fibre Channel Host Bus Adapters and multipath devices. Some can require vendor-provided drivers.
- Xen block devices on Intel processors in Xen virtual machines.
- VirtIO block devices on Intel processors in KVM virtual machines.

Red Hat does not support installation to USB drives or SD memory cards. For information about support for third-party virtualization technologies, see the [Red Hat Hardware Compatibility List](#).

B.3. SYSTEM SPECIFICATIONS

The Red Hat Enterprise Linux installation program automatically detects and installs your system's hardware, so you should not have to supply any specific system information. However, for certain Red Hat Enterprise Linux installation scenarios, it is recommended that you record system specifications for future reference. These scenarios include:

Installing RHEL with a customized partition layout

Record: The model numbers, sizes, types, and interfaces of the hard drives attached to the system. For example, Seagate ST3320613AS 320 GB on SATA0, Western Digital WD7500AAKS 750 GB on SATA1.

Installing RHEL as an additional operating system on an existing system

Record: Partitions used on the system. This information can include file system types, device node names, file system labels, and sizes, and allows you to identify specific partitions during the partitioning process. If one of the operating systems is a Unix operating system, Red Hat Enterprise Linux may

report the device names differently. Additional information can be found by executing the equivalent of the **mount** command and the **blkid** command, and in the **/etc/fstab** file.

If multiple operating systems are installed, the Red Hat Enterprise Linux installation program attempts to automatically detect them, and to configure boot loader to boot them. You can manually configure additional operating systems if they are not detected automatically. See *Configuring boot loader* in [Section 5.6, “Configuring software options”](#) for more information.

Installing RHEL from an image on a local hard drive

Record: The hard drive and directory that holds the image.

Installing RHEL from a network location

If the network has to be configured manually, that is, DHCP is not used.

Record:

- IP address
- Netmask
- Gateway IP address
- Server IP addresses, if required

Contact your network administrator if you need assistance with networking requirements.

Installing RHEL on an iSCSI target

Record: The location of the iSCSI target. Depending on your network, you may need a CHAP user name and password, and a reverse CHAP user name and password.

Installing RHEL if the system is part of a domain

Verify that the domain name is supplied by the DHCP server. If it is not, enter the domain name during installation.

B.4. DISK AND MEMORY REQUIREMENTS

If several operating systems are installed, it is important that you verify that the allocated disk space is separate from the disk space required by Red Hat Enterprise Linux.



NOTE

- For AMD64, Intel 64, and 64-bit ARM, at least two partitions (**/** and **swap**) must be dedicated to Red Hat Enterprise Linux.
- For IBM Power Systems servers, at least three partitions (**/**, **swap**, and a **PReP** boot partition) must be dedicated to Red Hat Enterprise Linux.

You must have a minimum of 10 GiB of available disk space.

To install Red Hat Enterprise Linux, you must have a minimum of 10 GiB of space in either unpartitioned disk space or in partitions that can be deleted. See [Appendix C, Partitioning reference](#) for more information.

Table B.1. Minimum RAM requirements

Installation type	Recommended minimum RAM
Local media installation (USB, DVD)	768 MiB
NFS network installation	768 MiB
HTTP, HTTPS or FTP network installation	1.5 GiB

**NOTE**

It is possible to complete the installation with less memory than the recommended minimum requirements. The exact requirements depend on your environment and installation path. It is recommended that you test various configurations to determine the minimum required RAM for your environment. Installing Red Hat Enterprise Linux using a Kickstart file has the same recommended minimum RAM requirements as a standard installation. However, additional RAM may be required if your Kickstart file includes commands that require additional memory, or write data to the RAM disk. See the [Performing an advanced RHEL installation](#) document for more information.

B.5. RAID REQUIREMENTS

It is important to understand how storage technologies are configured and how support for them may have changed between major versions of Red Hat Enterprise Linux.

Hardware RAID

Any RAID functions provided by the mainboard of your computer, or attached controller cards, need to be configured before you begin the installation process. Each active RAID array appears as one drive within Red Hat Enterprise Linux.

Software RAID

On systems with more than one hard drive, you can use the Red Hat Enterprise Linux installation program to operate several of the drives as a Linux software RAID array. With a software RAID array, RAID functions are controlled by the operating system rather than the dedicated hardware.

**NOTE**

When a pre-existing RAID array's member devices are all unpartitioned disks/drives, the installation program treats the array as a disk and there is no method to remove the array.

USB Disks

You can connect and configure external USB storage after installation. Most devices are recognized by the kernel, but some devices may not be recognized. If it is not a requirement to configure these disks during installation, disconnect them to avoid potential problems.

NVDIMM devices

To use a Non-Volatile Dual In-line Memory Module (NVDIMM) device as storage, the following conditions must be satisfied:

- Version of Red Hat Enterprise Linux is 7.6 or later.
- The architecture of the system is Intel 64 or AMD64.
- The device is configured to sector mode. Anaconda can reconfigure NVDIMM devices to this mode.
- The device must be supported by the `nd_pmem` driver.

Booting from an NVDIMM device is possible under the following additional conditions:

- The system uses UEFI.
- The device must be supported by firmware available on the system, or by a UEFI driver. The UEFI driver may be loaded from an option ROM of the device itself.
- The device must be made available under a namespace.

To take advantage of the high performance of NVDIMM devices during booting, place the `/boot` and `/boot/efi` directories on the device.



NOTE

The Execute-in-place (XIP) feature of NVDIMM devices is not supported during booting and the kernel is loaded into conventional memory.

Considerations for Intel BIOS RAID Sets

Red Hat Enterprise Linux uses **mdraid** for installing on Intel BIOS RAID sets. These sets are automatically detected during the boot process and their device node paths can change across several booting processes. For this reason, local modifications to the `/etc/fstab`, `/etc/crypttab` or other configuration files that refer to the devices by their device node paths may not work in Red Hat Enterprise Linux. It is recommended that you replace device node paths (such as `/dev/sda`) with file system labels or device UUIDs. You can find the file system labels and device UUIDs using the **blkid** command.

APPENDIX C. PARTITIONING REFERENCE

C.1. SUPPORTED DEVICE TYPES

Standard partition

A standard partition can contain a file system or swap space. Standard partitions are most commonly used for **/boot** and the **BIOS Boot** and **EFI System partitions**. LVM logical volumes are recommended for most other uses.

LVM

Choosing **LVM** (or Logical Volume Management) as the device type creates an LVM logical volume. If no LVM volume group currently exists, one is automatically created to contain the new volume; if an LVM volume group already exists, the volume is assigned. LVM can improve performance when using physical disks, and it allows for advanced setups such as using multiple physical disks for one mount point, and setting up software RAID for increased performance, reliability, or both.

LVM thin provisioning

Using thin provisioning, you can manage a storage pool of free space, known as a thin pool, which can be allocated to an arbitrary number of devices when needed by applications. You can dynamically expand the pool when needed for cost-effective allocation of storage space.



WARNING

The installation program does not support overprovisioned LVM thin pools.

C.2. SUPPORTED FILE SYSTEMS

This section describes the file systems available in Red Hat Enterprise Linux.

xfs

XFS is a highly scalable, high-performance file system that supports file systems up to 16 exabytes (approximately 16 million terabytes), files up to 8 exabytes (approximately 8 million terabytes), and directory structures containing tens of millions of entries. **XFS** also supports metadata journaling, which facilitates quicker crash recovery. The maximum supported size of a single XFS file system is 500 TB. **XFS** is the default and recommended file system on Red Hat Enterprise Linux.

ext4

The **ext4** file system is based on the **ext3** file system and features a number of improvements. These include support for larger file systems and larger files, faster and more efficient allocation of disk space, no limit on the number of subdirectories within a directory, faster file system checking, and more robust journaling. The maximum supported size of a single **ext4** file system is 50 TB.

ext3

The **ext3** file system is based on the **ext2** file system and has one main advantage - journaling. Using a journaling file system reduces the time spent recovering a file system after it terminates unexpectedly, as there is no need to check the file system for metadata consistency by running the `fsck` utility every time.

ext2

An **ext2** file system supports standard Unix file types, including regular files, directories, or symbolic links. It provides the ability to assign long file names, up to 255 characters.

swap

Swap partitions are used to support virtual memory. In other words, data is written to a swap partition when there is not enough RAM to store the data your system is processing.

vfat

The **VFAT** file system is a Linux file system that is compatible with Microsoft Windows long file names on the FAT file system.



NOTE

Support for **VFAT** file system is not available for Linux system partitions. For example, **/**, **/var**, **/usr** and so on.

BIOS Boot

A very small partition required for booting from a device with a GUID partition table (GPT) on BIOS systems and UEFI systems in BIOS compatibility mode.

EFI System Partition

A small partition required for booting a device with a GUID partition table (GPT) on a UEFI system.

PReP

This small boot partition is located on the first partition of the hard drive. The **PReP** boot partition contains the GRUB2 boot loader, which allows other IBM Power Systems servers to boot Red Hat Enterprise Linux.

C.3. SUPPORTED RAID TYPES

RAID stands for Redundant Array of Independent Disks, a technology which allows you to combine multiple physical disks into logical units. Some setups are designed to enhance performance at the cost of reliability, while others improve reliability at the cost of requiring more disks for the same amount of available space.

This section describes supported software RAID types which you can use with LVM and LVM Thin Provisioning to set up storage on the installed system.

None

No RAID array is set up.

RAID 0

Performance: Distributes data across multiple disks. RAID 0 offers increased performance over standard partitions and can be used to pool the storage of multiple disks into one large virtual device. Note that RAID 0 offers no redundancy and that the failure of one device in the array destroys data in the entire array. RAID 0 requires at least two disks.

RAID 1

Redundancy: Mirrors all data from one partition onto one or more other disks. Additional devices in the array provide increasing levels of redundancy. RAID 1 requires at least two disks.

RAID 4

Error checking: Distributes data across multiple disks and uses one disk in the array to store parity information which safeguards the array in case any disk in the array fails. As all parity information is stored on one disk, access to this disk creates a "bottleneck" in the array's performance. RAID 4 requires at least three disks.

RAID 5

Distributed error checking: Distributes data and parity information across multiple disks. RAID 5 offers the performance advantages of distributing data across multiple disks, but does not share the performance bottleneck of RAID 4 as the parity information is also distributed through the array. RAID 5 requires at least three disks.

RAID 6

Redundant error checking: RAID 6 is similar to RAID 5, but instead of storing only one set of parity data, it stores two sets. RAID 6 requires at least four disks.

RAID 10

Performance and redundancy: RAID 10 is nested or hybrid RAID. It is constructed by distributing data over mirrored sets of disks. For example, a RAID 10 array constructed from four RAID partitions consists of two mirrored pairs of striped partitions. RAID 10 requires at least four disks.

C.4. RECOMMENDED PARTITIONING SCHEME

Red Hat recommends that you create separate file systems at the following mount points:

- **/boot**
- **/** (root)
- **/home**
- **swap**
- **/boot/efi**
- **PReP**

/boot partition - recommended size at least 1 GiB

The partition mounted on **/boot** contains the operating system kernel, which allows your system to boot Red Hat Enterprise Linux 8, along with files used during the bootstrap process. Due to the limitations of most firmwares, creating a small partition to hold these is recommended. In most scenarios, a 1 GiB boot partition is adequate. Unlike other mount points, using an LVM volume for **/boot** is not possible – **/boot** must be located on a separate disk partition.



WARNING

Normally, the **/boot** partition is created automatically by the installation program. However, if the **/** (root) partition is larger than 2 TiB and (U)EFI is used for booting, you need to create a separate **/boot** partition that is smaller than 2 TiB to boot the machine successfully.



NOTE

If you have a RAID card, be aware that some BIOS types do not support booting from the RAID card. In such a case, the **/boot** partition must be created on a partition outside of the RAID array, such as on a separate hard drive.

root - recommended size of 10 GiB

This is where "/", or the root directory, is located. The root directory is the top-level of the directory structure. By default, all files are written to this file system unless a different file system is mounted in the path being written to, for example, **/boot** or **/home**.

While a 5 GiB root file system allows you to install a minimal installation, it is recommended to allocate at least 10 GiB so that you can install as many package groups as you want.



IMPORTANT

Do not confuse the **/** directory with the **/root** directory. The **/root** directory is the home directory of the root user. The **/root** directory is sometimes referred to as *slash root* to distinguish it from the root directory.

/home - recommended size at least 1 GiB

To store user data separately from system data, create a dedicated file system for the **/home** directory. Base the file system size on the amount of data that is stored locally, number of users, and so on. You can upgrade or reinstall Red Hat Enterprise Linux 8 without erasing user data files. If you select automatic partitioning, it is recommended to have at least 55 GiB of disk space available for the installation, to ensure that the **/home** file system is created.

swap partition - recommended size at least 1 GB

Swap file systems support virtual memory; data is written to a swap file system when there is not enough RAM to store the data your system is processing. Swap size is a function of system memory workload, not total system memory and therefore is not equal to the total system memory size. It is important to analyze what applications a system will be running and the load those applications will serve in order to determine the system memory workload. Application providers and developers can provide guidance.

When the system runs out of swap space, the kernel terminates processes as the system RAM memory is exhausted. Configuring too much swap space results in storage devices being allocated but idle and is a poor use of resources. Too much swap space can also hide memory leaks. The maximum size for a swap partition and other additional information can be found in the **mkswap(8)** manual page.

The following table provides the recommended size of a swap partition depending on the amount of RAM in your system and if you want sufficient memory for your system to hibernate. If you let the installation program partition your system automatically, the swap partition size is established using these guidelines. Automatic partitioning setup assumes hibernation is not in use. The maximum size of the swap partition is limited to 10 percent of the total size of the hard drive, and the installation program cannot create swap partitions more than 128 GB in size. To set up enough swap space to allow for hibernation, or if you want to set the swap partition size to more than 10 percent of the system's storage space, or more than 128 GB, you must edit the partitioning layout manually.

/boot/efi partition - recommended size of 200 MiB

UEFI-based AMD64, Intel 64, and 64-bit ARM require a 200 MiB EFI system partition. The recommended minimum size is 200 MiB, the default size is 600 MiB, and the maximum size is 600 MiB. BIOS systems do not require an EFI system partition.

Table C.1. Recommended System Swap Space

Amount of RAM in the system	Recommended swap space	Recommended swap space if allowing for hibernation
Less than 2 GB	2 times the amount of RAM	3 times the amount of RAM
2 GB - 8 GB	Equal to the amount of RAM	2 times the amount of RAM
8 GB - 64 GB	4 GB to 0.5 times the amount of RAM	1.5 times the amount of RAM
More than 64 GB	Workload dependent (at least 4GB)	Hibernation not recommended

At the border between each range, for example, a system with 2 GB, 8 GB, or 64 GB of system RAM, discretion can be exercised with regard to chosen swap space and hibernation support. If your system resources allow for it, increasing the swap space can lead to better performance.

Distributing swap space over multiple storage devices - particularly on systems with fast drives, controllers and interfaces - also improves swap space performance.

Many systems have more partitions and volumes than the minimum required. Choose partitions based on your particular system needs.



NOTE

- Only assign storage capacity to those partitions you require immediately. You can allocate free space at any time, to meet needs as they occur.
- If you are unsure about how to configure partitions, accept the automatic default partition layout provided by the installation program.

PReP boot partition - recommended size of 4 to 8 MiB

When installing Red Hat Enterprise Linux on IBM Power System servers, the first partition of the hard drive should include a **PReP** boot partition. This contains the GRUB2 boot loader, which allows other IBM Power Systems servers to boot Red Hat Enterprise Linux.

C.5. ADVICE ON PARTITIONS

There is no best way to partition every system; the optimal setup depends on how you plan to use the system being installed. However, the following tips may help you find the optimal layout for your needs:

- Create partitions that have specific requirements first, for example, if a particular partition must be on a specific disk.
- Consider encrypting any partitions and volumes which might contain sensitive data. Encryption prevents unauthorized people from accessing the data on the partitions, even if they have

access to the physical storage device. In most cases, you should at least encrypt the **/home** partition, which contains user data.

- In some cases, creating separate mount points for directories other than **/**, **/boot** and **/home** may be useful; for example, on a server running a **MySQL** database, having a separate mount point for **/var/lib/mysql** will allow you to preserve the database during a reinstallation without having to restore it from backup afterward. However, having unnecessary separate mount points will make storage administration more difficult.
- Some special restrictions apply to certain directories with regards on which partitioning layouts can they be placed. Notably, the **/boot** directory must always be on a physical partition (not on an LVM volume).
- If you are new to Linux, consider reviewing the *Linux Filesystem Hierarchy Standard* at http://refspecs.linuxfoundation.org/FHS_2.3/fhs-2.3.html for information about various system directories and their contents.
- Each kernel installed on your system requires approximately 56 MB on the **/boot** partition:
 - 32 MB initramfs
 - 14 MB kdump initramfs
 - 3.5 MB system map
 - 6.6 MB vmlinuz



NOTE

For rescue mode, **initramfs** and **vmlinuz** require 80 MB.

The default partition size of 1 GB for **/boot** should suffice for most common use cases. However, it is recommended that you increase the size of this partition if you are planning on retaining multiple kernel releases or errata kernels.

- The **/var** directory holds content for a number of applications, including the **Apache** web server, and is used by the **DNF** package manager to temporarily store downloaded package updates. Make sure that the partition or volume containing **/var** has at least 3 GB.
 - The contents of the **/var** directory usually change very often. This may cause problems with older solid state drives (SSDs), as they can handle a lower number of read/write cycles before becoming unusable. If your system root is on an SSD, consider creating a separate mount point for **/var** on a classic (platter) HDD.
 - The **/usr** directory holds the majority of software on a typical Red Hat Enterprise Linux installation. The partition or volume containing this directory should therefore be at least 5 GB for minimal installations, and at least 10 GB for installations with a graphical environment.
 - If **/usr** or **/var** is partitioned separately from the rest of the root volume, the boot process becomes much more complex because these directories contain boot-critical components. In some situations, such as when these directories are placed on an iSCSI drive or an FCoE location, the system may either be unable to boot, or it may hang with a **Device is busy** error when powering off or rebooting.
- This limitation only applies to **/usr** or **/var**, not to directories below them. For example, a separate partition for **/var/www** will work without issues.

- Consider leaving a portion of the space in an LVM volume group unallocated. This unallocated space gives you flexibility if your space requirements change but you do not wish to remove data from other volumes. You can also select the **LVM Thin Provisioning** device type for the partition to have the unused space handled automatically by the volume.
- The size of an XFS file system can not be reduced - if you need to make a partition or volume with this file system smaller, you must back up your data, destroy the file system, and create a new, smaller one in its place. Therefore, if you expect needing to manipulate your partitioning layout later, you should use the ext4 file system instead.
- Use Logical Volume Management (LVM) if you anticipate expanding your storage by adding more hard drives or expanding virtual machine hard drives after the installation. With LVM, you can create physical volumes on the new drives, and then assign them to any volume group and logical volume as you see fit - for example, you can easily expand your system's **/home** (or any other directory residing on a logical volume).
- Creating a BIOS Boot partition or an EFI System Partition may be necessary, depending on your system's firmware, boot drive size, and boot drive disk label. See [Section C.4, "Recommended partitioning scheme"](#) for information about these partitions. Note that graphical installation will not let you create a BIOS Boot or EFI System Partition if your system does **not** require one - in that case, they will be hidden from the menu.
- If you need to make any changes to your storage configuration after the installation, Red Hat Enterprise Linux repositories offer several different tools which can help you do this. If you prefer a command line tool, try **system-storage-manager**.

APPENDIX D. BOOT OPTIONS REFERENCE

This section contains information about some of the boot options that you can use to modify the default behavior of the installation program. For Kickstart and advanced boot options, see the [Performing an advanced RHEL installation](#) document.

D.1. INSTALLATION SOURCE BOOT OPTIONS

This section contains information about the various installation source boot options.

inst.repo=

The **inst.repo=** boot option specifies the installation source, that is, the location providing the package repositories and a valid **.treeinfo** file that describes them. For example: **inst.repo=cdrom**. The target of the **inst.repo=** option must be one of the following installation media:

- an installable tree, which is a directory structure containing the installation program images, packages, and repository data as well as a valid **.treeinfo** file
- a DVD (a physical disk present in the system DVD drive)
- an ISO image of the full Red Hat Enterprise Linux installation DVD, placed on a hard drive or a network location accessible to the system.

You can use the **inst.repo=** boot option to configure different installation methods using different formats. The following table contains details of the **inst.repo=** boot option syntax:

Table D.1. inst.repo= installation source boot options

Source type	Boot option format	Source format
CD/DVD drive	inst.repo=cdrom[:device]	Installation DVD as a physical disk. [a]
Installable tree	inst.repo=hd:device:/path	Image file of the installation DVD, or an installable tree, which is a complete copy of the directories and files on the installation DVD.
NFS Server	inst.repo=nfs:[options:]server:/path	Image file of the installation DVD. [b]
HTTP Server	inst.repo=http://host/path	Installable tree, which is a complete copy of the directories and files on the installation DVD.
HTTPS Server	inst.repo=https://host/path	Installable tree, which is a complete copy of the directories and files on the installation DVD.
FTP Server	inst.repo=ftp://username:password@host/path	
HMC	inst.repo=hmc	

Source type	Boot option format	Source format
	<p>[a] If device is left out, installation program automatically searches for a drive containing the installation DVD.</p> <p>[b] The NFS Server option uses NFS protocol version 3 by default. To use a different version <i>X</i>, add +nfsvers=X to <i>options</i>.</p>	



NOTE

The NFS Server option uses NFS protocol version 3 by default. To use a different version, add **+nfsvers=X** to the option.

You can set disk device names with the following formats:

- Kernel device name, for example **/dev/sda1** or **sdb2**
- File system label, for example **LABEL=Flash** or **LABEL=RHEL8**
- File system UUID, for example **UUID=8176c7bf-04ff-403a-a832-9557f94e61db**
Non-alphanumeric characters must be represented as **\xNN**, where *NN* is the hexadecimal representation of the character. For example, **\x20** is a white space (" ").

inst.addrepo=

Use the **inst.addrepo=** boot option to add an additional repository that can be used as another installation source along with the main repository (**inst.repo=**). You can use the **inst.addrepo=** boot option multiple times during one boot. The following table contains details of the **inst.addrepo=** boot option syntax.



NOTE

The **REPO_NAME** is the name of the repository and is required in the installation process. These repositories are only used during the installation process; they are not installed on the installed system.

Table D.2. inst.addrepo installation source boot options

Installation source	Boot option format	Additional information
Installable tree at a URL	inst.addrepo=REPO_NAME, [http,https,ftp]://<host>/<path>	Looks for the installable tree at a given URL.
Installable tree at an NFS path	inst.addrepo=REPO_NAME, nfs://<server>/<path>	Looks for the installable tree at a given NFS path. A colon is required after the host. The installation program passes every thing after nfs:// directly to the mount command instead of parsing URLs according to RFC 2224.

Installation source	Boot option format	Additional information
Installable tree in the installation environment	inst.addrepo=REPO_NAME, file://<path>	Looks for the installable tree at the given location in the installation environment. To use this option, the repository must be mounted before the installation program attempts to load the available software groups. The benefit of this option is that you can have multiple repositories on one bootable ISO, and you can install both the main repository and additional repositories from the ISO. The path to the additional repositories is /run/install/source/REPO_IS_O_PATH . Additional, you can mount the repository directory in the %pre section in the Kickstart file. The path must be absolute and start with /, for example inst.addrepo=REPO_NAME, file:///<path>
Hard Drive	inst.addrepo=REPO_NAME, hd:<device>:<path>	Mounts the given <device> partition and installs from the ISO that is specified by the <path> . If the <path> is not specified, the installation program looks for a valid installation ISO on the <device> . This installation method requires an ISO with a valid installable tree.

inst.noverifyssl=

The **noverifyssl=** boot option prevents the installation program from verifying the SSL certificate for all HTTPS connections with the exception of the additional Kickstart repositories, where **--noverifyssl** can be set per repository.

inst.stage2=

Use the **inst.stage2=** boot option to specify the location of the installation program runtime image. This option expects a path to a directory containing a valid **.treeinfo** file. The location of the runtime image is read from the **.treeinfo** file. If the **.treeinfo** file is not available, the installation program attempts to load the image from **images/install.img**.

When the **inst.stage2** option is not specified, the installation program attempts to use the location specified with **inst.repo** option.

You should specify this option only for PXE boot. The installation DVD and Boot ISO already contain a correct **inst.stage2** option to boot the installation program from themselves.



NOTE

By default, the **inst.stage2=** boot option is used on the installation media and is set to a specific label, for example, **inst.stage2=hd:LABEL=RHEL-8-0-0-BaseOS-x86_64**. If you modify the default label of the file system containing the runtime image, or if you use a customized procedure to boot the installation system, you must verify that the **inst.stage2=** boot option is set to the correct value.

inst.stage2.all

The **inst.stage2.all** boot option is used to specify several HTTP, HTTPS, or FTP sources. You can use the **inst.stage2=** boot option multiple times with the **inst.stage2.all** option to fetch the image from the sources sequentially until one succeeds. For example:

```
inst.stage2.all
inst.stage2=http://hostname1/path_to_install_tree/
inst.stage2=http://hostname2/path_to_install_tree/
inst.stage2=http://hostname3/path_to_install_tree/
```

inst.dd=

The **inst.dd=** boot option is used to perform a driver update during the installation. See the [Performing an advanced RHEL installation](#) document for information on how to update drivers during installation.

inst.repo=hmc

When booting from a Binary DVD, the installation program prompts you to enter additional kernel parameters. To set the DVD as an installation source, append **inst.repo=hmc** to the kernel parameters. The installation program then enables **SE** and **HMC** file access, fetches the images for stage2 from the DVD, and provides access to the packages on the DVD for software selection. This option eliminates the requirement of an external network setup and expands the installation options.

inst.proxy

The **inst.proxy** boot option is used when performing an installation from a HTTP, HTTPS, FTP source. For example:

```
[PROTOCOL://][USERNAME[:PASSWORD]@]HOST[:PORT]
```

inst.nosave

Use the **inst.nosave** boot option to control which installation logs and related files are not saved to the installed system, for example **input_ks**, **output_ks**, **all_ks**, **logs** and **all**. Multiple values can be combined as a comma-separated list, for example: **input_ks,logs**.



NOTE

The **inst.nosave** boot option is used for excluding files from the installed system that can't be removed by a Kickstart %post script, such as logs and input/output Kickstart results.

Table D.3. inst.nosave boot options

Option	Description
--------	-------------

Option	Description
input_ks	Disables the ability to save the input Kickstart results.
output_ks	Disables the ability to save the output Kickstart results generated by the installation program.
all_ks	Disables the ability to save the input and output Kickstart results.
logs	Disables the ability to save all installation logs.
all	Disables the ability to save all Kickstart results, and all logs.

inst.multilib

Use the **inst.multilib** boot option to set DNF’s **multilib_policy** to **all**, instead of **best**.

memcheck

The **memcheck** boot option performs a check to verify that the system has enough RAM to complete the installation. If there isn’t enough RAM, the installation process is stopped. The system check is approximate and memory usage during installation depends on the package selection, user interface, for example graphical or text, and other parameters.

nomemcheck

The **nomemcheck** boot option does not perform a check to verify if the system has enough RAM to complete the installation. Any attempt to perform the installation with less than the recommended minimum amount of memory is unsupported, and might result in the installation process failing.

D.2. NETWORK BOOT OPTIONS

This section contains information about commonly used network boot options.



NOTE

Initial network initialization is handled by **dracut**. For a complete list, see the **dracut.cmdline(7)** man page.

ip=

Use the **ip=** boot option to configure one or more network interfaces. To configure multiple interfaces, you can use the **ip** option multiple times, once for each interface; to do so, you must use the **rd.neednet=1** option, and you must specify a primary boot interface using the **bootdev** option. Alternatively, you can use the **ip** option once, and then use Kickstart to set up further interfaces. This option accepts several different formats. The following tables contain information about the most common options.



NOTE

In the following tables:

- The **ip** parameter specifies the client IP address and requires square brackets, for example [2001:db8::99].
- The **gateway** parameter is the default gateway. IPv6 addresses are also accepted.
- The **netmask** parameter is the netmask to be used. This can be either a full netmask (for example, 255.255.255.0) or a prefix (for example, 64).
- The **hostname** parameter is the host name of the client system. This parameter is optional.

Table D.4. Network interface configuration boot option formats

Configuration method	Boot option format
Automatic configuration of any interface	ip=method
Automatic configuration of a specific interface	ip=interface:method
Static configuration	ip=ip::gateway:netmask:hostname:interface:none
Automatic configuration of a specific interface with an override	ip=ip::gateway:netmask:hostname:interface:method:mtu



NOTE

The method **automatic configuration of a specific interface with an override** brings up the interface using the specified method of automatic configuration, such as **dhcp**, but overrides the automatically-obtained IP address, gateway, netmask, host name or other specified parameters. All parameters are optional, so specify only the parameters that you want to override.

The **method** parameter can be any of the following:

Table D.5. Automatic interface configuration methods

Automatic configuration method	Value
DHCP	dhcp
IPv6 DHCP	dhcp6
IPv6 automatic configuration	auto6
iSCSI Boot Firmware Table (iBFT)	ibft



NOTE

- If you use a boot option that requires network access, such as **inst.ks=http://host:/path**, without specifying the **ip** option, the installation program uses **ip=dhcp**.
- To connect to an iSCSI target automatically, you must activate a network device for accessing the target. The recommended way to activate a network is to use the **ip=ibft** boot option.

nameserver=

The **nameserver=** option specifies the address of the name server. You can use this option multiple times.



NOTE

The **ip=** parameter requires square brackets. However, an IPv6 address does not work with square brackets. An example of the correct syntax to use for an IPv6 address is **nameserver=2001:db8::1**.

bootdev=

The **bootdev=** option specifies the boot interface. This option is mandatory if you use more than one **ip** option.

ifname=

The **ifname=** options assigns an interface name to a network device with a given MAC address. You can use this option multiple times. The syntax is **ifname=interface:MAC**. For example:

```
ifname=eth0:01:23:45:67:89:ab
```



NOTE

The **ifname=** option is the only supported way to set custom network interface names during installation.

inst.dhcpclass=

The **inst.dhcpclass=** option specifies the DHCP vendor class identifier. The **dhcpd** service sees this value as **vendor-class-identifier**. The default value is **anaconda-\$(uname -srm)**.

inst.waitfornet=

Using the **inst.waitfornet=SECONDS** boot option causes the installation system to wait for network connectivity before installation. The value given in the **SECONDS** argument specifies the maximum amount of time to wait for network connectivity before timing out and continuing the installation process even if network connectivity is not present.

Additional resources

- For more information about networking, see the [Configuring and managing networking](#) document.

D.3. CONSOLE BOOT OPTIONS

This section contains information about configuring boot options for your console, monitor display, and keyboard.

console=

Use the **console=** option to specify a device that you want to use as the primary console. For example, to use a console on the first serial port, use **console=ttyS0**. Use this option in conjunction with the **inst.text** option. You can use the **console=** option multiple times. If you do, the boot message is displayed on all specified consoles, but only the last one is used by the installation program. For example, if you specify **console=ttyS0 console=ttyS1**, the installation program uses **ttyS1**.

inst.lang=

Use the **inst.lang=** option to set the language that you want to use during the installation. The **locale -a | grep _** or **localectl list-locales | grep _** commands return a list of locales.

inst.singlelang

Use the **inst.singlelang** option to install in single language mode, which results in no available interactive options for the installation language and language support configuration. If a language is specified using the **inst.lang** boot option or the **lang** Kickstart command, then it is used. If no language is specified, the installation program defaults to **en_US.UTF-8**.

inst.geoloc=

Use the **inst.geoloc=** option to configure geolocation usage in the installation program. Geolocation is used to preset the language and time zone, and uses the following syntax: **inst.geoloc=value**. The **value** can be any of the following parameters:

Table D.6. Values for the inst.geoloc boot option

Value	Boot option format
Disable geolocation	inst.geoloc=0
Use the Fedora GeolIP API	inst.geoloc=provider_fedora_geoipl
Use the Hostip.info GeolIP API	inst.geoloc=provider_hostip

If you do not specify the **inst.geoloc=** option, the installation program uses **provider_fedora_geoipl**.

inst.keymap=

Use the **inst.keymap=** option to specify the keyboard layout that you want to use for the installation.

inst.cmdline

Use the **inst.cmdline** option to force the installation program to run in command-line mode. This mode does not allow any interaction, and you must specify all options in a Kickstart file or on the command line.

inst.graphical

Use the **inst.graphical** option to force the installation program to run in graphical mode. This mode is the default.

inst.text

Use the **inst.text** option to force the installation program to run in text mode instead of graphical mode.

inst.noninteractive

Use the **inst.noninteractive** boot option to run the installation program in a non-interactive mode. User interaction is not permitted in the non-interactive mode, and **inst.noninteractive** can be used with a graphical or text installation. When the **inst.noninteractive** option is used in text mode it behaves the same as the **inst.cmdline** option.

inst.resolution=

Use the **inst.resolution=** option to specify the screen resolution in graphical mode. The format is **NxM**, where *N* is the screen width and *M* is the screen height (in pixels). The lowest supported resolution is 1024x768.

inst.vnc=

Use the **inst.vnc=** option to run the graphical installation using VNC. You must use a VNC client application to interact with the installation program. When VNC sharing is enabled, multiple clients can connect. A system installed using VNC starts in text mode.

inst.vncpassword=

Use the **inst.vncpassword=** option to set a password on the VNC server that is used by the installation program.

inst.vncconnect=

Use the **inst.vncconnect=** option to connect to a listening VNC client at the given host location. For example **inst.vncconnect=<host>[:<port>]** The default port is 5900. This option can be used with **vncviewer -listen**.

inst.xdriver=

Use the **inst.xdriver=** option to specify the name of the X driver that you want to use both during installation and on the installed system.

inst.usefbx=

Use the **inst.usefbx** option to prompt the installation program to use the frame buffer X driver instead of a hardware-specific driver. This option is equivalent to **inst.xdriver=fbdev**.

modprobe.blacklist=

Use the **modprobe.blacklist=** option to blacklist or completely disable one or more drivers. Drivers (mods) that you disable using this option cannot load when the installation starts, and after the installation finishes, the installed system retains these settings. You can find a list of the blacklisted drivers in the **/etc/modprobe.d/** directory. Use a comma-separated list to disable multiple drivers. For example:

```
modprobe.blacklist=ahci,firewire_ohci
```

inst.xtimeout=

Use the **inst.xtimeout=** option to specify the timeout in seconds for starting X server.

inst.sshd

Use the **inst.sshd** option to start the **sshd** service during installation, so that you can connect to the system during the installation using SSH, and monitor the installation progress. For more information about SSH, see the **ssh(1)** man page. By default, the **sshd** option is automatically started only on the IBM Z architecture. On other architectures, **sshd** is not started unless you use the **inst.sshd** option.



NOTE

During installation, the root account has no password by default. You can set a root password during installation with the **sshpw** Kickstart command.

inst.kdump-addon=

Use the **inst.kdump-addon=** option to enable or disable the Kdump configuration screen (add-on) in the installation program. This screen is enabled by default; use **inst.kdump-addon=off** to disable it. Disabling the add-on disables the Kdump screens in both the graphical and text-based interface as well as the **%addon com_redhat_kdump** Kickstart command.

D.4. DEBUG BOOT OPTIONS

This section contains information about the options that you can use when debugging issues.

inst.rescue=

Use the **inst.rescue=** option to run the rescue environment. The option is useful for trying to diagnose and fix systems.

inst.updates=

Use the **inst.updates=** option to specify the location of the **updates.img** file that you want to apply during installation. There are a number of sources for the updates.

Table D.7. **inst.updates=** source updates

Source	Description	Example
Updates from a network	The easiest way to use inst.updates= is to specify the network location of updates.img . This does not require any modification to the installation tree. To use this method, edit the kernel command line to include inst.updates .	inst.updates=http://some.website.com/path/to/updates.img .
Updates from a disk image	You can save an updates.img on a floppy drive or a USB key. This can be done only with an ext2 filesystem type of updates.img . To save the contents of the image on your floppy drive, insert the floppy disc and run the command.	dd if=updates.img of=/dev/fd0 bs=72k count=20 . To use a USB key or flash media, replace /dev/fd0 with the device name of your USB key.
Updates from an installation tree	If you are using a CD, hard drive, HTTP, or FTP install, you can save the updates.img in the installation tree so that all installations can detect the .img file. Save the file in the images/ directory. The file name must be updates.img .	For NFS installs, there are two options: You can either save the image in the images/ directory, or in the RHupdates/ directory in the installation tree.

inst.loglevel=

Use the **inst.loglevel=** option to specify the minimum level of messages logged on a terminal. This

concerns only terminal logging; log files always contain messages of all levels. Possible values for this option from the lowest to highest level are: **debug**, **info**, **warning**, **error** and **critical**. The default value is **info**, which means that by default, the logging terminal displays messages ranging from **info** to **critical**.

inst.syslog=

When installation starts, the **inst.syslog=** option sends log messages to the **syslog** process on the specified host. The remote **syslog** process must be configured to accept incoming connections.

inst.virtiolog=

Use the **inst.virtiolog=** option to specify the virtio port (a character device at **/dev/virtio-ports/name**) that you want to use for forwarding logs. The default value is **org.fedoraproject.anaconda.log.0**; if this port is present, it is used.

inst.zram

The **inst.zram** option controls the usage of zRAM swap during installation. The option creates a compressed block device inside the system RAM and uses it for swap space instead of the hard drive. This allows the installation program to run with less available memory than is possible without compression, and it might also make the installation faster. By default, swap on zRAM is enabled on systems with 2 GiB or less RAM, and disabled on systems with more than 2 GiB of memory. You can use this option to change this behavior; on a system with more than 2 GiB RAM, use **inst.zram=1** to enable the feature, and on systems with 2 GiB or less memory, use **inst.zram=0** to disable the feature.

rd.live.ram

If the **rd.live.ram** option is specified, the **stage 2** image is copied into RAM. Using this option when the **stage 2** image is on an NFS server increases the minimum required memory by the size of the image by roughly 500 MiB.

inst.nokill

The **inst.nokill** option is a debugging option that prevents the installation program from rebooting when a fatal error occurs, or at the end of the installation process. Use the **inst.nokill** option to capture installation logs which would be lost upon reboot.

inst.noshell

Use **inst.noshell** option if you do not want a shell on terminal session 2 (tty2) during installation.

inst.notmux

Use **inst.notmux** option if you do not want to use tmux during installation. The output is generated without terminal control characters and is meant for non-interactive uses.

remotelog

You can use the **remotelog** option to send all of the logs to a remote **host:port** using a TCP connection. The connection is retired if there is no listener and the installation proceeds as normal.

D.5. STORAGE BOOT OPTIONS

inst.nodmraid

Use the **inst.nodmraid** option to disable **dmraid** support.

**WARNING**

Use this option with caution. If you have a disk that is incorrectly identified as part of a firmware RAID array, it might have some stale RAID metadata on it that must be removed using the appropriate tool, for example, **dmraid** or **wipefs**.

inst.nompath

Use the **inst.nompath** option to disable support for multipath devices. This option can be used for systems on which a false-positive is encountered which incorrectly identifies a normal block device as a multipath device. There is no other reason to use this option.

**WARNING**

Use this option with caution. You should not use this option with multipath hardware. Using this option to attempt to install to a single path of a multipath is not supported.

inst.gpt

The **inst.gpt** boot option forces the installation program to install partition information to a GUID Partition Table (GPT) instead of a Master Boot Record (MBR). This option is not valid on UEFI-based systems, unless they are in BIOS compatibility mode. Normally, BIOS-based systems and UEFI-based systems in BIOS compatibility mode attempt to use the MBR schema for storing partitioning information, unless the disk is 232 sectors in size or larger. Disk sectors are typically 512 bytes in size, meaning that this is usually equivalent to 2 TiB. Using the **inst.gpt** boot option changes this behavior, allowing a GPT to be written to smaller disks.

D.6. DEPRECATED BOOT OPTIONS

This section contains information about deprecated boot options. These options are still accepted by the installation program but they are deprecated and are scheduled to be removed in a future release of Red Hat Enterprise Linux.

method

The **method** option is an alias for **inst.repo**.

repo=nfsiso

The **repo=nfsiso:** option is the same as **inst.repo=nfs:**

dns

Use **nameserver** instead of **dns**. Note that nameserver does not accept comma-separated lists; use multiple nameserver options instead.

netmask, gateway, hostname

The **netmask**, **gateway**, and **hostname** options are provided as part of the **ip** option.

ip=bootif

A PXE-supplied **BOOTIF** option is used automatically, so there is no requirement to use **ip=bootif**.
ksdevice

Table D.8. Values for the **ksdevice** boot option

Value	Information
Not present	N/A
ksdevice=link	Ignored as this option is the same as the default behavior
ksdevice=bootif	Ignored as this option is the default if BOOTIF= is present
ksdevice=ibft	Replaced with ip=ibft . See ip for details
ksdevice=<MAC>	Replaced with BOOTIF=\${MAC/:/-}
ksdevice=<DEV>	Replaced with bootdev

D.7. REMOVED BOOT OPTIONS

This section contains the boot options that have been removed from Red Hat Enterprise Linux.



NOTE

dracut provides advanced boot options. For more information about **dracut**, see the **dracut.cmdline(7)** man page.

askmethod, **asknetwork**

initramfs is completely non-interactive, so the **askmethod** and **asknetwork** options have been removed. Instead, use **inst.repo** or specify the appropriate network options.

blacklist, **nofirewire**

The **modprobe** option handles blacklisting kernel modules; use **modprobe.blacklist=<mod1>, <mod2>**. You can blacklist the firewire module by using **modprobe.blacklist=firewire_ohci**.

inst.headless=

The **headless=** option specified that the system that is being installed to does not have any display hardware, and that the installation program is not required to look for any display hardware.

inst.decorated

The **inst.decorated** option was used to specify the graphical installation in a decorated window. By default, the window is not decorated, so it doesn't have a title bar, resize controls, and so on. This option was no longer required.

serial

Use the **console=ttyS0** option.

updates

Use the **inst.updates** option.

essid, wepkey, wpakey

Dracut does not support wireless networking.

ethtool

This option was no longer required.

gdb

This option was removed as there are many options available for debugging dracut-based **initramfs**.

inst.mediacheck

Use the **dracut option rd.live.check** option.

ks=floppy

Use the **inst.ks=hd:<device>** option.

display

For a remote display of the UI, use the **inst.vnc** option.

utf8

This option was no longer required as the default TERM setting behaves as expected.

noipv6

ipv6 is built into the kernel and cannot be removed by the installation program. You can disable ipv6 using **ipv6.disable=1**. This setting is used by the installed system.

upgradeany

This option was no longer required as the installation program no longer handles upgrades.

APPENDIX E. CHANGING A SUBSCRIPTION SERVICE

To manage the subscriptions, you can register a RHEL system with either Red Hat Subscription Management Server or Red Hat Satellite Server. If required, you can change the subscription service at a later point. To change the subscription service under which you are registered, unregister the system from the current service and then register it with a new service.

This section contains information about how to unregister your RHEL system from the Red Hat Subscription Management Server and Red Hat Satellite Server.

Prerequisites

You have registered your system with any one of the following:

- Red Hat Subscription Management Server
- Red Hat Satellite Server



NOTE

To receive the system updates, register your system with either of the management server.

E.1. UNREGISTERING FROM SUBSCRIPTION MANAGEMENT SERVER

This section contains information about how to unregister a RHEL system from Red Hat Subscription Management Server, using a command line and the Subscription Manager user interface.

E.1.1. Unregistering using command line

Use the **unregister** command to unregister a RHEL system from Red Hat Subscription Management Server.

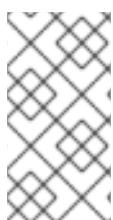
Procedure

1. Run the unregister command as a root user, without any additional parameters.

```
# subscription-manager unregister
```

2. When prompted, provide a root password.

The system is unregistered from the Subscription Management Server, and the status 'The system is currently not registered' is displayed with the **Register** button enabled.



NOTE

To continue uninterrupted services, re-register the system with either of the management services. If you do not register the system with a management service, you may fail to receive the system updates. For more information about registering a system, see [Registering your system using the command line](#)

For more information about Red Hat Subscription Management server, see the [Using and Configuring Red Hat Subscription Manager](#) document.

E.1.2. Unregistering using Subscription Manager user interface

This section contains information about how to unregister a RHEL system from Red Hat Subscription Management Server, using Subscription Manager user interface.

Procedure

1. Log in to your system.
2. From the top left-hand side of the window, click **Activities**.
3. From the menu options, click the **Show Applications** icon.
4. Click the **Red Hat Subscription Manager** icon, or enter **Red Hat Subscription Manager** in the search.
5. Enter your administrator password in the **Authentication Required** dialog box. The **Subscriptions** window appears and displays the current status of Subscriptions, System Purpose, and installed products. Unregistered products display a red X.



NOTE

Authentication is required to perform privileged tasks on the system.

6. Click the **Unregister** button.

The system is unregistered from the Subscription Management Server, and the status 'The system is currently not registered' is displayed with the **Register** button enabled.



NOTE

To continue uninterrupted services, re-register the system with either of the management services. If you do not register the system with a management service, you may fail to receive the system updates. For more information about registering a system, see [Registering your system using the Subscription Manager User Interface](#)

For more information about Red Hat Subscription Management server, see the [Using and Configuring Red Hat Subscription Manager](#) document.

E.2. UNREGISTERING FROM SATELLITE SERVER

To unregister a Red Hat Enterprise Linux system from Satellite Server, remove the system from Satellite Server.

For more information, see [Removing a Host from Red Hat Satellite](#) in the *Managing Hosts* guide from Satellite Server documentation.

APPENDIX F. ISCSI DISKS IN INSTALLATION PROGRAM

The Red Hat Enterprise Linux installer can discover and log in to iSCSI disks in two ways:

- When the installer starts, it checks if the BIOS or add-on boot ROMs of the system support iSCSI Boot Firmware Table (iBFT), a BIOS extension for systems that can boot from iSCSI. If the BIOS supports iBFT, the installer reads the iSCSI target information for the configured boot disk from the BIOS and logs in to this target, making it available as an installation target.



IMPORTANT

To connect automatically to an iSCSI target, activate a network device for accessing the target. To do so, use **ip=ibft** boot option. For more information, see [Network boot options](#).

- You can discover and add iSCSI targets manually in the installer’s graphical user interface. For more information, see [Configuring storage devices](#).



IMPORTANT

You cannot place the **/boot** partition on iSCSI targets that you have manually added using this method – an iSCSI target containing a **/boot** partition must be configured for use with iBFT. However, in instances where the installed system is expected to boot from iSCSI with iBFT configuration provided by a method other than firmware iBFT, for example using iPXE, you can remove the **/boot** partition restriction using the **inst.nonibftiscsiboot** installer boot option.

While the installer uses **iscsiadm** to find and log into iSCSI targets, **iscsiadm** automatically stores any information about these targets in the **iscsiadm** iSCSI database. The installer then copies this database to the installed system and marks any iSCSI targets that are not used for root partition, so that the system automatically logs in to them when it starts. If the root partition is placed on an iSCSI target, initrd logs into this target and the installer does not include this target in start up scripts to avoid multiple attempts to log into the same target.

CHAPTER 7. COMPOSING A CUSTOMIZED RHEL SYSTEM IMAGE

CHAPTER 8. IMAGE BUILDER DESCRIPTION

8.1. INTRODUCTION TO IMAGE BUILDER

You can use Image Builder to create customized system images of Red Hat Enterprise Linux, including system images prepared for deployment on cloud platforms. Image Builder automatically handles details of setup for each output type and is thus easier to use and faster to work with than manual methods of image creation. You can access Image Builder functionality through a command-line interface in the **composer-cli** tool, or a graphical user interface in the RHEL 8 web console.

Image Builder runs as a system service **lorax-composer**. You can interact with this service through two interfaces:

- CLI tool **composer-cli** for running commands in the terminal. Prefer this method.
- GUI plugin for the RHEL 8 web console.

8.2. IMAGE BUILDER TERMINOLOGY

Blueprint

Blueprints define customized system images by listing packages and customizations that will be part of the system. Blueprints can be edited and they are versioned. When a system image is created from a blueprint, the image is associated with the blueprint in the Image Builder interface of the RHEL 8 web console.

Blueprints are presented to the user as plain text in the Tom's Obvious, Minimal Language (TOML) format.

Compose

Composes are individual builds of a system image, based on a particular version of a particular blueprint. Compose as a term refers to the system image, the logs from its creation, inputs, metadata, and the process itself.

Customizations

Customizations are specifications for the system, which are not packages. This includes users, groups, and SSH keys.

8.3. IMAGE BUILDER OUTPUT FORMATS

Image Builder can create images in multiple output formats shown in the following table.

Table 8.1. Image Builder output formats

Description	CLI name	file extension
QEMU QCOW2 Image	qcow2	.qcow2
Ext4 File System Image	ext4-filesystem	.img
Raw Partitioned Disk Image	partitioned-disk	.img
Live Bootable ISO	live-iso	.iso

Description	CLI name	file extension
TAR Archive	tar	.tar
Amazon Machine Image Disk	ami	.ami
Azure Disk Image	vhd	.vhd
VMware Virtual Machine Disk	vmdk	.vmdk
Openstack	openstack	.qcow2

8.4. IMAGE BUILDER SYSTEM REQUIREMENTS

The **lorax** tool underlying Image Builder performs a number of potentially insecure and unsafe actions while creating the system images. For this reason, use a virtual machine to run Image Builder.

The environment where Image Builder runs, for example the virtual machine, must meet requirements listed in the following table.

Table 8.2. Image Builder system requirements

Parameter	Minimal Required Value
System type	A dedicated virtual machine
Processor	2 cores
Memory	4 GiB
Disk space	20 GiB
Access privileges	Administrator level (root)
Network	Connectivity to Internet



NOTE

There is no support for creating images on virtual machine directly installed on UEFI systems.

CHAPTER 9. INSTALLING IMAGE BUILDER

Image Builder is a tool for creating custom system images. Before using Image Builder, you must install Image Builder in a virtual machine.

9.1. IMAGE BUILDER SYSTEM REQUIREMENTS

The **lorax** tool underlying Image Builder performs a number of potentially insecure and unsafe actions while creating the system images. For this reason, use a virtual machine to run Image Builder.

The environment where Image Builder runs, for example the virtual machine, must meet requirements listed in the following table.

Table 9.1. Image Builder system requirements

Parameter	Minimal Required Value
System type	A dedicated virtual machine
Processor	2 cores
Memory	4 GiB
Disk space	20 GiB
Access privileges	Administrator level (root)
Network	Connectivity to Internet



NOTE

There is no support for creating images on virtual machine directly installed on UEFI systems.

9.2. INSTALLING IMAGE BUILDER IN A VIRTUAL MACHINE

To install Image Builder on a dedicated virtual machine, follow these steps:

Prerequisites

- Connect to the virtual machine.
- The virtual machine for Image Builder must be installed, subscribed, and running.

Procedure

1. Install the Image Builder and other necessary packages on the virtual machine:
 - lorax-composer
 - composer-cli

- cockpit-composer
- bash-completion

```
# yum install lorax-composer composer-cli cockpit-composer bash-completion
```

The web console is installed as a dependency of the *cockpit-composer* package.

2. Enable Image Builder to start after each reboot:

```
# systemctl enable lorax-composer.socket
# systemctl enable cockpit.socket
```

The **lorax-composer** and **cockpit** services start automatically on first access.

3. Configure the system firewall to allow access to the web console:

```
# firewall-cmd --add-service=cockpit && firewall-cmd --add-service=cockpit --permanent
```

4. Load the shell configuration script so that the autocomplete feature for the **composer-cli** command starts working immediately without reboot:

```
$ source /etc/bash_completion.d/composer-cli
```

CHAPTER 10. CREATING SYSTEM IMAGES WITH IMAGE BUILDER COMMAND-LINE INTERFACE

Image Builder is a tool for creating custom system images. To control Image Builder and create your custom system images, use the command-line interface which is currently the preferred method to use Image Builder.

10.1. IMAGE BUILDER COMMAND-LINE INTERFACE

Image Builder command-line interface is currently the preferred method to use Image Builder. It offers more functionality than the Web console interface. To use this interface, run the **composer-cli** command with suitable options and subcommands.

The workflow for the command-line interface can be summarized as follows:

1. Export (save) the blueprint definition to a plain text file
2. Edit this file in a text editor
3. Import (push) the blueprint text file back into Image Builder
4. Run a compose to build an image from the blueprint
5. Export the image file to download it

Apart from the basic subcommands to achieve this procedure, the **composer-cli** command offers many subcommands to examine the state of configured blueprints and composes.

To run the **composer-cli** command as non-root, user must be in the **weldr** or **root** groups.

10.2. CREATING AN IMAGE BUILDER BLUEPRINT WITH COMMAND-LINE INTERFACE

This procedure describes how to create a new Image Builder blueprint using the command-line interface.

Procedure

1. Create a plain text file with the following contents:

```
name = "BLUEPRINT-NAME"
description = "LONG FORM DESCRIPTION TEXT"
version = "0.0.1"
modules = []
groups = []
```

Replace *BLUEPRINT-NAME* and *LONG FORM DESCRIPTION TEXT* with a name and description for your blueprint.

Replace *0.0.1* with a version number according to the [Semantic Versioning](#) scheme.

2. For every package that you want to be included in the blueprint, add the following lines to the file:

```
[[packages]]
name = "package-name"
version = "package-version"
```

Replace *package-name* with name of the package, such as `httpd`, `gdb-doc`, or `coreutils`.

Replace *package-version* with a version to use. This field supports `dnf` version specifications:

- For a specific version, use the exact version number such as `8.30`.
- For latest available version, use the asterisk `*`.
- For a latest minor version, use format such as `8.*`.

3. Blueprints can be customized in a number of ways. For this example, Simultaneous Multi Threading (SMT) can be disabled by performing the steps below. For additional customizations available, please see [Section 10.7, "Supported Image Customizations"](#).

```
[customizations.kernel]
append = "nosmt=force"
```

4. Save the file as *BLUEPRINT-NAME.toml* and close the text editor.
5. Push (import) the blueprint:

```
# composer-cli blueprints push BLUEPRINT-NAME.toml
```

Replace *BLUEPRINT-NAME* with the value you used in previous steps.

6. To verify that the blueprint has been pushed and exists, list the existing blueprints:

```
# composer-cli blueprints list
```

7. Check whether the components and versions listed in the blueprint and their dependencies are valid:

```
# composer-cli blueprints depsolve BLUEPRINT-NAME
```

10.3. EDITING AN IMAGE BUILDER BLUEPRINT WITH COMMAND-LINE INTERFACE

This procedure describes how to edit an existing Image Builder blueprint in the command-line interface.

Procedure

1. Save (export) the blueprint to a local text file:

```
# composer-cli blueprints save BLUEPRINT-NAME
```

2. Edit the *BLUEPRINT-NAME.toml* file with a text editor of your choice and make your changes.
3. Before finishing with the edits, make sure the file is a valid blueprint:

- a. Remove this line, if present:

```
packages = []
```

- b. Increase the version number. Remember that Image Builder blueprint versions must use the [Semantic Versioning](#) scheme. Note also that if you do not change the version, the `patch` component of version is increased automatically.
- c. Check if the contents are valid TOML specifications. See the [TOML documentation](#) for more information.



NOTE

TOML documentation is a community product and is not supported by Red Hat. You can report any issues with the tool at <https://github.com/toml-lang/toml/issues>

4. Save the file and close the editor.
5. Push (import) the blueprint back into Image Builder:

```
# composer-cli blueprints push BLUEPRINT-NAME.toml
```

Note that you must supply the file name including the `.toml` extension, while in other commands you use only the name of the blueprint.

6. To verify that the contents uploaded to Image Builder match your edits, list the contents of blueprint:

```
# composer-cli blueprints show BLUEPRINT-NAME
```

7. Check whether the components and versions listed in the blueprint and their dependencies are valid:

```
# composer-cli blueprints depsolve BLUEPRINT-NAME
```

10.4. CREATING A SYSTEM IMAGE WITH IMAGE BUILDER IN THE COMMAND-LINE INTERFACE

This procedure shows how to build a custom image using the Image Builder command-line interface.

Prerequisites

- You have a blueprint prepared for the image.

Procedure

1. Start the compose:

```
# composer-cli compose start BLUEPRINT-NAME IMAGE-TYPE
```

Replace *BLUEPRINT-NAME* with name of the blueprint, and *IMAGE-TYPE* with the type of image. For possible values, see output of the **composer-cli compose types** command.

The compose process starts in the background and the UUID of the compose is shown.

2. Wait until the compose is finished. Please, notice that this may take several minutes.
To check the status of the compose:

```
# composer-cli compose status
```

A finished compose shows a status value **FINISHED**. Identify the compose in the list by its UUID.

3. Once the compose is finished, download the resulting image file:

```
# composer-cli compose image UUID
```

Replace *UUID* with the UUID value shown in the previous steps.

Alternatively, you can access the image file directly under the path **/var/lib/lorax/composer/results/*UUID*/**.

You can also download the logs using the **composer-cli compose logs *UUID*** command, or the metadata using the **composer-cli compose metadata *UUID*** command.

10.5. BASIC IMAGE BUILDER COMMAND-LINE COMMANDS

The Image Builder command-line interface offers the following subcommands.

Blueprint manipulation

List all available blueprints

```
# composer-cli blueprints list
```

Show a blueprint contents in the TOML format

```
# composer-cli blueprints show BLUEPRINT-NAME
```

Save (export) blueprint contents in the TOML format into a file *BLUEPRINT-NAME.toml*

```
# composer-cli blueprints save BLUEPRINT-NAME
```

Remove a blueprint

```
# composer-cli blueprints delete BLUEPRINT-NAME
```

Push (import) a blueprint file in the TOML format into Image Builder

```
# composer-cli blueprints push BLUEPRINT-NAME
```

Composing images from blueprints

Start a compose

```
# composer-cli compose start BLUEPRINT COMPOSE-TYPE
```

Replace *BLUEPRINT* with name of the blueprint to build and *COMPOSE-TYPE* with the output image type.

List all composes

```
# composer-cli compose list
```

List all composes and their status

```
# composer-cli compose status
```

Cancel a running compose

```
# composer-cli compose cancel COMPOSE-UUID
```

Delete a finished compose

```
# composer-cli compose delete COMPOSE-UUID
```

Show detailed information about a compose

```
# composer-cli compose info COMPOSE-UUID
```

Download image file of a compose

```
# composer-cli compose image COMPOSE-UUID
```

Related information

- The *composer-cli(1)* manual page provides a full list of the available subcommands and options:

```
$ man composer-cli
```

- The **composer-cli** command provides help on the subcommands and options:

```
# composer-cli help
```

10.6. IMAGE BUILDER BLUEPRINT FORMAT

Image Builder blueprints are presented to the user as plain text in the Tom's Obvious, Minimal Language (TOML) format.

The elements of a typical blueprint file include:

The blueprint metadata

■

```
name = "BLUEPRINT-NAME"
description = "LONG FORM DESCRIPTION TEXT"
version = "VERSION"
```

Replace *BLUEPRINT-NAME* and *LONG FORM DESCRIPTION TEXT* with a name and description for your blueprint.

Replace *VERSION* with a version number according to the [Semantic Versioning](#) scheme.

This part is present only once for the whole blueprint file.

The entry *modules* describe the package names and matching version glob to be installed into the image.

The entry *group* describes a group of packages to be installed into the image. Groups categorize their packages in:

- Mandatory
 - Default
 - Optional
- Blueprints installs the mandatory packages. There is no mechanism for selecting optional packages.

Groups to include in the image

```
[[groups]]
name = "group-name"
```

Replace *group-name* with the name of the group, such as `anaconda-tools`, `widget`, `wheel` or `users`.

Packages to include in the image

```
[[packages]]
name = "package-name"
version = "package-version"
```

Replace *package-name* with the name of the package, such as `httpd`, `gdb-doc`, or `coreutils`.

Replace *package-version* with a version to use. This field supports `dnf` version specifications:

- For a specific version, use the exact version number such as `8.30`.
- For latest available version, use the asterisk `*`.
- For a latest minor version, use format such as `8.*`.

Repeat this block for every package to include.

10.7. SUPPORTED IMAGE CUSTOMIZATIONS

A number of image customizations are supported at this time within blueprints. In order to make use of these options, they must be initially configured in the blueprint and imported (pushed) to Image Builder.



NOTE

These customizations are not currently supported within the accompanying cockpit-composer GUI.

Set the image hostname

```
[customizations]
hostname = "baseimage"
```

User specifications for the resulting system image

```
[[customizations.user]]
name = "USER-NAME"
description = "USER-DESCRIPTION"
password = "PASSWORD-HASH"
key = "PUBLIC-SSH-KEY"
home = "/home/USER-NAME/"
shell = "/usr/bin/bash"
groups = ["users", "wheel"]
uid = NUMBER
gid = NUMBER
```



IMPORTANT

To generate the hash, you must install **python3** on your system. The following command will install the **python3** package.

```
# yum install python3
```

Replace *PASSWORD-HASH* with the actual password hash. To generate the hash, use a command such as:

```
$ python3 -c 'import crypt,getpass;pw=getpass.getpass();print(crypt.crypt(pw) if
(pw==getpass.getpass("Confirm: ")) else exit())'
```

Replace *PUBLIC-SSH-KEY* with the actual public key.

Replace the other placeholders with suitable values.

Leave out any of the lines as needed, only the user name is required.

Repeat this block for every user to include.

Group specifications for the resulting system image

```
[[customizations.group]]
name = "GROUP-NAME"
gid = NUMBER
```

Repeat this block for every group to include.

Set an existing users ssh key

```
[[customizations.sshkey]]  
user = "root"  
key = "PUBLIC-SSH-KEY"
```



NOTE

This option is only applicable for existing users. To create a user and set an ssh key, use the [User specifications for the resulting system image](#) customization.

Append a kernel boot parameter option to the defaults

```
[customizations.kernel]  
append = "KERNEL-OPTION"
```

Set the image host name

```
[customizations]  
hostname = "BASE-IMAGE"
```

Add a group for the resulting system image

```
[[customizations.group]]  
name = "USER-NAME"  
gid = "NUMBER"
```

Only the name is required and GID is optional.

Set the timezone and the *Network Time Protocol* (NTP) servers for the resulting system image

```
[customizations.timezone]  
timezone = "TIMEZONE"  
ntpservers = "NTP_SERVER"
```

If you do not set a timezone, the system uses *Universal Time, Coordinated (UTC)* as default. Setting NTP servers is optional.

Set the locale settings for the resulting system image

```
[customizations.locale]  
language = "[LANGUAGE]"  
keyboard = "KEYBOARD"
```

Setting both language and keyboard options is mandatory. You can add multiple languages. The first language you add will be the primary language and the other languages will be secondary.

Set the firewall for the resulting system image

```
[customizations.firewall]  
port = "[PORTS]"
```

You can use the numeric ports, or theirs names from the **/etc/services** file to enable or disable lists.

Set which services to enable during the boot time

```
[customizations.services]
enabled = "[SERVICES]"
disabled = "[SERVICES]"
```

You can control which services to enable during the boot time. Some image types already have services enabled or disabled so that the image works correctly and this setup cannot be overridden.

Add files from a git repository to your blueprint

```
[[repos.git]]
rpmname = "_RPM-NAME"
rpmversion = "[RPM-VERSION]"
rpmrelease = "[RPM-RELEASE]"
summary = "[RPM-SUMMARY]"
repo = "[REPO-URL]"
ref = "[GIT-REF]"
destination = "[SERVICES]"
```

You can use entries to add files from a git repository to the created image.

For example, to create an RPM package named **server-config-1.0-1.noarch.rpm**, add the following information to your blueprint:

Replace *_RPM-NAME* with the name of the RPM package to create. This is also the prefix name in the resulting tar archive.

Replace *RPM-VERSION* with the version of the RPM package, for example, "1.0.0".

Replace *RPM-RELEASE* with the version of the RPM package release, for example, "1".

Replace *RPM-SUMMARY* with the summary string for the RPM package.

Replace *REPO-URL* with the URL of the get repository to clone and create the archive from it.

Replace *GIT-REF* with the git reference to check out, for example, **origin/branch-name**, **git tag**, or **git commit hash**.

Replace *SERVICES* with the path to install the directory of the git repository when installing the RPM package.

As a consequence, the git repository you provided is cloned, the specified git reference is checked out and an RPM package is created to install the files to a destination path, for example, **/opt/server/**. The RPM includes a summary with the details of the repository and reference used to create it. The RPM package is also included in the image build metadata.



NOTE

Each time a build starts, it clones the repository. If you refer to a repository with a large amount of history, it might take a while to clone and use a significant amount of disk space. Also, the clone is temporary and is removed once the RPM package is created.

10.8. INSTALLED PACKAGES

When you create a system image using Image Builder, by default, the system installs a set of base packages. The base list of packages are the members of the **comps core** group.

Table 10.1. Default packages to support image type creation

Image type	Default Packages
alibaba.ks	kernel, selinux-policy-targeted, cloud-init
ami.ks	kernel, selinux-policy-targeted, chrony, cloud-init
ext4-filesystem.ks	policycoreutils, selinux-policy-targeted, kernel
google.ks	kernel, selinux-policy-targeted
live-iso.ks	isomd5sum, kernel, dracut-config-generic, dracut-live, system-logos, selinux-policy-targeted
openstack.ks	kernel, selinux-policy-targeted
partitioned-disk.ks	kernel, selinux-policy-targeted
qcow2.ks	kernel, selinux-policy-targeted
tar.ks	policycoreutils, selinux-policy-targeted
vhd.ks	kernel, selinux-policy-targeted, chrony, WALinuxAgent, python3, net-tools, cloud-init, cloud-utils-growpart, gdisk
vmdk.ks	kernel, selinux-policy-targeted, chrony, open-vm-tools



NOTE

When you add additional components to your blueprint, you must make sure that the packages in the components you added do not conflict with any other package components, otherwise the system fails to solve dependencies. As a consequence, you are not able to create your customized image.

Additional resources

- For more information about the available image types, see [Image Builder output formats](#).

10.9. ENABLED SERVICES

When you configure the custom image, the services enabled are the defaults services for the RHEL8 release you are running lorax-composer from, additionally the services enabled for specific image types.

For example, the **.ami** image type enables the services **sshd**, **chronyd** and **cloud-init** and without these services, the custom image does not boot.

Table 10.2. Enabled services to support image type creation

Image type	Enabled Services
alibaba.ks	sshd,cloud-init
ami.ks	No default service
ext4-filesystem.ks	No default service
google.ks	No default service
live-iso.ks	NetworkManager
openstack.ks	sshd, cloud-init, cloud-init-local, cloud-config, cloud-final
partitioned-disk.ks	No default service
qcow2.ks	No default service
tar.ks	No default service
vhd.ks	sshd, chronyd, waagent, cloud-init, cloud-init-local, cloud-config, cloud-final
vmdk.ks	sshd, chronyd, vmtoolsd

Note: You can customize which services to enable during the system boot. However, for image types with services enabled by default, the customization does not override this feature.

Additional resources

- For more information about service customization, see [Section 10.7, “Supported Image Customizations”](#)

10.10. DISKS AND PARTITIONS CONFIGURATION USING IMAGE BUILDER

Image Builder does not allow disks to be partitioned. The output types that have a partitioned disk will have a single partition and additionally any platform-specific partitions that are required to boot the system image. For example, **qcow2** image type has a single root partition, and possibly a platform specific boot partition - like **PReP** for **PPC64** system - that the image requires to boot.

CHAPTER 11. CREATING SYSTEM IMAGES WITH IMAGE BUILDER WEB CONSOLE INTERFACE

Image Builder is a tool for creating custom system images. To control Image Builder and create your custom system images, you can use the web console interface. Note that the command line interface is the currently preferred alternative, because it offers more features.

11.1. ACCESSING IMAGE BUILDER GUI IN THE RHEL 8 WEB CONSOLE

The **cockpit-composer** plugin for the RHEL 8 web console enables users to manage Image Builder blueprints and composes with a graphical interface. Note that the preferred method for controlling Image Builder is at the moment using the command-line interface.

Prerequisites

- You must have root access to the system.

Procedure

1. Open <https://localhost:9090/> in a web browser on the system where Image Builder is installed. For more information how to remotely access Image Builder, see [managing systems using the RHEL 8 web console](#).
2. Log into the web console with credentials for an user account with sufficient privileges on the system.
3. To display the Image Builder controls, click the **Image Builder** icon, which is in the upper-left corner of the window.
The Image Builder view opens, listing existing blueprints.

11.2. CREATING AN IMAGE BUILDER BLUEPRINT IN THE WEB CONSOLE INTERFACE

To describe the customized system image, create a blueprint first.

Prerequisites

- You have opened the Image Builder interface of the RHEL 8 web console in a browser.

Procedure

1. Click **Create Blueprint** in the top right corner.
A pop-up appears with fields for the blueprint name and description.
2. Fill in the name of the blueprint, its description, then click **Create**.
The screen changes to blueprint editing mode.
3. Add components that you want to include in the system image:
 1. On the left, enter all or part of the component name in the **Available Components** field and press **Enter**.
The search is added to the list of filters under the text entry field, and the list of components below is reduced to these that match the search.

If the list of components is too long, add further search terms in the same way.

2. The list of components is paged. To move to other result pages, use the arrows and entry field above the component list.
3. Click on name of the component you intend to use to display its details. The right pane fills with details of the components, such as its version and dependencies.
4. Select the version you want to use in the **Component Options** box, with the **Version Release** dropdown.
5. Click **Add** in the top left.
6. If you added a component by mistake, remove it by clicking the ... button at the far right of its entry in the right pane, and select **Remove** in the menu.



NOTE

If you do not intend to select version for some components, you can skip the component details screen and version selection by clicking the + buttons on the right side of the component list.

4. To save the blueprint, click **Commit** in the top right. A dialog with a summary of the changes pops up. Click **Commit**.
A small pop-up on the right informs you of the saving progress and then the result.
5. To exit the editing screen, click **Back to Blueprints** in the top left.
The Image Builder view opens, listing existing blueprints.

11.3. EDITING AN IMAGE BUILDER BLUEPRINT IN THE WEB CONSOLE INTERFACE

To change the specifications for a custom system image, edit the corresponding blueprint.

Prerequisites

- You have opened the Image Builder interface of the RHEL 8 web console in a browser.
- A blueprint exists.

Procedure

1. Locate the blueprint that you want to edit by entering its name or a part of it into the search box at top left, and press **Enter**.
The search is added to the list of filters under the text entry field, and the list of blueprints below is reduced to these that match the search.
If the list of blueprints is too long, add further search terms in the same way.
2. On the right side of the blueprint, press the **Edit Blueprint** button that belongs to the blueprint.
The view changes to the blueprint editing screen.
3. Remove unwanted components by clicking their ... button at the far right of its entry in the right pane, and select **Remove** in the menu.

4. Change version of existing components:

- On the Blueprint Components search field, enter component name or a part of it into the field under the heading **Blueprint Components** and press **Enter**.

The search is added to the list of filters under the text entry field, and the list of components below is reduced to these that match the search.

If the list of components is too long, add further search terms in the same way.

- Click the **...** button at the far right of the component entry, and select **View** in the menu. A component details screen opens in the right pane.

- Select the desired version in the **Version Release** drop-down menu and click **Apply Change** in top right.

The change is saved and the right pane returns to listing the blueprint components.

5. Add new components:

- On the left, enter component name or a part of it into the field under the heading **Available Components** and press **Enter**.

The search is added to the list of filters under the text entry field, and the list of components below is reduced to these that match the search.

If the list of components is too long, add further search terms in the same way.

- The list of components is paged. To move to other result pages, use the arrows and entry field above the component list.

- Click on name of the component you intend to use to display its details. The right pane fills with details of the components, such as its version and dependencies.

- Select the version you want to use in the **Component Options** box, with the **Version Release** drop-down menu.

- Click **Add** in the top right.

- If you added a component by mistake, remove it by clicking the **...** button at the far right of its entry in the right pane, and select **Remove** in the menu.



NOTE

If you do not intend to select version for some components, you can skip the component details screen and version selection by clicking the **+** buttons on the right side of the component list.

1. Commit a new version of the blueprint with your changes:

- Click the **Commit** button in top right.

A pop-up window with a summary of your changes appears.

- Review your changes and confirm them by clicking **Commit**.

A small pop-up on the right informs you of the saving progress and then result. A new version of the blueprint is created.

- In the top left, click **Back to Blueprints** to exit the editing screen.

The Image Builder view opens, listing existing blueprints.

11.4. ADDING USERS AND GROUPS TO AN IMAGE BUILDER BLUEPRINT IN THE WEB CONSOLE INTERFACE

Adding customizations such as users and groups to blueprints in the web console interface is currently not possible. To work around this limitation, use the **Terminal** tab in web console to use the command-line interface (CLI) workflow.

Prerequisites

- A blueprint must exist.
- A CLI text editor such as **vim**, **nano**, or **emacs** must be installed. To install them:

```
# yum install editor-name
```

Procedure

1. Find out the name of the blueprint: Open the Image Builder (**Image builder**) tab on the left in the RHEL 8 web console to see the name of the blueprint.
2. Navigate to the CLI in web console: Open the system administration tab on the left, then select the last item **Terminal** from the list on the left.
3. Enter the super-user (root) mode:

```
$ sudo bash
```

Provide your credentials when asked. Note that the terminal does not reuse your credentials you entered when logging into the web console.

A new shell with root privileges starts in your home directory.

4. Export the blueprint to a file:

```
# composer-cli blueprints save BLUEPRINT-NAME
```

5. Edit the file *BLUEPRINT-NAME.toml* with a CLI text editor of your choice and add the users and groups.



IMPORTANT

RHEL 8 web console does not have any built-in feature to edit text files on the system, so the use of a CLI text editor is required for this step.

- a. For every user to be added, add this block to the file:

```
[[customizations.user]]
name = "USER-NAME"
description = "USER-DESCRIPTION"
password = "PASSWORD-HASH"
key = "ssh-rsa (...) key-name"
home = "/home/USER-NAME/"
shell = "/usr/bin/bash"
```

```
groups = ["users", "wheel"]
uid = NUMBER
gid = NUMBER
```

Replace *PASSWORD-HASH* with the actual password hash. To generate the hash, use a command such as this:

```
$ python3 -c 'import crypt, getpass; pw=getpass.getpass(); print(crypt.crypt(pw) if
(pw==getpass.getpass("Confirm: ")) else exit())'
```

Replace *ssh-rsa (...)* *key-name* with the actual public key.

Replace the other placeholders with suitable values.

Leave out any of the lines as needed, only the user name is required.

- For every user group to be added, add this block to the file:

```
[[customizations.group]]
name = "GROUP-NAME"
gid = NUMBER
```

- Increase the version number.
- Save the file and close the editor.

- Import the blueprint back into Image Builder:

```
# composer-cli blueprints push BLUEPRINT-NAME.toml
```

Note that you must supply the file name including the **.toml** extension, while in other commands you use only the name of the blueprint.

- To verify that the contents uploaded to Image Builder match your edits, list the contents of blueprint:

```
# composer-cli blueprints show BLUEPRINT-NAME
```

Check if the version matches what you put in the file and if your customizations are present.



IMPORTANT

The Image Builder plugin for RHEL 8 web console does not show any information that could be used to verify that the changes have been applied, unless you edited also the packages included in the blueprint.

- Exit the privileged shell:

```
# exit
```

- Open the Image Builder (**Image builder**) tab on the left and refresh the page, in all browsers and all tabs where it was opened.

This prevents state cached in the loaded page from accidentally reverting your changes.

11.5. CREATING A SYSTEM IMAGE WITH IMAGE BUILDER IN THE WEB CONSOLE INTERFACE

The following steps below describe creating a system image.

Prerequisites

- You have opened the Image Builder interface of the RHEL 8 web console in a browser.
- A blueprint exists.

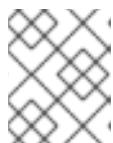
Procedure

1. Locate the blueprint that you want to build an image by entering its name or a part of it into the search box at top left, and press **Enter**.

The search is added to the list of filters under the text entry field, and the list of blueprints below is reduced to these that match the search.

If the list of blueprints is too long, add further search terms in the same way.

2. On the right side of the blueprint, press the **Create Image** button that belongs to the blueprint. A pop-up window appears.
3. Select the image type and press **Create**. A small pop-up in the top right informs you that the image creation has been added to the queue.
4. Click the name of the blueprint. A screen with details of the blueprint opens.
5. Click the **Images** tab to switch to it. The image that is being created is listed with the status **In Progress**.



NOTE

Image creation takes a longer time, measured in minutes. There is no indication of progress while the image is created.

To abort image creation, press its **Stop** button on the right.

6. Once the image is successfully created, the **Stop** button is replaced by a **Download** button. Click this button to download the image to your system.

11.6. ADDING A SOURCE TO A BLUEPRINT

The sources defined in Image Builder provide the contents that you can add to blueprints. These sources are global and therefore available to all blueprints. The System sources are repositories that are set up locally on your computer and cannot be removed from Image Builder. You can add additional custom sources and thus be able to access other contents than the System sources available on your system.

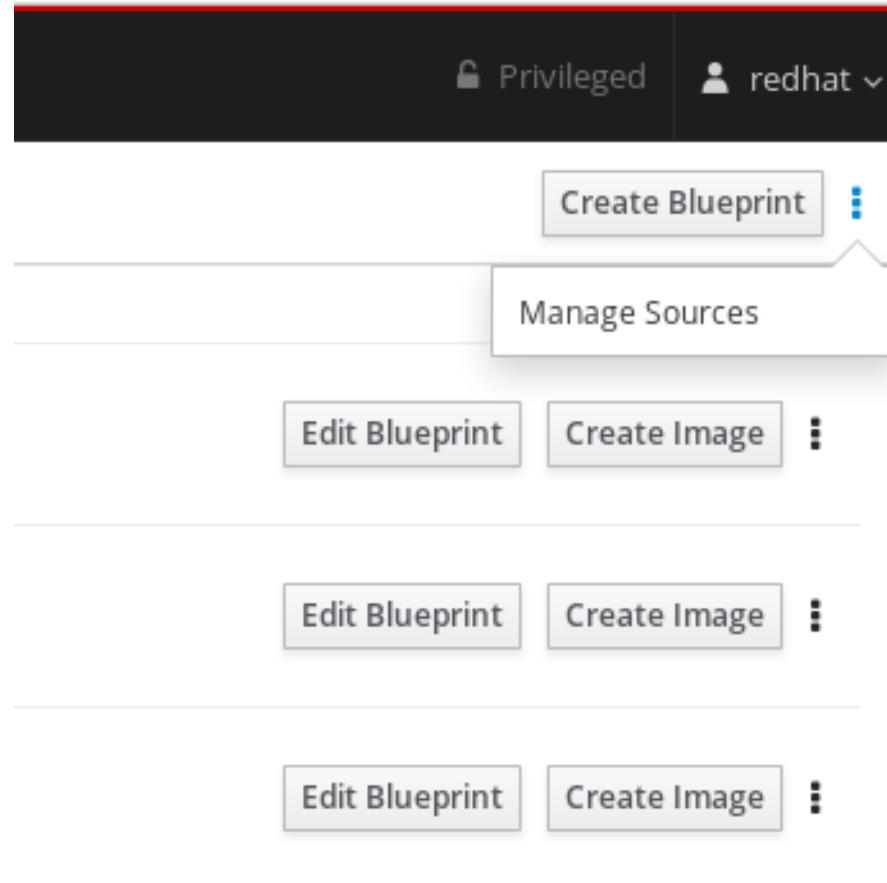
The following steps describe how to add a Source to your local system.

Prerequisites

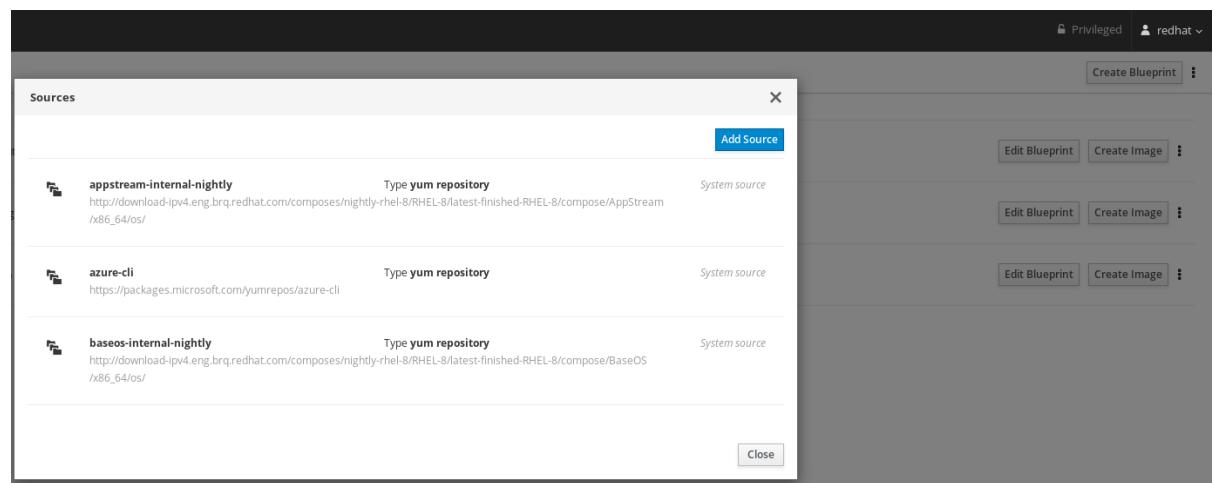
- You have opened the Image Builder interface of the RHEL 8 web console in a browser.

Procedure

1. Click the **Manage Sources** button in the top right corner.



A pop-up window appears with the available sources, their names and descriptions.



2. On the right side of the pop-up window, click the **Add Source** button.
3. Add the desired **Source name**, the **Source path**, and the **Source Type**. The **Security** field is optional.

The fields marked with * are required.

* Name:

* Source path:

* Type:

Security: Check SSL certificate
 Check GPG key

- Click **Add Source**. The screen shows the available sources window and list the source you have added.

As a result, the new System source is available and ready to be used or edited.

11.7. CREATING A USER ACCOUNT FOR A BLUEPRINT

The images created by Image Builder have the root account locked and no other accounts included. Such configuration is provided in order to ensure that you cannot accidentally build and deploy an image without a password. Image Builder enables you to create a user account with password for a blueprint so that you can log in to the image created from the blueprint.

Prerequisites

- You have opened the Image Builder interface of the RHEL 8 web console in a browser.
- You have an existing blueprint.

Procedure

- Locate the blueprint that you want to create a user account for by entering its name or a part of it into the search box at the top left, and press **Enter**.
The search is added to the list of filters under the text entry field, and the list of blueprints below is reduced to those that match the search.
- Click on the blueprint name to display the blueprint details.

RED HAT ENTERPRISE LINUX

Privileged redhat

Back to Blueprints > example-atlas

example-atlas Automatically Tuned Linear Algebra Software

Details Selected Components Images

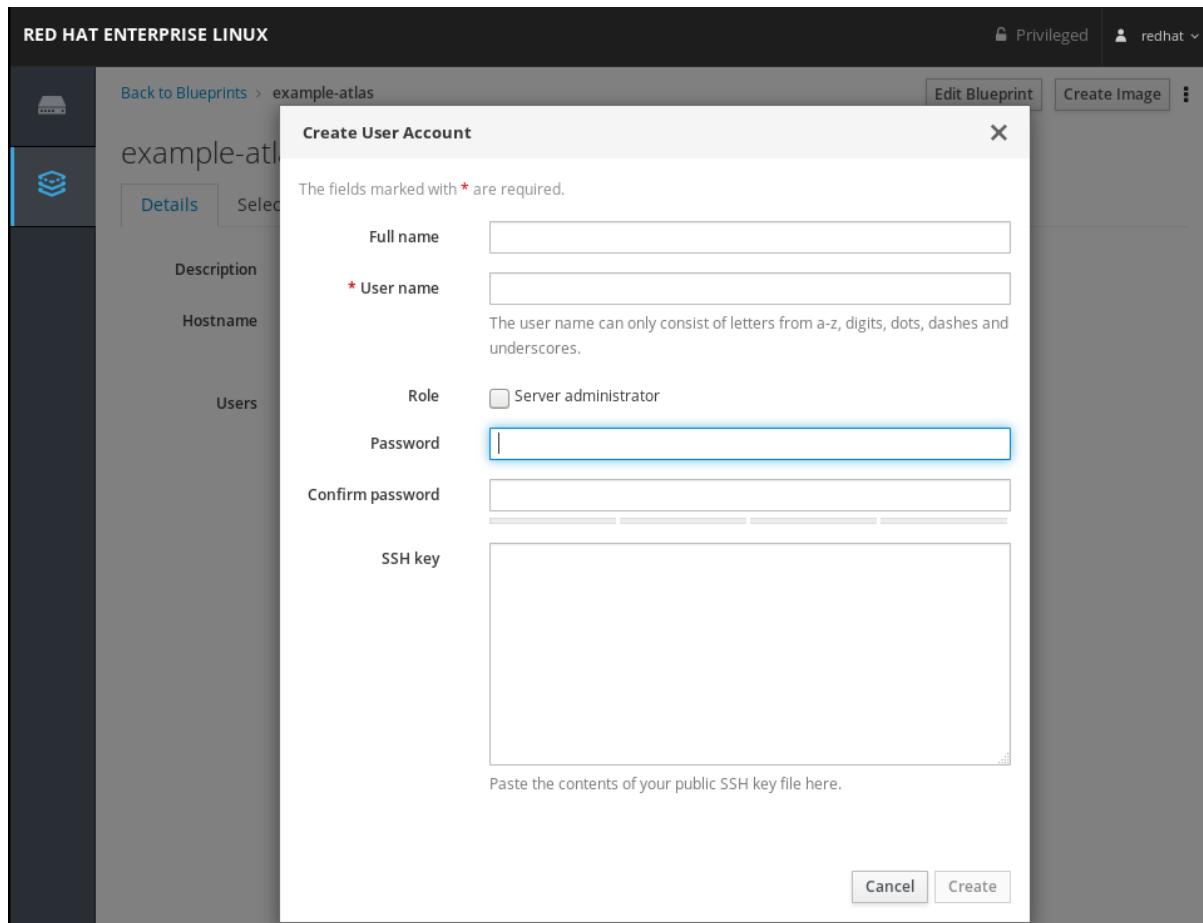
Description: Automatically Tuned Linear Algebra Software

Hostname: If no hostname is provided, the hostname will be determined by the OS.

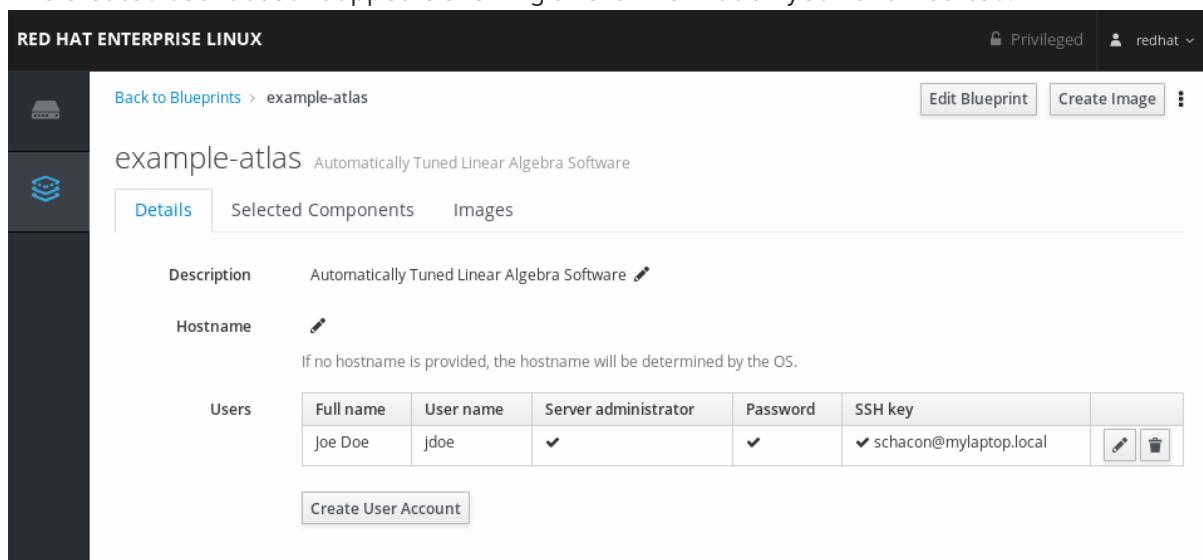
Users:

- Click **Create User Account**

This will open a window with fields for user account creation.



- Fill in the details. Notice that when you insert the name, the **User name** field autocompletes, suggesting a username.
- Once you have inserted all the desired details, click **Create**.
- The created user account appears showing all the information you have inserted.



- To create further user accounts for the blueprint, repeat the process.

11.8. CREATING A USER ACCOUNT WITH SSH KEY

The images created by Image Builder have the root account locked and no other accounts included. Such configuration is provided in order to ensure that images are secure, by not having a default password. Image Builder enables you to create a user account with SSH key for a blueprint so that you can authenticate to the image that created from the blueprint. To do so, first, create a blueprint. Then, you will create a user account with a password and an SSH key. The following example shows how to create a Server administrator user with an SSH key configured.

Prerequisites

- You have created an SSH key that will be paired with the created user later on in the process.
- You have opened the Image Builder interface of the RHEL 8 web console in a browser.
- You have an existing blueprint

Procedure

1. Locate the blueprint that you want to create a user account for by entering its name or a part of it into the search box at the top left, and press **Enter**.

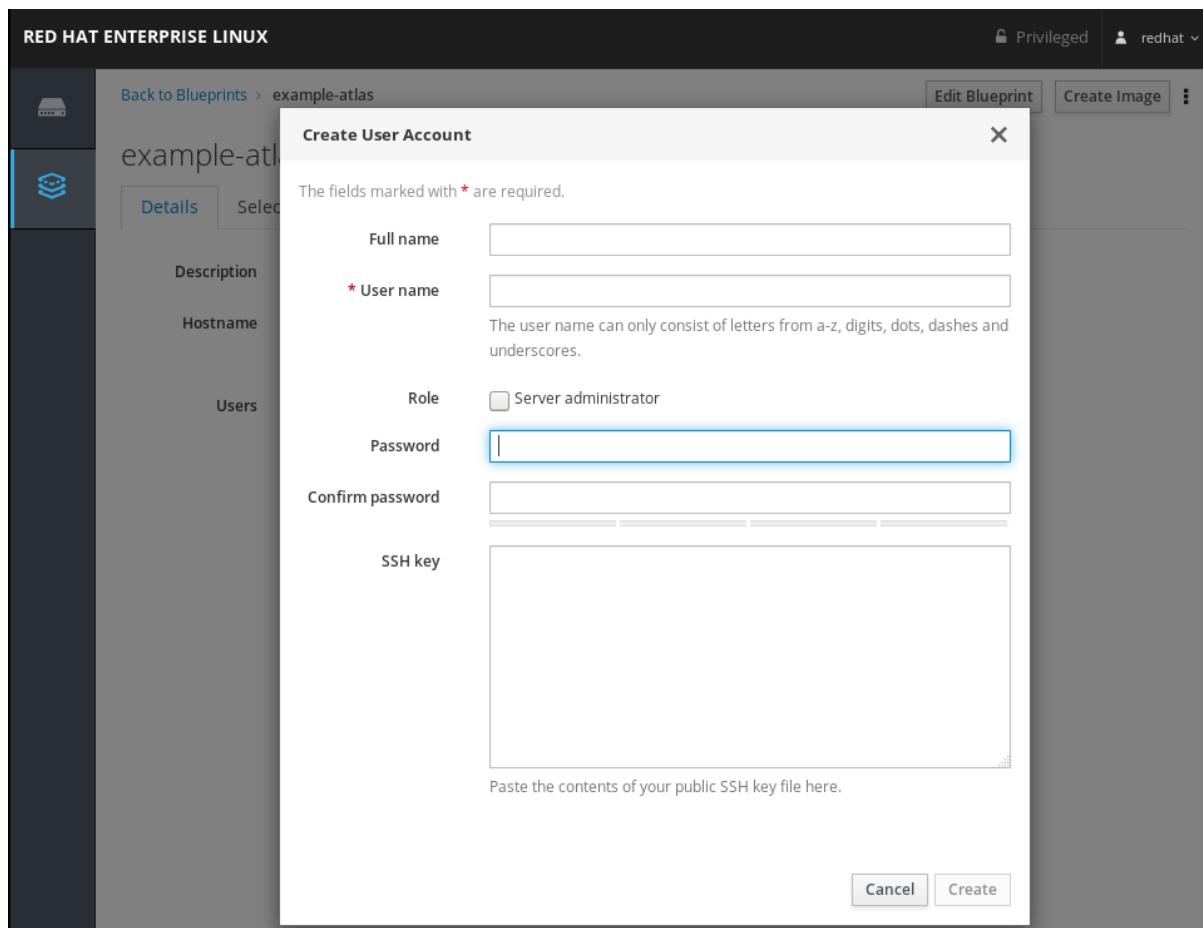
The search is added to the list of filters under the text entry field, and the list of blueprints below is reduced to those that match the search.

2. Click on the blueprint name to display the blueprint details.

The screenshot shows the 'example-atlas' blueprint details page. The top navigation bar includes 'RED HAT ENTERPRISE LINUX', 'Privileged', 'redhat', 'Edit Blueprint', 'Create Image', and a three-dot menu. The main content area shows the blueprint name 'example-atlas' and its description 'Automatically Tuned Linear Algebra Software'. It has tabs for 'Details', 'Selected Components', and 'Images'. Under 'Details', there are fields for 'Description' (Automatically Tuned Linear Algebra Software), 'Hostname' (with a note: 'If no hostname is provided, the hostname will be determined by the OS'), and 'Users' (with a 'Create User Account' button). On the left, there is a sidebar with a vertical bar and a 'Back to Blueprints' link.

3. Click **Create User Account**

This will open a window with fields for user account creation

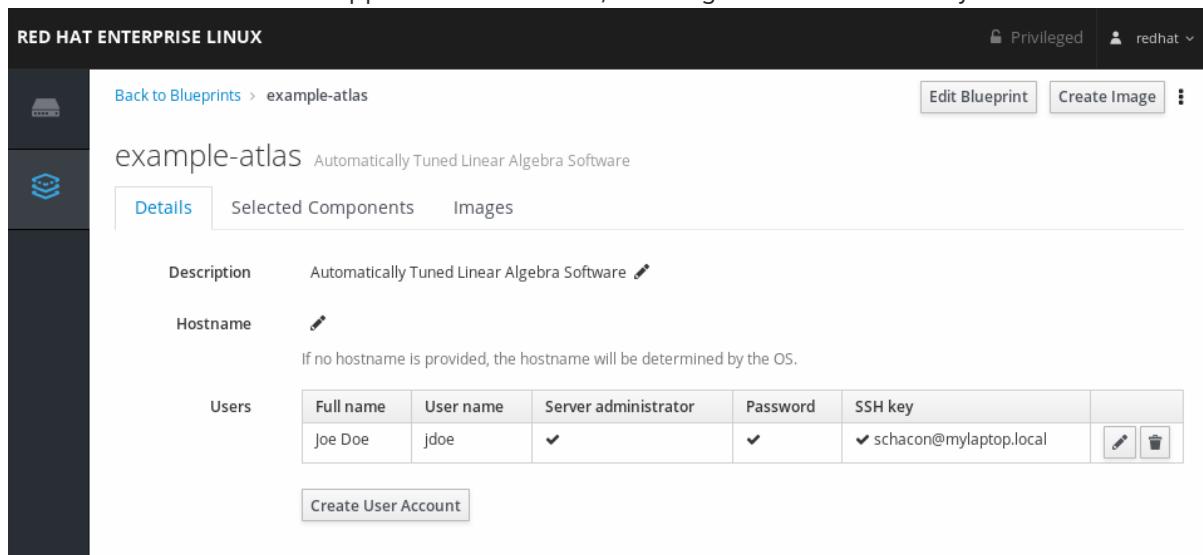


4. Fill in the details. Notice that when you insert the name, the **User name** field autocompletes, suggesting a username.

If you want to provide administrators rights to the user account you are creating, check the **Role** field.

Paste the content of your public SSH key file.

5. Once you have inserted all the desired details, click **Create**.
6. The new user account will appear in the user list, showing all the information you have inserted.



1. To create further user accounts for the blueprint,

Additional resources

- For more details on SSH key, see the [Using SSH Keys](#).

CHAPTER 12. PREPARING AND UPLOADING CLOUD IMAGES WITH IMAGE BUILDER

Image Builder can create custom system images ready for use in clouds of various providers. To use your customized RHEL system image in a cloud, create the system image with Image Builder using the respective output type, configure your system for uploading the image, and upload the image to your cloud account.

12.1. PREPARING FOR UPLOADING AWS AMI IMAGES

This describes steps to configure a system for uploading AWS AMI images.

Prerequisites

- You must have an Access Key ID configured in the [AWS IAM account manager](#).
- You must have a writable [S3 bucket](#) prepared.

Procedure

1. Install Python 3 and the **pip** tool:

```
# yum install python3
# yum install python3-pip
```

2. Install the [AWS command-line tools](#) with **pip**:

```
# pip3 install awscli
```

3. Configure the AWS command-line client according to your AWS access details:

```
$ aws configure
AWS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [None]:
Default output format [None]:
```

4. Configure the AWS command-line client to use your bucket:

```
$ BUCKET=bucketname
$ aws s3 mb s3://$BUCKET
```

Replace *bucketname* with the actual bucket name.

5. Create a **vmimport** S3 Role in IAM and grant it permissions to access S3, if you have not already done so in the past:

```
$ printf '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Principal": { "Service": "vmie.amazonaws.com" }, "Action": "sts:AssumeRole", "Condition": { "StringEquals":{ "sts:ExternalId": "vmimport" } } } ] }' > trust-policy.json
$ printf '{ "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": [ "s3:GetBucketLocation", "s3:GetObject", "s3>ListBucket" ], "Resource": [ "arn:aws:s3:::%s", "arn:aws:s3:::%s/*" ] }, { "Effect": "Allow", "Action": [ "ec2:ModifySnapshotAttribute",
```

```

"ec2:CopySnapshot", "ec2:RegisterImage", "ec2:Describe*" ], "Resource": "*" } ] } } $BUCKET
$BUCKET > role-policy.json
$ aws iam create-role --role-name vmimport --assume-role-policy-document file://trust-
policy.json
$ aws iam put-role-policy --role-name vmimport --policy-name vmimport --policy-document
file://role-policy.json

```

12.2. UPLOADING AN AMI IMAGE TO AWS

This section describes how to upload an AMI image to AWS.

Prerequisites

- Your system must be set up for uploading AWS images.
- You must have an AWS image created by Image Builder. Use the **ami** output type in CLI or **Amazon Machine Image Disk (.ami)** in GUI when creating the image.

Procedure

1. Push the image to S3:

```

$ AMI=8db1b463-91ee-4fd9-8065-938924398428-disk.ami
$ aws s3 cp $AMI s3://$BUCKET
Completed 24.2 MiB/4.4 GiB (2.5 MiB/s) with 1 file(s) remaining
...

```

2. After the upload to S3 ends, import the image as a snapshot into EC2:

```

$ printf '{ "Description": "my-image", "Format": "raw", "UserBucket": { "S3Bucket": "%s",
"S3Key": "%s" } }' $BUCKET $AMI > containers.json
$ aws ec2 import-snapshot --disk-container file://containers.json

```

Replace *my-image* with the name of the image.

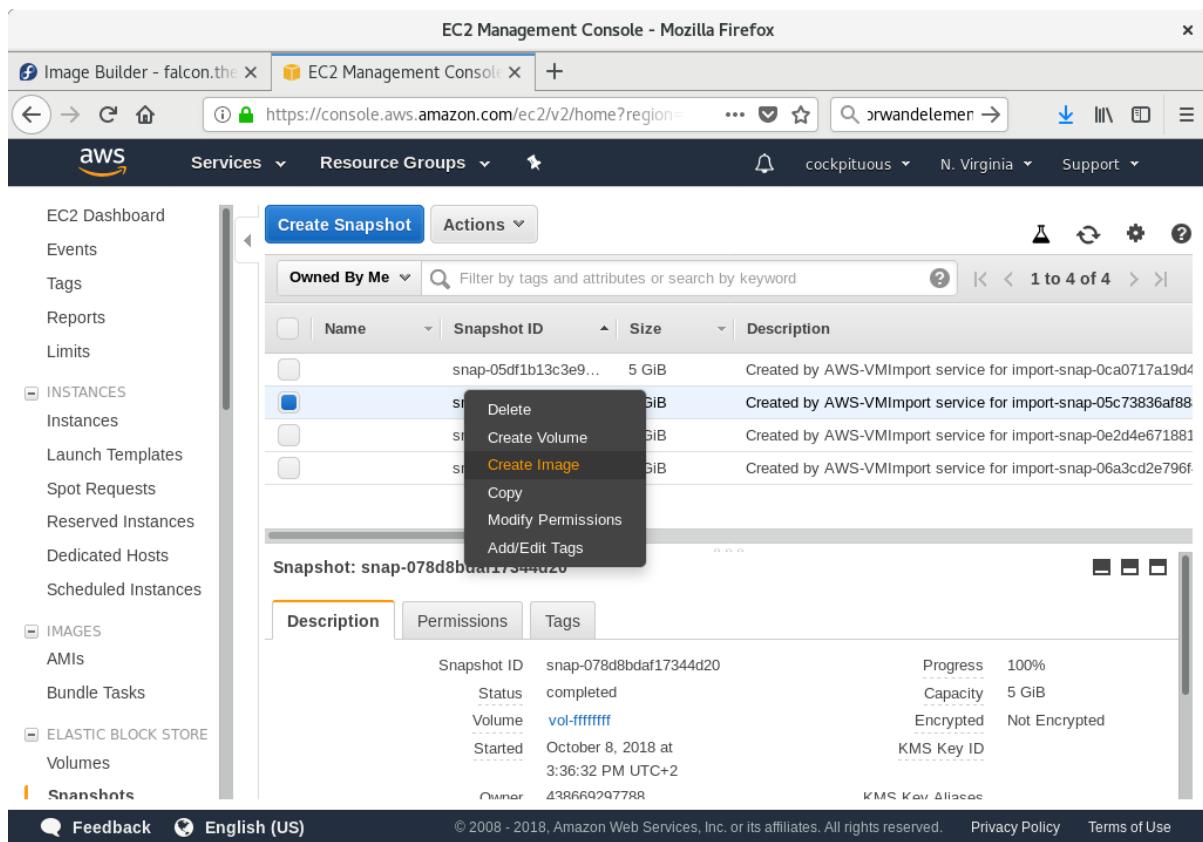
To track progress of the import, run:

```

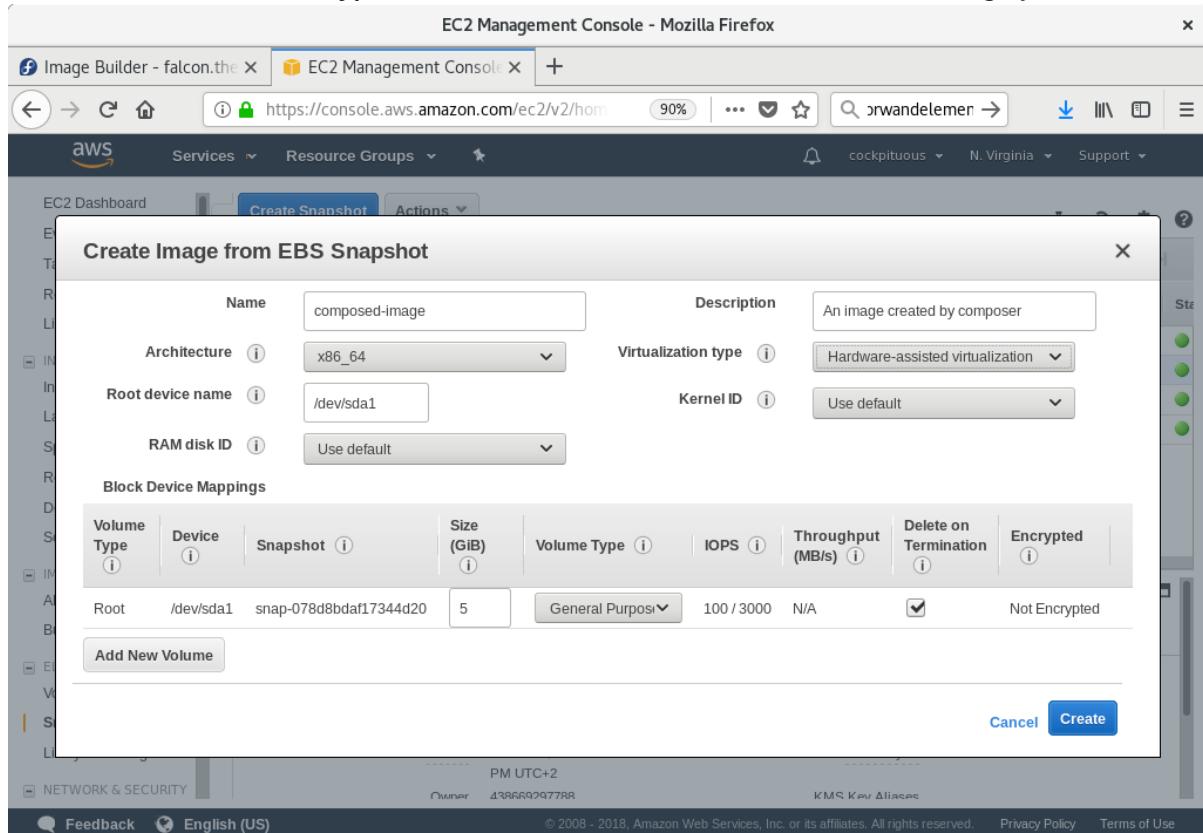
$ aws ec2 describe-import-snapshot-tasks --filters Name=task-state,Values=active

```

3. Create an image from the uploaded snapshot by selecting the snapshot in the EC2 console, right clicking on it and selecting **Create Image**:



4. Select the **Virtualization type of Hardware-assisted virtualization** in the image you create:



5. Now you can run an instance using whatever mechanism you like (CLI or AWS Console) from the snapshot. Use your private key via SSH to access the resulting EC2 instance. Log in as **ec2-user**.

12.3. PREPARING FOR UPLOADING AZURE VHD IMAGES

This describes steps to upload an VHD image to Azure.

Prerequisites

- You must have a usable Azure resource group and storage account.

Procedure

1. Install python2:

```
# yum install python2
```



NOTE

python2 package must be installed because since the AZ CLI depends specifically on python 2.7

2. Import the Microsoft repository key:

```
# rpm --import https://packages.microsoft.com/keys/microsoft.asc
```

3. Create a local azure-cli repository information:

```
# sh -c 'echo -e "[azure-cli]\nname=Azure\nCLI\nbaseurl=https://packages.microsoft.com/yumrepos/azure-\ncli\nenabled=1\npgpcheck=1\npgpkey=https://packages.microsoft.com/keys/microsoft.asc" >\n/etc/yum.repos.d/azure-cli.repo'
```

4. Install the Azure CLI:

```
# yumdownloader azure-cli
# rpm -ivh --nodeps azure-cli-2.0.64-1.el7.x86_64.rpm
```



NOTE

The downloaded version of the Azure CLI package may vary depending on the current downloaded version.

5. Run the Azure CLI:

```
$ az login
```

The terminal shows the message 'Note, we have launched a browser for you to login. For old experience with device code, use "az login --use-device-code"' and opens a browser where you can login.



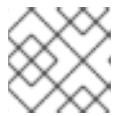
NOTE

If you are running a remote (SSH) session, the link will not open in the browser. In this case, you can use the link provided and thus be able to login and authenticate your remote session. To sign in, use a web browser to open the page <https://microsoft.com/devicelogin> and enter the code XXXXXXXXX to authenticate.

6. List the keys for the storage account in Azure:

```
$ GROUP=resource-group-name
$ ACCOUNT=storage-account-name
$ az storage account keys list --resource-group $GROUP --account-name $ACCOUNT
```

Replace *resource-group-name* with name of the Azure resource group and *storage-account-name* with name of the Azure storage account.



NOTE

You can list the available resources using the command:

```
$ az resource list
```

7. Make note of value **key1** in the output of the previous command, and assign it to an environment variable:

```
$ KEY1=value
```

8. Create a storage container:

```
$ CONTAINER=storage-account-name
$ az storage container create --account-name $ACCOUNT \
--account-key $KEY1 --name $CONTAINER
```

Replace *storage-account-name* with name of the storage account.

Additional resources

- [Azure CLI](#)

12.4. UPLOADING VHD IMAGES TO AZURE

This describes steps to upload an VHD image to Azure.

Prerequisites

- Your system must be set up for uploading Azure VHD images.
- You must have an Azure VHD image created by Image Builder. Use the **vhd** output type in CLI or **Azure Disk Image (.vhd)** in GUI when creating the image.

Procedure

1. Push the image to Azure and create an instance from it:

```
$ VHD=25ccb8dd-3872-477f-9e3d-c2970cd4bbaf-disk.vhd
$ az storage blob upload --account-name $ACCOUNT --container-name $CONTAINER --file
$VHD --name $VHD --type page
...
```

- Once the upload to the Azure BLOB completes, create an Azure image from it:

```
$ az image create --resource-group $GROUP --name $VHD --os-type linux --location eastus
--source https://$ACCOUNT.blob.core.windows.net/$CONTAINER/$VHD
- Running ...
```

- Create an instance either with the Azure portal, or a command similar to the following:

```
$ az vm create --resource-group $GROUP --location eastus --name $VHD --image $VHD --
admin-username azure-user --generate-ssh-keys
- Running ...
```

- Use your private key via SSH to access the resulting instance. Log in as **azure-user**.

12.5. UPLOADING VMDK IMAGES TO VSphere

Image Builder can generate images suitable for uploading to a VMware ESXi or vSphere system. This describes steps to upload an VMDK image to VMware vSphere.



NOTE

Because VMWare deployments typically does not have cloud-init configured to inject user credentials to virtual machines, we must perform that task ourselves on the blueprint.

Prerequisites

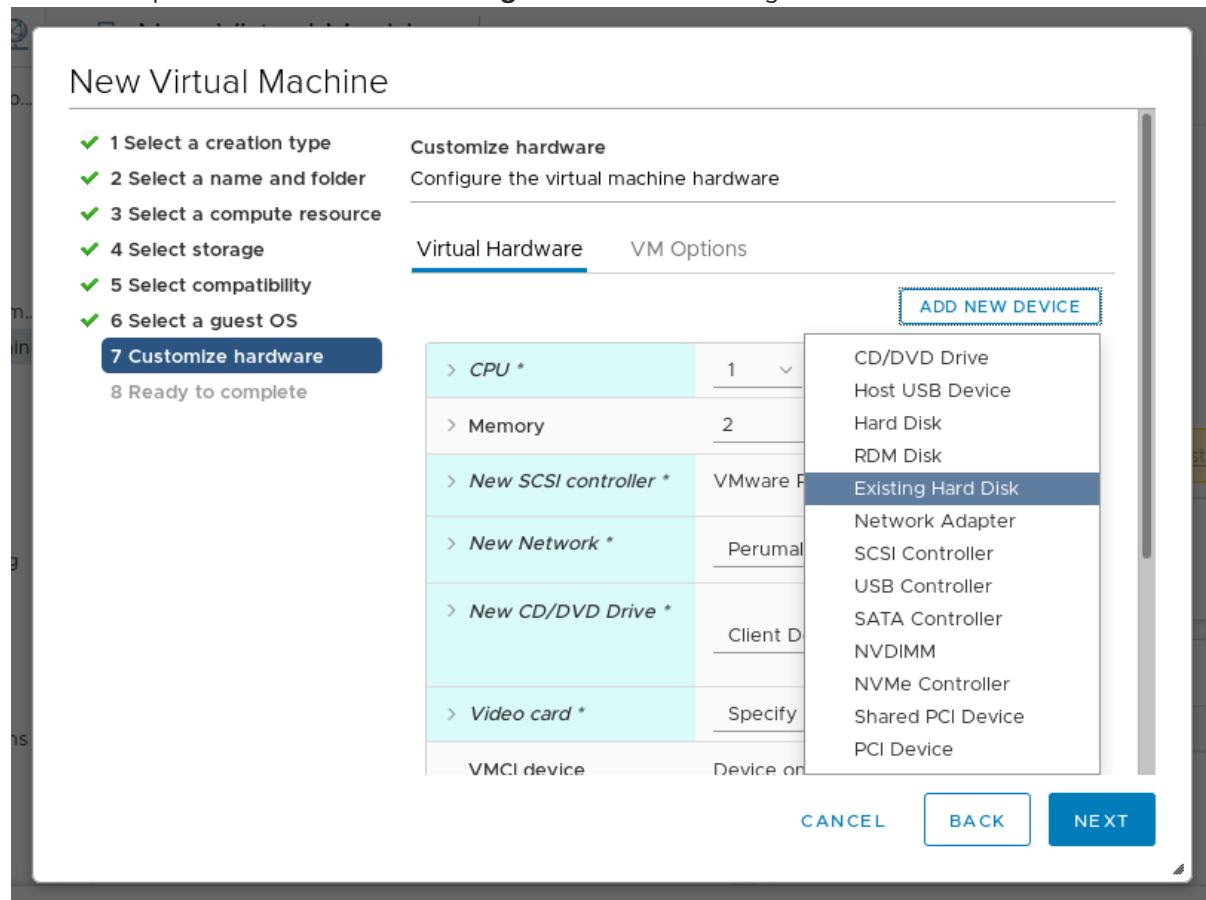
- You must have an VMDK image created by Image Builder. Use the **vmdk** output type in CLI or **VMware Virtual Machine Disk (.vmdk)** in GUI when creating the image.

Procedure

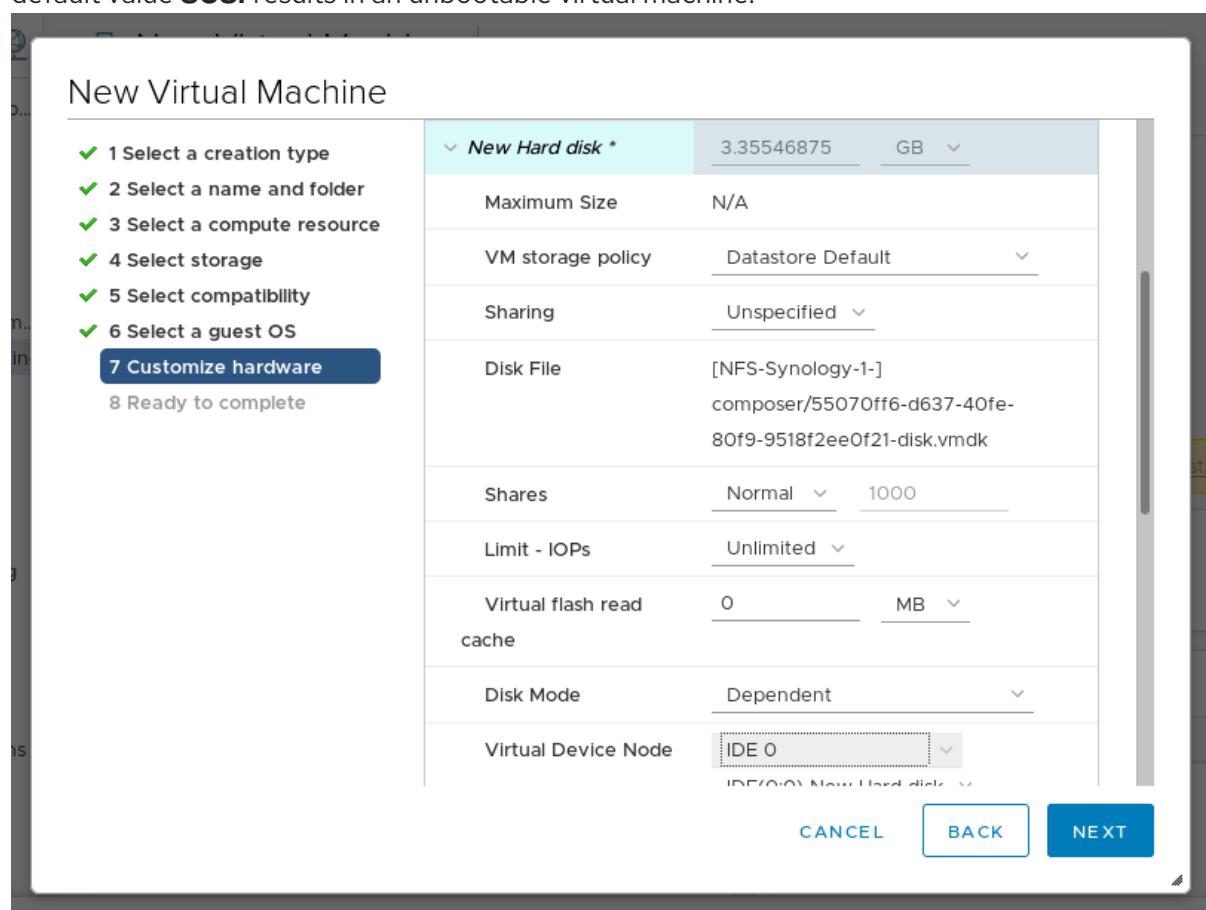
- Upload the image into vSphere via HTTP. Click on **Upload Files** in the vCenter:

Name	Size	Modified	Type	Path
55070ff6...	1,396,864 KB	10/10/2018, 10:0...	Virtual Disk	[NFS-Synolog...
ba00ea7...	1,527,488 KB	10/10/2018, 10:0...	Virtual Disk	[NFS-Synolog...
disk.vmdk	1,693,312 KB	10/09/2018, 9:0...	Virtual Disk	[NFS-Synolog...
Isilogic-5b...	588,630.5 KB	10/09/2018, 11:0...	Virtual Disk	[NFS-Synolog...
New Virtu...	0.08 KB	10/10/2018, 7:1...	File	[NFS-Synolog...
rhel8-scsi...	1,393,216 KB	10/10/2018, 10:0...	Virtual Disk	[NFS-Synolog...
Test with ...	0.08 KB	10/10/2018, 7:1...	File	[NFS-Synolog...

2. When you create a VM, on the **Device Configuration**, delete the default **New Hard Disk** and use the drop-down to select an **Existing Hard Disk** disk image:



3. Make sure you use an **IDE** device as the **Virtual Device Node** for the disk you create. The default value **SCSI** results in an unbootable virtual machine.



12.6. UPLOADING QCOW2 IMAGE TO OPENSTACK

Image Builder can generate images suitable for uploading to OpenStack cloud deployments, and starting instances there. This describes steps to upload an QCOW2 image to OpenStack.

Prerequisites

- You must have an OpenStack-specific image created by Image Builder. Use the **openstack** output type in CLI or **OpenStack Image (.qcow2)** in GUI when creating the image.



WARNING

Image Builder also offers a generic QCOW2 image type output format as **qcow2** or **QEMU QCOW2 Image (.qcow2)**. Do not mistake it with the OpenStack image type which is also in the QCOW2 format, but contains further changes specific to OpenStack.

Procedure

1. Upload the image to OpenStack and start an instance from it. Use the **Images** interface to do this:

Create An Image

Name: *

Description:

Image Source:

Image File

Image File

96268ffb-2c71-4e97-a855-7ac25e983a6e-disk.qcow2

Format: *

QCOW2 - QEMU Emulator

Architecture:

x86_64

Minimum Disk (GB):

5

Minimum Ram (MB):

1024

Public:

Protected:

2. Start an instance with that image:

Launch Instance

Details * Access & Security * Networking * Post-Creation Advanced Options

Availability Zone:
nova

Instance Name: *
my-instance

Flavor: *
m1.small

Some flavors not meeting minimum image requirements have been disabled.

Instance Count: *
1

Instance Boot Source: *
Boot from image

Image Name:
96268ffb-2c71-4e97-a855-7ac25e983a6e-disk.qcow2

Specify the details for launching an instance.
The chart below shows the resources used by this project in relation to the project's quotas.

Flavor Details

Name	m1.small
VCPUs	1
Root Disk	20 GB
Ephemeral Disk	0 GB
Total Disk	20 GB
RAM	2,048 MB

Project Limits

Number of Instances	4 of 10 Used
Number of VCPUs	17 of 20 Used
Total RAM	34,816 of 51,200 MB Used

Cancel **Launch**

3. You can run the instance using any mechanism (CLI or OpenStack web UI) from the snapshot. Use your private key via SSH to access the resulting instance. Log in as **cloud-user**.

12.7. PREPARING FOR UPLOADING IMAGES TO ALIBABA



NOTE

The custom image verification is an optional task. Image Builder generates images that conform to Alibaba's requirements.

This section describes steps to verify custom images that you can deploy on Alibaba Cloud. The images will need a specific configuration to boot successfully, because Alibaba Cloud requests the custom images to meet certain requirements before you use it. For this, it is recommended that you use the Alibaba **image_check** tool.

Prerequisites

- You must have an Alibaba image created by Image Builder.

Procedure

1. Connect to the system containing the image you want to check it by the Alibaba **image_check tool**.
2. Download the **image_check tool**:

```
$ curl -O http://docs-aliyun.cn-hangzhou.oss.aliyun-inc.com/assets/attach/73848/cn_zh/1557459863884/image_check
```

3. Change the file permission of the image compliance tool:

```
# chmod +x image_check
```

4. Run the command to start the image compliance tool checkup:

```
# ./image_check
```

The tool verifies the system configuration and generate a report that is displayed on your screen. The `image_check` tool saves this report in the same folder where the image compliance tool is running.

5. If any of the **Detection Items** fail, follow the instructions to correct it. For more information, see link: [Detection items section](#).

Additional resources

- For more details, see [Image Compliance Tool](#).

12.8. UPLOADING IMAGES TO ALIBABA

This section describes how to upload an Alibaba image to Object Storage Service (OSS).

Prerequisites

- Your system is set up for uploading Alibaba images.
- You must have an Alibaba image created by Image Builder. Use the **ami** output type on RHEL 7 or Alibaba on RHEL 8 when creating the image.
- You have a bucket. See [Creating a bucket](#).
- You have an [active Alibaba Account](#).
- You activated [OSS](#).

Procedure

1. Log in to the [OSS console](#).
2. On the left side Bucket menu, select the bucket to which you want to upload an image.
3. On the right upper menu, click **Files** tab.
4. Click **Upload**. A window dialog opens on the right side. Choose the following information:
 - **Upload To:** Choose to upload the file to the **Current** directory or to a **Specified** directory.

- **File ACL:** Choose the type of permission of the uploaded file.
5. Click **Upload**.
 6. Choose the image you want to upload.
 7. Click **Open**.

As a result, the custom image is uploaded to OSS Console.

Additional resources

- For more details on uploading custom images to Alibaba Cloud, see [Upload an object](#).
- For more details on creating instances from custom images, see [Creating an instance from custom images](#).
- For more details on importing custom images to Alibaba, see [Importing images](#).

12.9. IMPORTING IMAGES TO ALIBABA

This section describes how to import an Alibaba image to Elastic Cloud Console (ECS).

Prerequisites

- You have uploaded the image to Object Storage Service (OSS).

Procedure

1. Log in to the [ECS console](#).
 - i. On the left side menu, click **Images**.
 - ii. On the right upper side, click **Import Image**. A window dialog opens.
 - iii. Confirm that you have set up the correct region where the image is located. Enter the following information:
 - a. **OSS Object Address:** See how to obtain [OSS Object Address](#).
 - b. **Image Name:**
 - c. **Operating System:**
 - d. **System Disk Size:**
 - e. **System Architecture:**
 - f. **Platform:** Red Hat
 - iv. Optionally, provide the following details:
 - g. **Image Format:** qcow2 or ami, depending on the uploaded image format.
 - h. **Image Description:**
 - i. **Add Images of Data Disks**

The address can be determined in the OSS management console after selecting the required bucket in the left menu, select Files section and then click on Details link on the right for the appropriate image. A window will appear on the right side of the screen, showing image details. The OSS object address is in the URL box.

2. Click **OK**.



NOTE

The importing process time can vary depending on the image size.

As a result, the custom image is imported to ECS Console. You can create an instance from the custom image.

Additional resources

- For more details on importing custom images to Alibaba Cloud, see [Notes for importing images](#).
- For more details on creating instances from custom images, see [Creating an instance from custom images](#).
- For more details on creating instances from custom images, see [Upload an object](#).

12.10. CREATING AN INSTANCE OF A CUSTOM IMAGE USING ALIBABA

You can create instances of the custom image using Alibaba ECS Console.

Prerequisites

- You have activated [OSS](#) and uploaded your custom image.
- You have successfully imported your image to ECS Console.

Procedure

1. Log in to the [ECS console](#).
2. On the left side menu, choose **Instances**.
3. In the top corner, click **Create Instance**. You are redirected to a new window.
4. Fill in all the required information. See [Creating an instance by using the wizard](#) for more details.
5. Click **Create Instance** and confirm the order.



NOTE

You can see the option **Create Order** instead of **Create Instance**, depending on your subscription.

As a result, you have an active instance ready for deployment.

Additional resources

- For further details on creating an instance, see [Creating an instance by using a custom image](#) .
- For more details on providing details when creating an instance, see [Create an instance by using the wizard](#).

CHAPTER 13. PERFORMING AN AUTOMATED INSTALLATION USING KICKSTART

CHAPTER 14. KICKSTART INSTALLATION BASICS

The following provides basic information about Kickstart and how to use it to automate installing Red Hat Enterprise Linux.

14.1. WHAT ARE KICKSTART INSTALLATIONS

Kickstart provides a way to automate the RHEL installation process, either partially or fully.

Kickstart files contain some or all of the RHEL installation options. For example, the time zone, how the drives should be partitioned, or which packages should be installed. Providing a prepared Kickstart file allows an installation without the need for any user intervention. This is especially useful when deploying Red Hat Enterprise Linux on a large number of systems at once.

Kickstart files also provide more options regarding software selection. When installing Red Hat Enterprise Linux manually using the graphical installation interface, the software selection is limited to pre-defined environments and add-ons. A Kickstart file allows you to install or remove individual packages as well.

Kickstart files can be kept on a single server system and read by individual computers during the installation. This installation method supports the use of a single Kickstart file to install Red Hat Enterprise Linux on multiple machines, making it ideal for network and system administrators.

All Kickstart scripts and log files of their execution are stored in the `/tmp` directory of the newly installed system to assist with debugging installation issues.



NOTE

In previous versions of Red Hat Enterprise Linux, Kickstart could be used for upgrading systems. Starting with Red Hat Enterprise Linux 7, this functionality has been removed and system upgrades are instead handled by specialized tools. For details on upgrading to Red Hat Enterprise Linux 8, see [Upgrading from RHEL 7 to RHEL 8](#) and [Considerations in adopting RHEL 8](#).

14.2. AUTOMATED INSTALLATION WORKFLOW

Kickstart installations can be performed using a local DVD, a local hard drive, or a NFS, FTP, HTTP, or HTTPS server. This section provides a high level overview of Kickstart usage.

1. Create a Kickstart file. You can write it by hand, copy a Kickstart file saved after a manual installation, or use an online generator tool to create the file, and edit it afterward. See [Creating Kickstart files](#).
2. Make the Kickstart file available to the installation program on removable media, a hard drive or a network location using an HTTP(S), FTP, or NFS server. See [Making Kickstart files available to the installation program](#).
3. Create the boot medium which will be used to begin the installation. See [Creating a bootable installation medium](#) and [Preparing to install from the network using PXE](#).
4. Make the installation source available to the installation program. See [Creating installation sources for Kickstart installations](#).
5. Start the installation using the boot medium and the Kickstart file. See [Starting Kickstart installations](#).

If the Kickstart file contains all mandatory commands and sections, the installation finishes automatically. If one or more of these mandatory parts are missing, or if an error occurs, the installation requires manual intervention to finish.

CHAPTER 15. CREATING KICKSTART FILES

You can create a Kickstart file using the following methods:

- Use the online Kickstart configuration tool.
- Copy the Kickstart file created as a result of a manual installation.
- Write the entire Kickstart file manually. Note that editing an already existing file from the other methods is faster, so this method is not recommended.
- Convert the Red Hat Enterprise Linux 7 Kickstart file for Red Hat Enterprise Linux 8 installation.

Note that some highly specific installation options can be configured only by manual editing of the Kickstart file.

15.1. CREATING A KICKSTART FILE WITH THE KICKSTART CONFIGURATION TOOL

Users with a Red Hat Customer Portal account can use the Kickstart Generator tool in the Customer Portal Labs to generate Kickstart files online. This tool will walk you through the basic configuration and enables you to download the resulting Kickstart file.



NOTE

The tool currently does not support any advanced partitioning.

Prerequisites

- You must have a Red Hat Customer Portal account.

Procedure

1. Open the Kickstart generator lab information page at <https://access.redhat.com/labsinfo/kickstartconfig>
2. Click the **Go to Application** button to the left of heading and wait for the next page to load.
3. Select **Red Hat Enterprise Linux 8** in the drop-down menu and wait for the page to update.
4. Describe the system to be installed using the fields in the form.
You can use the links on the left side of the form to quickly navigate between sections of the form.
5. To download the generated Kickstart file, click the red **Download** button at the top of the page.
Your web browser will save the file.

15.2. CREATING A KICKSTART FILE BY PERFORMING A MANUAL INSTALLATION

The recommended approach to creating Kickstart files is to use the file created by a manual installation of Red Hat Enterprise Linux. After an installation completes, all choices made during the installation are saved into a Kickstart file named **anaconda-ks.cfg**, located in the **/root/** directory on the installed

system. You can use this file to reproduce the installation in the same way as before. Alternatively, copy this file, make any changes you need, and use the resulting configuration file for further installations.

Procedure

1. Install RHEL. For more details, see [Performing a standard RHEL installation](#). During the installation, create a user with administrator privileges.
2. Finish the installation and reboot into the installed system.
3. Log into the system with the administrator account.
4. Copy the file **/root/anaconda-ks.cfg** to a location of your choice.
 - To display the file contents in terminal:

```
# cat /root/anaconda-ks.cfg
```

You can copy the output and save to another file of your choice.

- To copy the file to another location, use the file manager. Remember to change permissions on the copy, so that the file can be read by non-root users.

CAUTION

The file contains information about users and passwords.

Additional resources

- [Performing a standard RHEL installation](#)

15.3. CONVERTING A RHEL 7 KICKSTART FILE FOR RHEL 8 INSTALLATION

You can use the Kickstart Converter tool to convert a RHEL 7 Kickstart file for use in a new RHEL 8 installation. For more information about the tool and how to use it to convert a RHEL 7 Kickstart file, see <https://access.redhat.com/labs/kickstartconvert/>

CHAPTER 16. MAKING KICKSTART FILES AVAILABLE TO THE INSTALLATION PROGRAM

The following provides information about making the Kickstart file available to the installation program on the target system.

16.1. PORTS FOR NETWORK-BASED INSTALLATION

The following table lists the ports that must be open on the server providing the files for each type of network-based installation.

Table 16.1. Ports for network-based installation

Protocol used	Ports to open
HTTP	80
HTTPS	443
FTP	21
NFS	2049, 111, 20048
TFTP	69

Additional resources

- See the [Securing networks](#) document for more information.

16.2. MAKING A KICKSTART FILE AVAILABLE ON AN NFS SERVER

This procedure describes how to store the Kickstart script file on an NFS server. This method enables you to install multiple systems from a single source without having to use physical media for the Kickstart file.

Prerequisites

- You must have administrator level access to a server with Red Hat Enterprise Linux 8 on the local network.
- The system to be installed must be able to connect to the server.
- Firewall on the server must allow connections from the system you are installing to.

Procedure

1. Install the **nfs-utils** package by running the following command as root:

```
# yum install nfs-utils
```

2. Copy the Kickstart file to a directory on the NFS server.

3. Open the **/etc/exports** file using a text editor and add a line with the following syntax:

```
/exported_directory/ clients
```

4. Replace */exported_directory/* with the full path to the directory holding the Kickstart file.

Instead of *clients*, use the host name or IP address of the computer that is to be installed from this NFS server, the subnetwork from which all computers are to have access the ISO image, or the asterisk sign (*) if you want to allow any computer with network access to the NFS server to use the ISO image. See the *exports(5)* man page for detailed information about the format of this field.

A basic configuration that makes the **/rhel8-install/** directory available as read-only to all clients is:

```
/rhel8-install *
```

5. Save the **/etc/exports** file and exit the text editor.

6. Start the nfs service:

```
# systemctl start nfs-server.service
```

If the service was running before you changed the **/etc/exports** file, enter the following command, in order for the running NFS server to reload its configuration:

```
# systemctl reload nfs-server.service
```

The Kickstart file is now accessible over NFS and ready to be used for installation.



NOTE

When specifying the Kickstart source, use **nfs:** as the protocol, the server's host name or IP address, the colon sign (:), and the path inside directory holding the file. For example, if the server's host name is **myserver.example.com** and you have saved the file in **/rhel8-install/my-ks.cfg**, specify **inst.ks=nfs:myserver.example.com:/rhel8-install/my-ks.cfg** as the installation source boot option.

Additional resources

- For details on setting up TFTP server for PXE boot from network, see [Preparing to install from the network using PXE](#) in the *Performing an advanced RHEL installation* document.

16.3. MAKING A KICKSTART FILE AVAILABLE ON AN HTTP OR HTTPS SERVER

This procedure describes how to store the Kickstart script file on an HTTP or HTTPS server. This method enables you to install multiple systems from a single source without having to use physical media for the Kickstart file.

Prerequisites

- You must have administrator level access to a server with Red Hat Enterprise Linux 8 on the local network.
- The system to be installed must be able to connect to the server.
- Firewall on the server must allow connections from the system you are installing to.

Procedure

1. Install the **httpd** package by running the following command as root:

```
# yum install httpd
```



WARNING

If your Apache web server configuration enables SSL security, verify that you only enable the TLSv1 protocol, and disable SSLv2 and SSLv3. This is due to the POODLE SSL vulnerability (CVE-2014-3566). See <https://access.redhat.com/solutions/1232413> for details.



IMPORTANT

If you use an HTTPS server with a self-signed certificate, you must boot the installation program with the **inst.noverifyssl** option.

2. Copy the Kickstart file to the HTTP(S) server into a subdirectory of the **/var/www/html/** directory.
3. Start the httpd service:

```
# systemctl start httpd.service
```

The Kickstart file is now accessible and ready to be used for installation.



NOTE

When specifying the location of the Kickstart file, use **http://** or **https://** as the protocol, the server's host name or IP address, and the path of the Kickstart file, relative to the HTTP server root. For example, if you are using HTTP, the server's host name is **myserver.example.com**, and you have copied the Kickstart file as **/var/www/html/rhel8-install/my-ks.cfg**, specify <http://myserver.example.com/rhel8-install/my-ks.cfg> as the file location.

Additional resources

- For more information about HTTP and FTP servers, see [Deploying different types of servers](#).

16.4. MAKING A KICKSTART FILE AVAILABLE ON AN FTP SERVER

This procedure describes how to store the Kickstart script file on an FTP server. This method enables you to install multiple systems from a single source without having to use physical media for the Kickstart file.

Prerequisites

- You must have administrator level access to a server with Red Hat Enterprise Linux 8 on the local network.
- The system to be installed must be able to connect to the server.
- Firewall on the server must allow connections from the system you are installing to.

Procedure

1. Install the **vsftpd** package by running the following command as root:

```
# yum install vsftpd
```

2. Open and edit the **/etc/vsftpd/vsftpd.conf** configuration file in a text editor.

- a. Change the line **anonymous_enable=NO** to **anonymous_enable=YES**.
- b. Change the line **write_enable=YES** to **write_enable=NO**.
- c. Add lines **pasv_min_port=min_port** and **pasv_max_port=max_port**. Replace *min_port* and *max_port* with the port number range used by FTP server in passive mode, e. g. **10021** and **10031**.
This step can be necessary in network environments featuring various firewall/NAT setups.
- d. Optionally, add custom changes to your configuration. For available options, see the **vsftpd.conf(5)** man page. This procedure assumes that default options are used.



WARNING

If you configured SSL/TLS security in your **vsftpd.conf** file, ensure that you enable only the TLSv1 protocol, and disable SSLv2 and SSLv3. This is due to the POODLE SSL vulnerability (CVE-2014-3566). See <https://access.redhat.com/solutions/1234773> for details.

3. Configure the server firewall.

- a. Enable the firewall:

```
# systemctl enable firewalld  
# systemctl start firewalld
```

- b. Enable in your firewall the FTP port and port range from previous step:

```
# firewall-cmd --add-port min_port-max_port/tcp --permanent
# firewall-cmd --add-service ftp --permanent
# firewall-cmd --reload
```

Replace *min_port-max_port* with the port numbers you entered into the **/etc/vsftpd/vsftpd.conf** configuration file.

4. Copy the Kickstart file to the FTP server into the **/var/ftp/** directory or its subdirectory.
5. Make sure that the correct SELinux context and access mode is set on the file:

```
# restorecon -r /var/ftp/your-kickstart-file.ks
# chmod 444 /var/ftp/your-kickstart-file.ks
```

6. Start the **vsftpd** service:

```
# systemctl start vsftpd.service
```

If the service was running before you changed the **/etc/vsftpd/vsftpd.conf** file, restart the service to load the edited file:

```
# systemctl restart vsftpd.service
```

Enable the **vsftpd** service to start during the boot process:

```
# systemctl enable vsftpd
```

The Kickstart file is now accessible and ready to be used for installations by systems on the same network.



NOTE

When configuring the installation source, use **ftp://** as the protocol, the server's host name or IP address, and the path of the Kickstart file, relative to the FTP server root. For example, if the server's host name is **myserver.example.com** and you have copied the file to **/var/ftp/my-ks.cfg**, specify **ftp://myserver.example.com/my-ks.cfg** as the installation source.

16.5. MAKING A KICKSTART FILE AVAILABLE ON A LOCAL VOLUME

This procedure describes how to store the Kickstart script file on a volume on the system to be installed. This method enables you to bypass the need for another system.

Prerequisites

- You must have a drive that can be moved to the machine to be installed, such as a USB stick.
- The drive must contain a partition that can be read by the installation program. The supported types are **ext2**, **ext3**, **ext4**, **xfs**, and **fat**.
- The drive must be already connected to the system and its volumes mounted.

Procedure

1. List volume information and note the UUID of the volume to which you want to copy the Kickstart file.

```
# lsblk -l -p -o name,rm,ro,hotplug,size,type,mountpoint,uuid
```

2. Navigate to the file system on the volume.
3. Copy the Kickstart file to this file system.
4. Make a note of the string to use later with the **inst.ks=** option. This string is in the form **hd:UUID=volume-UUID:path/to/kickstart-file.cfg**. Note that the path is relative to the file system root, not to the / root of file system hierarchy. Replace *volume-UUID* with the UUID you noted earlier.
5. Unmount all drive volumes:

```
# umount /dev/xyz ...
```

Add all the volumes to the command, separated by spaces.

16.6. MAKING A KICKSTART FILE AVAILABLE ON A LOCAL VOLUME FOR AUTOMATIC LOADING

A specially named Kickstart file can be present in the root of a specially named volume on the system to be installed. This lets you bypass the need for another system, and makes the installation program load the file automatically.

Prerequisites

- You must have a drive that can be moved to the machine to be installed, such as a USB stick.
- The drive must contain a partition that can be read by the installation program. The supported types are **ext2**, **ext3**, **ext4**, **xfs**, and **fat**.
- The drive must be already connected to the system and its volumes mounted.

Procedure

1. List volume information and note the UUID of the volume to which you want to copy the Kickstart file.

```
# lsblk -l -p
```

2. Navigate to the file system on the volume.
3. Copy the Kickstart file into the root of this file system.
4. Rename the Kickstart file to **ks.cfg**.
5. Rename the volume as **OEMDRV**:

- For **ext2**, **ext3**, and **ext4** file systems:

```
# e2label /dev/xyz OEMDRV
```

- For the XFS file system:

```
# xfs_admin -L OEMDRV /dev/xyz
```

Replace `/dev/xyz` with the path to the volume's block device.

6. Unmount all drive volumes:

```
# umount /dev/xyz ...
```

Add all the volumes to the command, separated by spaces.

CHAPTER 17. CREATING INSTALLATION SOURCES FOR KICKSTART INSTALLATIONS

This section describes how to create an installation source for the Boot ISO image using the Binary DVD ISO image that contains the required repositories and software packages.

17.1. TYPES OF INSTALLATION SOURCE

You can use one of the following installation sources for minimal boot images:

- **DVD:** Burn the Binary DVD ISO image to a DVD. The installation program will automatically install the software packages from the DVD.
- **Hard drive or USB drive:** Copy the Binary DVD ISO image to the drive and configure the installation program to install the software packages from the drive. If you use a USB drive, verify that it is connected to the system before the installation begins. The installation program cannot detect media after the installation begins.
 - **Hard drive limitation:** The Binary DVD ISO image on the hard drive must be on a partition with a file system that the installation program can mount. The supported file systems are **xfs**, **ext2**, **ext3**, **ext4**, and **vfat (FAT32)**.



WARNING

On Microsoft Windows systems, the default file system used when formatting hard drives is NTFS. The exFAT file system is also available. However, neither of these file systems can be mounted during the installation. If you are creating a hard drive or a USB drive as an installation source on Microsoft Windows, verify that you formatted the drive as FAT32. Note that the FAT32 file system cannot store files larger than 4 GiB.

In Red Hat Enterprise Linux 8, you can enable installation from a directory on a local hard drive. To do so, you need to copy the contents of the DVD ISO image to a directory on a hard drive and then specify the directory as the installation source instead of the ISO image. For example:

inst.repo=hd:<device>:<path to the directory>

- **Network location:** Copy the Binary DVD ISO image or the installation tree (extracted contents of the Binary DVD ISO image) to a network location and perform the installation over the network using the following protocols:
 - **NFS:** The Binary DVD ISO image is in a Network File System (NFS) share.
 - **HTTPS, HTTP or FTP:** The installation tree is on a network location that is accessible over HTTP, HTTPS or FTP.

17.2. PORTS FOR NETWORK-BASED INSTALLATION

The following table lists the ports that must be open on the server providing the files for each type of network-based installation.

Table 17.1. Ports for network-based installation

Protocol used	Ports to open
HTTP	80
HTTPS	443
FTP	21
NFS	2049, 111, 20048
TFTP	69

Additional resources

- See the [Securing networks](#) document for more information.

17.3. CREATING AN INSTALLATION SOURCE ON AN NFS SERVER

Follow the steps in this procedure to place the installation source on an NFS server. Use this installation method to install multiple systems from a single source, without having to connect to physical media.

Prerequisites

- You have administrator level access to a server with Red Hat Enterprise Linux 8, and this server is on the same network as the system to be installed.
- You have downloaded a Binary DVD image. See [Downloading the installation ISO image](#) from the *Performing a standard RHEL installation* document for more information.
- You have created a bootable CD, DVD, or USB device from the image file. See [Creating installation media](#) from the *Performing a standard RHEL installation* document for more information.
- You have verified that your firewall allows the system you are installing to access the remote installation source. See [Ports for network-based installation](#) from the *Performing a standard RHEL installation* document for more information.

Procedure

1. Install the **nfs-utils** package:

```
# yum install nfs-utils
```

2. Copy the Binary DVD ISO image to a directory on the NFS server.
3. Open the **/etc/exports** file using a text editor and add a line with the following syntax:

`/exported_directory/ clients`

4. Replace `/exported_directory/` with the full path to the directory with the ISO image. Replace `clients` with the host name or IP address of the target system, the subnetwork that all target systems can use to access the ISO image, or the asterisk sign (*) if you want to allow any system with network access to the NFS server to use the ISO image. See the **exports(5)** man page for detailed information about the format of this field.

A basic configuration that makes the `/rhel8-install/` directory available as read-only to all clients is:

`/rhel8-install *`

5. Save the **/etc/exports** file and exit the text editor.

6. Start the nfs service:

`# systemctl start nfs-server.service`

If the service was running before you changed the **/etc/exports** file, run the following command for the running NFS server to reload its configuration:

`# systemctl reload nfs-server.service`

The ISO image is now accessible over NFS and ready to be used as an installation source.



NOTE

When configuring the installation source, use **nfs**: as the protocol, the server host name or IP address, the colon sign (:), and the directory holding the ISO image. For example, if the server host name is **myserver.example.com** and you have saved the ISO image in `/rhel8-install/`, specify **nfs:myserver.example.com:/rhel8-install/** as the installation source.

17.4. CREATING AN INSTALLATION SOURCE USING HTTP OR HTTPS

Follow the steps in this procedure to create an installation source for a network-based installation using an installation tree, which is a directory containing extracted contents of the Binary DVD ISO image and a valid **.treeinfo** file. The installation source is accessed over HTTP or HTTPS.

Prerequisites

- You have administrator level access to a server with Red Hat Enterprise Linux 8, and this server is on the same network as the system to be installed.
- You have downloaded a Binary DVD image. See [Downloading the installation ISO image](#) from the *Performing a standard RHEL installation* document for more information.
- You have created a bootable CD, DVD, or USB device from the image file. See [Creating installation media](#) from the *Performing a standard RHEL installation* document for more information.

- You have verified that your firewall allows the system you are installing to access the remote installation source. See [Ports for network-based installation](#) from the *Performing a standard RHEL installation* document for more information.

Procedure

1. Install the **httpd** package:

```
# yum install httpd
```



WARNING

If your Apache web server configuration enables SSL security, verify that you enable only the TLSv1 protocol, and disable SSLv2 and SSLv3. This is due to the POODLE SSL vulnerability (CVE-2014-3566). See <https://access.redhat.com/solutions/1232413> for details.



IMPORTANT

If you use an HTTPS server with a self-signed certificate, you must boot the installation program with the **noverifyssl** option.

2. Copy the Binary DVD ISO image to the HTTP(S) server.
3. Mount the Binary DVD ISO image, using the **mount** command, to a suitable directory:

```
# mkdir /mnt/rhel8-install/
# mount -o loop,ro -t iso9660 /image_directory/image.iso /mnt/rhel8-install/
```

Replace */image_directory/image.iso* with the path to the Binary DVD ISO image.

4. Copy the files from the mounted image to the HTTP(S) server root. This command creates the **/var/www/html/rhel8-install/** directory with the contents of the image.

```
# cp -r /mnt/rhel8-install/ /var/www/html/
```

This command creates the **/var/www/html/rhel8-install/** directory with the content of the image. Note that some copying methods can skip the **.treeinfo** file which is required for a valid installation source. Running the **cp** command for whole directories as shown in this procedure will copy **.treeinfo** correctly.

5. Start the **httpd** service:

```
# systemctl start httpd.service
```

The installation tree is now accessible and ready to be used as the installation source.



NOTE

When configuring the installation source, use **http://** or **https://** as the protocol, the server host name or IP address, and the directory that contains the files from the ISO image, relative to the HTTP server root. For example, if you are using HTTP, the server host name is **myserver.example.com**, and you have copied the files from the image to **/var/www/html/rhel8-install/**, specify **http://myserver.example.com/rhel8-install/** as the installation source.

Additional resources

- For more information about HTTP servers, see the [Deploying different types of servers](#) document.

17.5. CREATING AN INSTALLATION SOURCE USING FTP

Follow the steps in this procedure to create an installation source for a network-based installation using an installation tree, which is a directory containing extracted contents of the Binary DVD ISO image and a valid **.treeinfo** file. The installation source is accessed over FTP.

Prerequisites

- You have administrator level access to a server with Red Hat Enterprise Linux 8, and this server is on the same network as the system to be installed.
- You have downloaded a Binary DVD image. See [Downloading the installation ISO image](#) from the *Performing a standard RHEL installation* document for more information.
- You have created a bootable CD, DVD, or USB device from the image file. See [Creating installation media](#) from the *Performing a standard RHEL installation* document for more information.
- You have verified that your firewall allows the system you are installing to access the remote installation source. See [Ports for network-based installation](#) from the *Performing a standard RHEL installation* document for more information.

Procedure

- Install the **vsftpd** package by running the following command as root:


```
# yum install vsftpd
```
- Open and edit the **/etc/vsftpd/vsftpd.conf** configuration file in a text editor.
 - Change the line **anonymous_enable=NO** to **anonymous_enable=YES**
 - Change the line **write_enable=YES** to **write_enable=NO**.
 - Add lines **pasv_min_port=min_port** and **pasv_max_port=max_port**. Replace *min_port* and *max_port* with the port number range used by FTP server in passive mode, e. g. **10021** and **10031**.
This step can be necessary in network environments featuring various firewall/NAT setups.
 - Optionally, add custom changes to your configuration. For available options, see the **vsftpd.conf(5)** man page. This procedure assumes that default options are used.



WARNING

If you configured SSL/TLS security in your **vsftpd.conf** file, ensure that you enable only the TLSv1 protocol, and disable SSLv2 and SSLv3. This is due to the POODLE SSL vulnerability (CVE-2014-3566). See <https://access.redhat.com/solutions/1234773> for details.

3. Configure the server firewall.

- a. Enable the firewall:

```
# systemctl enable firewalld
# systemctl start firewalld
```

- b. Enable in your firewall the FTP port and port range from previous step:

```
# firewall-cmd --add-port min_port-max_port/tcp --permanent
# firewall-cmd --add-service ftp --permanent
# firewall-cmd --reload
```

Replace *min_port-max_port* with the port numbers you entered into the **/etc/vsftpd/vsftpd.conf** configuration file.

4. Copy the Binary DVD ISO image to the FTP server.

5. Mount the Binary DVD ISO image, using the mount command, to a suitable directory:

```
# mkdir /mnt/rhel8-install
# mount -o loop,ro -t iso9660 /image-directory/image.iso /mnt/rhel8-install
```

Replace */image-directory/image.iso* with the path to the Binary DVD ISO image.

6. Copy the files from the mounted image to the FTP server root:

```
# mkdir /var/ftp/rhel8-install
# cp -r /mnt/rhel8-install/ /var/ftp/
```

This command creates the **/var/ftp/rhel8-install/** directory with the content of the image. Note that some copying methods can skip the **.treeinfo** file which is required for a valid installation source. Running the **cp** command for whole directories as shown in this procedure will copy **.treeinfo** correctly.

7. Make sure that the correct SELinux context and access mode is set on the copied content:

```
# restorecon -r /var/ftp/rhel8-install
# find /var/ftp/rhel8-install -type f -exec chmod 444 {} \;
# find /var/ftp/rhel8-install -type d -exec chmod 755 {} \;
```

8. Start the **vsftpd** service:

```
# systemctl start vsftpd.service
```

If the service was running before you changed the `/etc/vsftpd/vsftpd.conf` file, restart the service to load the edited file:

```
# systemctl restart vsftpd.service
```

Enable the **vsftpd** service to start during the boot process:

```
# systemctl enable vsftpd
```

The installation tree is now accessible and ready to be used as the installation source.



NOTE

When configuring the installation source, use **ftp://** as the protocol, the server host name or IP address, and the directory in which you have stored the files from the ISO image, relative to the FTP server root. For example, if the server host name is **myserver.example.com** and you have copied the files from the image to **/var/ftp/rhel8-install/**, specify **ftp://myserver.example.com/rhel8-install/** as the installation source.

CHAPTER 18. STARTING KICKSTART INSTALLATIONS

You can start Kickstart installations in multiple ways:

- Manually by entering the installation program boot menu and specifying the options including Kickstart file there.
- Automatically by editing the boot options in PXE boot.
- Automatically by providing the file on a volume with specific name.

Learn how to perform each of these methods in the following sections.

18.1. STARTING A KICKSTART INSTALLATION MANUALLY

This section explains how to start a Kickstart installation manually, which means some user interaction is required (adding boot options at the **boot:** prompt). Use the boot option **inst.ks=location** when booting the installation system, replacing location with the location of your Kickstart file. The exact way to specify the boot option depends on your system's architecture.

Prerequisites

- You have a Kickstart file ready in a location accessible from the system to be installed

Procedure

1. Boot the system using a local media (a CD, DVD, or a USB flash drive).
2. At the boot prompt, specify the required boot options.
 - a. If the Kickstart file or a required repository is in a network location, you may need to configure the network using the **ip=** option. The installer tries to configure all network devices using the DHCP protocol by default without this option.
 - b. Add the **inst.ks=** boot option and the location of the Kickstart file.
 - c. In order to access a software source from which necessary packages will be installed, you may need to add the **inst.repo=** option. If you do not specify this option, you must specify the installation source in the Kickstart file.
3. Start the installation by confirming your added boot options.
The installation begins now, using the options specified in the Kickstart file. If the Kickstart file is valid and contains all required commands, the installation is completely automated from this point forward.

18.2. STARTING A KICKSTART INSTALLATION AUTOMATICALLY USING PXE

AMD64, Intel 64, and 64-bit ARM systems and IBM Power Systems servers have the ability to boot using a PXE server. When you configure the PXE server, you can add the boot option into the boot loader configuration file, which in turn lets you start the installation automatically. Using this approach, it is possible to automate the installation completely, including the boot process.

This procedure is intended as a general reference; detailed steps differ based on your system's architecture, and not all options are available on all architectures (for example, you cannot use PXE boot on IBM Z).

Prerequisites

- You must have a Kickstart file ready in a location accessible from the system to be installed.
- You must have a PXE server which can be used to boot the system and begin the installation.

Procedure

1. Open the boot loader configuration file on your PXE server, and add the **inst.ks=** boot option to the appropriate line. The name of the file and its syntax depends on your system's architecture and hardware:

- On AMD64 and Intel 64 systems with BIOS, the file name can be either default or based on your system's IP address. In this case, add the **inst.ks=** option to the append line in the installation entry. A sample append line in the configuration file looks similar to the following:

```
append initrd=initrd.img inst.ks=http://10.32.5.1/mnt/archive/RHEL-8.8.x/x86_64/kickstarts/ks.cfg
```

- On systems using the GRUB2 boot loader (AMD64, Intel 64, and 64-bit ARM systems with UEFI firmware and IBM Power Systems servers), the file name will be **grub.cfg**. In this file, append the **inst.ks=** option to the kernel line in the installation entry. A sample kernel line in the configuration file will look similar to the following:

```
kernel vmlinuz inst.ks=http://10.32.5.1/mnt/archive/RHEL-8.8.x/x86_64/kickstarts/ks.cfg
```

2. Boot the installation from the network server.

The installation begins now, using the installation options specified in the Kickstart file. If the Kickstart file is valid and contains all required commands, the installation is completely automated.

18.3. STARTING A KICKSTART INSTALLATION AUTOMATICALLY USING A LOCAL VOLUME

You can start a Kickstart installation by putting a Kickstart file with a specific name on a specifically labelled storage volume.

Prerequisites

- You must have a volume prepared with label **OEMDRV** and the Kickstart file present in its root as **ks.cfg**.
- A drive containing this volume must be available on the system as the installation program boots.

Procedure

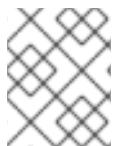
1. Boot the system using a local media (a CD, DVD, or a USB flash drive).

2. At the boot prompt, specify the required boot options.
 - a. If a required repository is in a network location, you may need to configure the network using the **ip=** option. The installer tries to configure all network devices using the DHCP protocol by default without this option.
 - b. In order to access a software source from which necessary packages will be installed, you may need to add the **inst.repo=** option. If you do not specify this option, you must specify the installation source in the Kickstart file.
3. Start the installation by confirming your added boot options.

The installation begins now, and the Kickstart file is automatically detected and used to start an automated Kickstart installation.

CHAPTER 19. CONSOLES AND LOGGING DURING INSTALLATION

The Red Hat Enterprise Linux installer uses the **tmux** terminal multiplexer to display and control several windows in addition to the main interface. Each of these windows serve a different purpose; they display several different logs, which can be used to troubleshoot issues during the installation process. One of the windows provides an interactive shell prompt with **root** privileges, unless this prompt was specifically disabled using a boot option or a Kickstart command.



NOTE

In general, there is no reason to leave the default graphical installation environment unless you need to diagnose an installation problem.

The terminal multiplexer is running in virtual console 1. To switch from the actual installation environment to **tmux**, press **Ctrl+Alt+F1**. To go back to the main installation interface which runs in virtual console 6, press **Ctrl+Alt+F6**.



NOTE

If you choose text mode installation, you will start in virtual console 1 (**tmux**), and switching to console 6 will open a shell prompt instead of a graphical interface.

The console running **tmux** has five available windows; their contents are described in the following table, along with keyboard shortcuts. Note that the keyboard shortcuts are two-part: first press **Ctrl+b**, then release both keys, and press the number key for the window you want to use.

You can also use **Ctrl+b n**, **Alt+Tab**, and **Ctrl+b p** to switch to the next or previous **tmux** window, respectively.

Table 19.1. Available **tmux** windows

Shortcut	Contents
Ctrl+b 1	Main installation program window. Contains text-based prompts (during text mode installation or if you use VNC direct mode), and also some debugging information.
Ctrl+b 2	Interactive shell prompt with root privileges.
Ctrl+b 3	Installation log; displays messages stored in /tmp/anaconda.log .
Ctrl+b 4	Storage log; displays messages related to storage devices and configuration, stored in /tmp/storage.log .
Ctrl+b 5	Program log; displays messages from utilities executed during the installation process, stored in /tmp/program.log .

CHAPTER 20. MAINTAINING KICKSTART FILES

You can run automated checks on Kickstart files. Typically, you will want to verify that a new or problematic Kickstart file is valid.

20.1. INSTALLING KICKSTART MAINTENANCE TOOLS

To use the Kickstart maintenance tools, you must install the package that contains them.

Procedure

- Install the **pykickstart** package:

```
# yum install pykickstart
```

20.2. VERIFYING A KICKSTART FILE

Use the **ksvalidator** command line utility to verify that your Kickstart file is valid. This is useful when you make extensive changes to a Kickstart file.

Procedure

- Run **ksvalidator** on your Kickstart file:

```
$ ksvalidator /path/to/kickstart.ks
```

Replace */path/to/kickstart.ks* with the path to the Kickstart file you want to verify.



IMPORTANT

The validation tool cannot guarantee the installation will be successful. It ensures only that the syntax is correct and that the file does not include deprecated options. It does not attempt to validate the **%pre**, **%post** and **%packages** sections of the Kickstart file.

Additional resources

- The *ksvalidator(1)* manual page.

CHAPTER 21. REGISTERING AND INSTALLING RHEL FROM THE CDN USING KICKSTART

This section contains information about how to register your system, attach RHEL subscriptions, and install from the Red Hat Content Delivery Network (CDN) using Kickstart.

21.1. REGISTERING AND INSTALLING RHEL FROM THE CDN

Use this procedure to register your system, attach RHEL subscriptions, and install from the Red Hat Content Delivery Network (CDN) using the **rhsm** Kickstart command, which supports the **syspurpose** command as well as Red Hat Insights. The **rhsm** Kickstart command removes the requirement of using custom **%post** scripts when registering the system.



IMPORTANT

The CDN feature is supported by the **Boot ISO** and **Binary DVD ISO** image files. However, it is recommended that you use the **Boot ISO** image file as the installation source defaults to CDN for the Boot ISO image file.

Prerequisites

- Your system is connected to a network that can access the CDN.
- You created a Kickstart file and made it available to the installation program on removable media, a hard drive or a network location using an HTTP(S), FTP, or NFS server.
- The Kickstart file is in a location that is accessible by the system that is to be installed.
- You created the boot media used to begin the installation and made the installation source available to the installation program.



IMPORTANT

- The installation source repository used after system registration is dependent on how the system was booted. For more information, see the *Installation source repository after system registration* section in the [Performing a standard RHEL installation](#) document.
- Repository configuration is not required in a Kickstart file as your subscription governs which CDN subset and repositories the system can access.

Procedure

1. Open the Kickstart file.
2. Edit the file to add the **rhsm** Kickstart command and its options to the file:

Organization (required)

Enter the organization id. An example is:

```
--organization=1234567
```

**NOTE**

For security reasons, Red Hat username and password account details are not supported by Kickstart when registering and installing from the CDN.

Activation Key (required)

Enter the Activation Key. You can enter multiple keys as long as the activation keys are registered to your subscription. An example is:

```
--activation-key="Test_key_1" --activation-key="Test_key_2"
```

Red Hat Insights (optional)

Connect the target system to Red Hat Insights.

**NOTE**

Red Hat Insights is a Software-as-a-Service (SaaS) offering that provides continuous, in-depth analysis of registered Red Hat-based systems to proactively identify threats to security, performance and stability across physical, virtual and cloud environments, and container deployments. Unlike the GUI setting, connecting to Red Hat Insights is not enabled by default when using Kickstart.

An example is:

```
--connect-to-insights
```

HTTP proxy (optional)

Set the HTTP proxy. An example is:

```
--proxy="user:password@hostname:9000"
```

**NOTE**

Only the hostname is mandatory. If the proxy is required to run on a default port with no authentication, then the option is: **--proxy="hostname"**

Server hostname (optional)**NOTE**

The Server hostname does not require the HTTP protocol, for example, **nameofhost.com**.

Set the server hostname if you are running Satellite Server or performing internal testing. An example is:

```
--server-hostname="nameofhost.com"
```

rhsm baseurl (optional)



NOTE

The rhsm baseurl does require the HTTP protocol, for example, <http://nameofhost.com>.

Set the rhsm baseurl option if you are running Satellite Server or performing internal testing. An example is:

```
--rhsm-baseurl="http://nameofhost.com"
```

System Purpose (optional)

Set the System Purpose role, SLA, and usage using the command:

```
syspurpose --role="Red Hat Enterprise Linux Server" --sla="Premium" --usage="Production"
```

Example

The following example displays a minimal Kickstart file with all **rhsm** Kickstart command options.

```
graphical
lang en_US.UTF-8
keyboard us
rootpw 12345
timezone America/New_York
zerombr
clearpart --all --initlabel
autopart
syspurpose --role="Red Hat Enterprise Linux Server" --sla="Premium" --usage="Production"
rhsm --organization="12345" --activation-key="test_key" --connect-to-insights --server-hostname="nameofhost.com"
--rhsm-baseurl="http://nameofhost.com" --proxy="user:password@hostname:9000"
%packages
vim
%end
```

3. Save the Kickstart file and start the installation process.

Additional resources

- For more information about System Purpose, see the *Configuring System Purpose* section of this document.
- For more information about how to start a Kickstart installation, see *Starting Kickstart installations*.
- For information about Red Hat Insights, see the *Red Hat Insights product documentation*.

- For information about Activation Keys, see the [Understanding Activation Keys](#) chapter of the *Using Red Hat Subscription Management* document.
- For information about how to set up an HTTP proxy for Subscription Manager, see the [Using an HTTP proxy](#) chapter of the *Using and Configuring Red Hat Subscription Manager* document.

21.2. VERIFYING YOUR SYSTEM REGISTRATION FROM THE CDN

Use this procedure to verify that your system is registered to the CDN.

Prerequisites

- You have completed the registration and installation process as documented in [Section 21.1, “Registering and installing RHEL from the CDN”](#).
- You have started the Kickstart installation as documented in [Starting Kickstart installations](#).
- The installed system has rebooted and a terminal window is open.

Procedure

1. From the terminal window, log in as a **root** user and verify the registration:

```
# subscription-manager list
```

The output displays the attached subscription details, for example:

```
Installed Product Status
```

```
Product Name: Red Hat Enterprise Linux for x86_64
Product ID: 486
Version: 8.2
Arch: x86_64
Status: Subscribed
Status Details
Starts: 11/4/2019
Ends: 11/4/2020
```

2. To view a detailed report, run the command:

```
# subscription-manager list --consumed
```

21.3. UNREGISTERING YOUR SYSTEM FROM THE CDN

Use this procedure to unregister your system from the Red Hat CDN.

Prerequisites

- You have completed the registration and installation process as documented in [Section 21.1, “Registering and installing RHEL from the CDN”](#).
- You have started the Kickstart installation as documented in [Starting Kickstart installations](#).
- The installed system has rebooted and a terminal window is open.

Procedure

1. From the terminal window, log in as a **root** user and unregister:

```
# subscription-manager unregister
```

The attached subscription is unregistered from the system and the connection to CDN is removed.

CHAPTER 22. PERFORMING A REMOTE RHEL INSTALLATION USING VNC

This section describes how to perform a remote RHEL installation using Virtual Network Computing (VNC).

22.1. OVERVIEW

The graphical user interface is the recommended method of installing RHEL when you boot the system from a CD, DVD, or USB flash drive, or from a network using PXE. However, many enterprise systems, for example, IBM Power Systems and IBM Z, are located in remote data center environments that are run autonomously and are not connected to a display, keyboard, and mouse. These systems are often referred to as *headless systems* and they are typically controlled over a network connection. The RHEL installation program includes a Virtual Network Computing (VNC) installation that runs the graphical installation on the target machine, but control of the graphical installation is handled by another system on the network. The RHEL installation program offers two VNC installation modes: **Direct** and **Connect**. Once a connection is established, the two modes do not differ. The mode you select depends on your environment.

Direct mode

In Direct mode, the RHEL installation program is configured to start on the target system and wait for a VNC viewer that is installed on another system before proceeding. As part of the Direct mode installation, the IP address and port are displayed on the target system. You can use the VNC viewer to connect to the target system remotely using the IP address and port, and complete the graphical installation.

Connect mode

In Connect mode, the VNC viewer is started on a remote system in *listening* mode. The VNC viewer waits for an incoming connection from the target system on a specified port. When the RHEL installation program starts on the target system, the system host name and port number are provided by using a boot option or a Kickstart command. The installation program then establishes a connection with the listening VNC viewer using the specified system host name and port number. To use Connect mode, the system with the listening VNC viewer must be able to accept incoming network connections.

22.2. CONSIDERATIONS

Consider the following items when performing a remote RHEL installation using VNC:

- **VNC client application:** A VNC client application is required to perform both a VNC Direct and Connect installation. VNC client applications are available in the repositories of most Linux distributions, and free VNC client applications are also available for other operating systems such as Windows. The following VNC client applications are available in RHEL:
 - **tigervnc** is independent of your desktop environment and is installed as part of the **tigervnc** package.
 - **vinagre** is part of the GNOME desktop environment and is installed as part of the **vinagre** package.



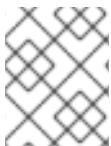
NOTE

A VNC server is included in the installation program and doesn't need to be installed.

- **Network and firewall:**
 - If the target system is not allowed inbound connections by a firewall, then you must use Connect mode or disable the firewall. Disabling a firewall can have security implications.
 - If the system that is running the VNC viewer is not allowed incoming connections by a firewall, then you must use Direct mode, or disable the firewall. Disabling a firewall can have security implications. See the [Security hardening](#) document for more information on configuring the firewall.
- **Custom Boot Options:** You must specify custom boot options to start a VNC installation and the installation instructions might differ depending on your system architecture.
- **VNC in Kickstart installations:** You can use VNC-specific commands in Kickstart installations. Using only the **vnc** command runs a RHEL installation in Direct mode. Additional options are available to set up an installation using Connect mode. For more information about Kickstart installations, see [Section 14.1, "What are Kickstart installations"](#).

22.3. PERFORMING A REMOTE RHEL INSTALLATION IN VNC DIRECT MODE

Use this procedure to perform a remote RHEL installation in VNC Direct mode. Direct mode expects the VNC viewer to initiate a connection to the target system that is being installed with RHEL. In this procedure, the system with the VNC viewer is called the **remote** system. You are prompted by the RHEL installation program to initiate the connection from the VNC viewer on the remote system to the target system.



NOTE

This procedure uses **TigerVNC** as the VNC viewer. Specific instructions for other viewers might differ, but the general principles apply.

Prerequisites

- As root, you have installed a VNC viewer on a remote system, for example:


```
# yum install tigervnc
```
- You have set up a network boot server and booted the installation on the target system. For more information, see [Section 26.1, "Network install overview"](#).

Procedure

1. From the RHEL boot menu on the target system, press the **Tab** key on your keyboard to edit the boot options.
2. Append the **inst.vnc** option to the end of the command line.
 - a. If you want to restrict VNC access to the system that is being installed, add the **inst.vncpassword=PASSWORD** boot option to the end of the command line. Replace **PASSWORD** with the password you want to use for the installation. The VNC password must be between 6 and 8 characters long.



IMPORTANT

Use a temporary password for the **inst.vncpassword=** option. It should not be an existing or root password.

3. Press **Enter** to start the installation. The target system initializes the installation program and starts the necessary services. When the system is ready, a message is displayed providing the IP address and port number of the system.
4. Open the VNC viewer on the remote system.
5. Enter the IP address and the port number into the **VNC server** field.
6. Click **Connect**.
7. Enter the VNC password and click **OK**. A new window opens with the VNC connection established, displaying the RHEL installation menu. From this window, you can install RHEL on the target system using the graphical user interface.

Additional resources

- For more information on how to perform a RHEL installation using the graphical user interface, see the *Installing RHEL using the Graphical User Interface* section in the [Performing a standard RHEL installation](#) document.

22.4. PERFORMING A REMOTE RHEL INSTALLATION IN VNC CONNECT MODE

Use this procedure to perform a remote RHEL installation in VNC Connect mode. In Connect mode, the target system that is being installed with RHEL initiates a connect to the VNC viewer that is installed on another system. In this procedure, the system with the VNC viewer is called the **remote** system.



NOTE

This procedure uses **TigerVNC** as the VNC viewer. Specific instructions for other viewers might differ, but the general principles apply.

Prerequisites

- As root, you have installed a VNC viewer on a remote system, for example:

```
# yum install tigervnc
```
- You have set up a network boot server to start the installation on the target system. For more information, see [Section 26.1, “Network install overview”](#).
- You have configured the target system to use the boot options for a VNC Connect installation.
- You have verified that the remote system with the VNC viewer is configured to accept an incoming connection on the required port. Verification is dependent on your network and system configuration. For more information, see the [Security hardening](#) and [Securing networks](#) documents.

Procedure

1. Start the VNC viewer on the remote system in *listening mode* by running the following command:

```
$ vncviewer -listen PORT
```

2. Replace PORT with the port number used for the connection.
3. The terminal displays a message indicating that it is waiting for an incoming connection from the target system.

```
TigerVNC Viewer 64-bit v1.8.0
Built on: 2017-10-12 09:20
Copyright (C) 1999-2017 TigerVNC Team and many others (see README.txt)
See http://www.tigervnc.org for information on TigerVNC.
```

```
Thu Jun 27 11:30:57 2019
main: Listening on port 5500
```

4. Boot the target system from the network.
5. From the RHEL boot menu on the target system, press the **Tab** key on your keyboard to edit the boot options.
6. Append the **inst.vnc inst.vncconnect=HOST:PORT** option to the end of the command line.
7. Replace *HOST* with the IP address of the remote system that is running the listening VNC viewer, and *PORT* with the port number that the VNC viewer is listening on.
8. Press **Enter** to start the installation. The system initializes the installation program and starts the necessary services. When the initialization process is finished, the installation program attempts to connect to the IP address and port provided.
9. When the connection is successful, a new window opens with the VNC connection established, displaying the RHEL installation menu. From this window, you can install RHEL on the target system using the graphical user interface.

Additional resources

- For more information on how to perform a RHEL installation using the graphical user interface, see the *Installing RHEL using the Graphical User Interface* section in the [Performing a standard RHEL installation](#) document.

CHAPTER 23. ADVANCED CONFIGURATION OPTIONS

CHAPTER 24. CONFIGURING SYSTEM PURPOSE

You use System Purpose to record the intended use of a Red Hat Enterprise Linux 8 system. Setting System Purpose enables the entitlement server to auto-attach the most appropriate subscription. This section describes how to configure System Purpose using Kickstart.

Benefits include:

- In-depth system-level information for system administrators and business operations.
- Reduced overhead when determining why a system was procured and its intended purpose.
- Improved customer experience of Subscription Manager auto-attach as well as automated discovery and reconciliation of system usage.

24.1. OVERVIEW

You can enter System Purpose data in one of the following ways:

- During image creation
- During a GUI installation when using **Connect to Red Hat** to register your system and attach your Red Hat subscription
- During a Kickstart installation when using Kickstart automation scripts
- After installation using the **syspurpose** command-line (CLI) tool

To record the intended purpose of your system, you can configure the following components of System Purpose. The selected values are used by the entitlement server upon registration to attach the most suitable subscription for your system.

- **Role**
 - Red Hat Enterprise Linux Server
 - Red Hat Enterprise Linux Workstation
 - Red Hat Enterprise Linux Compute Node
- **Service Level Agreement**
 - Premium
 - Standard
 - Self-Support
- **Usage**
 - Production
 - Development/Test
 - Disaster Recovery

Additional resources

- For more information about Image Builder, see the [Composing a customized RHEL system image](#) document.
- For more information about Kickstart, see the [Performing an advanced RHEL installation](#) document.
- For more information about Subscription Manager, see the [Using and Configuring Red Hat Subscription Manager](#) document.

24.2. CONFIGURING SYSTEM PURPOSE IN A KICKSTART FILE

Follow the steps in this procedure to use the **syspurpose** command to configure System Purpose in a Kickstart configuration file.



NOTE

While it is strongly recommended that you configure System Purpose, it is an optional feature of the Red Hat Enterprise Linux installation program. If you want to enable System Purpose after the installation completes, you can do so using the **syspurpose** command-line tool.

The following actions are available:

role

Set the intended role of the system. This action uses the following format:

```
syspurpose --role=
```

The assigned role can be:

- **Red Hat Enterprise Linux Server**
- **Red Hat Enterprise Linux Workstation**
- **Red Hat Enterprise Linux Compute Node**

SLA

Set the intended SLA of the system. This action uses the following format:

```
syspurpose --sla=
```

The assigned sla can be:

- **Premium**
- **Standard**
- **Self-Support**

usage

Set the intended usage of the system. This action uses the following format:

```
syspurpose --usage=
```

The assigned usage can be:

- **Production**
- **Development/Test**
- **Disaster Recovery**

24.3. RELATED INFORMATION

- For information about configuring System Purpose using the graphical user interface, or the **syspurpose** command-line utility, see the [*Performing a standard RHEL installation*](#) document.

CHAPTER 25. UPDATING DRIVERS DURING INSTALLATION

This section describes how to complete a driver update during the Red Hat Enterprise Linux installation process.



NOTE

This is an optional step of the installation process. Red Hat recommends that you do not perform a driver update unless it is necessary.

25.1. PREREQUISITE

You have been notified by Red Hat, your hardware vendor, or a trusted third-party vendor that a driver update is required during Red Hat Enterprise Linux installation.

25.2. OVERVIEW

Red Hat Enterprise Linux supports drivers for many hardware devices but some newly-released drivers may not be supported. A driver update should only be performed if an unsupported driver prevents the installation from completing. Updating drivers during installation is typically only required to support a particular configuration. For example, installing drivers for a storage adapter card that provides access to your system's storage devices.



WARNING

Driver update disks may disable conflicting kernel drivers. In rare cases, unloading a kernel module may cause installation errors.

25.3. TYPES OF DRIVER UPDATE

Red Hat, your hardware vendor, or a trusted third party provides the driver update as an ISO image file. Once you receive the ISO image file, choose the type of driver update.

Types of driver update

Automatic

The recommended driver update method; a storage device (including a CD, DVD, or USB flash drive) labeled **OEMDRV** is physically connected to the system. If the **OEMDRV** storage device is present when the installation starts, it is treated as a driver update disk, and the installation program automatically loads its drivers.

Assisted

The installation program prompts you to locate a driver update. You can use any local storage device with a label other than **OEMDRV**. The **inst.dd** boot option is specified when starting the installation. If you use this option without any parameters, the installation program displays all of the storage devices connected to the system, and prompts you to select a device that contains a driver update.

Manual

Manually specify a path to a driver update image or an RPM package. You can use any local storage

device with a label other than **OEMDRV**, or a network location accessible from the installation system. The **inst.dd=location** boot option is specified when starting the installation, where *location* is the path to a driver update disk or ISO image. When you specify this option, the installation program attempts to load any driver updates found at the specified location. With manual driver updates, you can specify local storage devices, or a network location (HTTP, HTTPS or FTP server).



NOTE

- You can use both **inst.dd=location** and **inst.dd** simultaneously, where *location* is the path to a driver update disk or ISO image. In this scenario, the installation program attempts to load any available driver updates from the location and also prompts you to select a device that contains the driver update.
- Initialize the network using the **ip= option** when loading a driver update from a network location.

Limitations

On UEFI systems with the Secure Boot technology enabled, all drivers must be signed with a valid certificate. Red Hat drivers are signed by one of Red Hat's private keys and authenticated by its corresponding public key in the kernel. If you load additional, separate drivers, verify that they are signed.

25.4. PREPARING A DRIVER UPDATE

This procedure describes how to prepare a driver update on a CD and DVD.

Prerequisites

- You received the driver update ISO image from Red Hat, your hardware vendor, or a trusted third-party vendor.
- You burned the driver update ISO image to a CD or DVD.



WARNING

If only a single ISO image file ending in **.iso** is available on the CD or DVD, the burn process has not been successful. See your system's burning software documentation for instructions on how to burn ISO images to a CD or DVD.

Procedure

1. Insert the driver update CD or DVD into your system's CD/DVD drive, and browse it using the system's file manager tool.
2. Verify that a single file **rhdd3** is available. **rhdd3** is a signature file that contains the driver description and a directory named **rpms**, which contains the RPM packages with the actual drivers for various architectures.

25.5. PERFORMING AN AUTOMATIC DRIVER UPDATE

This procedure describes how to perform an automatic driver update during installation.

Prerequisites

- You have placed the driver update image on a standard disk partition with an **OEMDRV** label or burnt the **OEMDRV** driver update image to a CD or DVD. Advanced storage, such as RAID or LVM volumes, may not be accessible during the driver update process.
- You have connected a block device with an **OEMDRV** volume label to your system, or inserted the prepared CD or DVD into your system's CD/DVD drive before starting the installation process.

Procedure

1. Once you have completed the prerequisite steps, the drivers are automatically loaded when the installation program starts, and installed on the system during the installation process.

25.6. PERFORMING AN ASSISTED DRIVER UPDATE

This procedure describes how to perform an assisted driver update during installation.

Prerequisites

You have connected a block device without an **OEMDRV** volume label to your system and copied the driver disk image to this device, or you have prepared a driver update CD or DVD and inserted it into your system's CD/DVD drive before starting the installation process.



NOTE

If you burned an ISO image file to a CD or DVD but it does not have the **OEMDRV** volume label, you can use the **inst.dd** option with no arguments. The installation program provides an option to scan and select drivers from the CD or DVD. In this scenario, the installation program does not prompt you to select a driver update ISO image. Another scenario is to use the CD or DVD with the **inst.dd=location** boot option; this allows the installation program to automatically scan the CD or DVD for driver updates. For more information, see [Section 25.7, "Performing a manual driver update"](#).

Procedure

1. From the boot menu window, press the **Tab** key on your keyboard to display the boot command line.
2. Append the **inst.dd** boot option to the command line and press **Enter** to execute the boot process.
3. From the menu, select a local disk partition or a CD or DVD device. The installation program scans for ISO files, or driver update RPM packages.
4. Optional: Select the driver update ISO file.



NOTE

This step is not required if the selected device or partition contains driver update RPM packages rather than an ISO image file, for example, an optical drive containing a driver update CD or DVD.

5. Select the required drivers.
 - a. Use the number keys on your keyboard to toggle the driver selection.
 - b. Press **c** to install the selected driver. The selected driver is loaded and the installation process starts.

25.7. PERFORMING A MANUAL DRIVER UPDATE

This procedure describes how to perform a manual driver update during installation.

Prerequisites

- Place the driver update ISO image file on a USB flash drive or a web server, and connect it to your computer.

Procedure

1. From the boot menu window, press the **Tab** key on your keyboard to display the boot command line.
2. Append the **inst.dd=location** boot option to the command line, where location is a path to the driver update. Typically, the image file is located on a web server, for example, <http://server.example.com/dd.iso>, or on a USB flash drive, for example, [/dev/sdb1](#). It is also possible to specify an RPM package containing the driver update, for example <http://server.example.com/dd.rpm>.
3. Press **Enter** to execute the boot process. The drivers available at the specified location are automatically loaded and the installation process starts.

Additional resources

- For more information about the **inst.dd** boot option, see the upstream [inst.dd boot option](#) content.
- For more information about all boot options, see the upstream [Boot Options](#) content.

25.8. DISABLING A DRIVER

This procedure describes how to disable a malfunctioning driver.

Prerequisites

- You have booted the installation program boot menu.

Procedure

1. From the boot menu, press the **Tab** key on your keyboard to display the boot command line.

2. Append the **modprobe.blacklist=driver_name** boot option to the command line.
3. Replace *driver_name* with the name of the driver or drivers you want to disable, for example:

```
modprobe.blacklist=ahci
```

Drivers disabled using the **modprobe.blacklist=** boot option remain disabled on the installed system and appear in the **/etc/modprobe.d/anaconda-blacklist.conf** file.

4. Press **Enter** to execute the boot process.

CHAPTER 26. PREPARING TO INSTALL FROM THE NETWORK USING PXE

This section describes how to configure TFTP and DHCP on a PXE server to enable PXE boot and network installation.

26.1. NETWORK INSTALL OVERVIEW

A network installation allows you to install Red Hat Enterprise Linux to a system that has access to an installation server. At a minimum, two systems are required for a network installation:

PXE Server: A system running a DHCP server, a TFTP server, and an HTTP, HTTPS, FTP, or NFS server. While each server can run on a different physical system, the procedures in this section assume a single system is running all servers.

Client: The system to which you are installing Red Hat Enterprise Linux. Once installation starts, the client queries the DHCP server, receives the boot files from the TFTP server, and downloads the installation image from the HTTP, HTTPS, FTP or NFS server. Unlike other installation methods, the client does not require any physical boot media for the installation to start.



NOTE

To boot a client from the network, configure it in BIOS/UEFI or a quick boot menu. On some hardware, the option to boot from a network might be disabled, or not available.

The workflow steps to prepare to install Red Hat Enterprise Linux from a network using PXE are as follows:

Steps

1. Export the installation ISO image or the installation tree to an NFS, HTTPS, HTTP, or FTP server.
2. Configure the TFTP server and DHCP server, and start the TFTP service on the PXE server.
3. Boot the client and start the installation.



IMPORTANT

The GRUB2 boot loader supports a network boot from HTTP in addition to a TFTP server. Sending the boot files, which are the kernel and initial RAM disk **vmlinuz** and **initrd**, over this protocol might be slow and result in timeout failures. An HTTP server does not carry this risk, but it is recommended that you use a TFTP server when sending the boot files.

Additional resources

- To export the installation ISO image to a network location, see [Chapter 17, Creating installation sources for Kickstart installations](#) for information.
- To configure the TFTP server and DHCP server, and start the TFTP service, see [Section 26.2, "Configuring a TFTP server for BIOS-based clients"](#), [Section 26.3, "Configuring a TFTP server for UEFI-based clients"](#), and [Section 26.4, "Configuring a network server for IBM Power](#)

[systems](#) for information.

- Red Hat Satellite can automate the setup of a PXE server. For more information, see the Red Hat Satellite product documentation.

26.2. CONFIGURING A TFTP SERVER FOR BIOS-BASED CLIENTS

Use this procedure to configure a TFTP server and DHCP server and start the TFTP service on the PXE server for BIOS-based AMD and Intel 64-bit systems.



IMPORTANT

All configuration files in this section are examples. Configuration details vary and are dependent on the architecture and specific requirements.

Procedure

1. As root, install the following packages:

```
# yum install tftp-server dhcp-server xinetd
```

2. Allow incoming connections to the **tftp service** in the firewall:

```
# firewall-cmd --add-service=tftp
```



NOTE

- This command enables temporary access until the next server reboot. To enable permanent access, add the **--permanent** option to the command.
- Depending on the location of the installation ISO file, you might have to allow incoming connections for HTTP or other services.

3. Configure your DHCP server to use the boot images packaged with **SYSLINUX** as shown in the following example **/etc/dhcp/dhcpd.conf** file:

```
option space pxelinux;
option pxelinux.magic code 208 = string;
option pxelinux.configfile code 209 = text;
option pxelinux.pathprefix code 210 = text;
option pxelinux.reboottime code 211 = unsigned integer 32;
option architecture-type code 93 = unsigned integer 16;

subnet 10.0.0.0 netmask 255.255.255.0 {
    option routers 10.0.0.254;
    range 10.0.0.2 10.0.0.253;

    class "pxeclients" {
        match if substring (option vendor-class-identifier, 0, 9) = "PXEClient";
        next-server 10.0.0.1;

        if option architecture-type = 00:07 {
            filename "uefi/shim.efd";
```

```
    } else {
        filename "pxelinux/pxelinux.0";
    }
}
```

4. Access the **pxelinux.0** file from the **SYSLINUX** package in the Binary DVD ISO image file, where *my_local_directory* is the name of the directory that you create:

```
# mount -t iso9660 /path_to_image/name_of_image.iso /mount_point -o loop,ro  
  
# cp -pr /mount_point/BaseOS/Packages/syslinux-tftpboot-version-architecture.rpm  
/my_local_directory  
  
# umount /mount_point
```

- ## 5. Extract the package:

```
# rpm2cpio syslinux-tftpboot-version-architecture.rpm | cpio -dimv
```

6. Create a **pxelinux**/ directory in **tftpboot/** and copy all the files from the directory into the **pxelinux**/ directory:

```
# mkdir /var/lib/tftpboot/pxelinux
```



```
# cp my_local_directory/tftpboot/* /var/lib/tftpboot/pxelinux
```

7. Create the directory **pxelinux.cfg** in the **pxelinux**/ directory:

```
# mkdir /var/lib/tftpboot/pxelinux/pxelinux.cfg
```

8. Add a default configuration file to the **pxelinux.cfg** directory as shown in the following example:

```
default vesamenu.c32
prompt 1
timeout 600

display boot.msg

label linux
menu label ^Install system
menu default
kernel images/RHEL-8.1/vmlinuz
append initrd=images/RHEL-8.1/initrd.img ip=dhcp inst.repo=http://10.32.5.1/RHEL-8.1/x86_64/iso-contents-root/
label vesa
menu label Install system with ^basic video driver
kernel images/RHEL-8.1/vmlinuz
append initrd=images/RHEL-8.1/initrd.img ip=dhcp inst.xdriver=vesa nomodeset
inst.repo=http://10.32.5.1/RHEL-8.1/x86_64/iso-contents-root/
label rescue
menu label ^Rescue installed system
```

```

kernel images/RHEL-8.1/vmlinuz
append initrd=images/RHEL-8.1/initrd.img rescue
label local
menu label Boot from ^local drive
localboot 0xffff

```



NOTE

- The installation program cannot boot without its runtime image. Use the **inst.stage2** boot option to specify location of the image. Alternatively, you can use the **inst.repo=** option to specify the image as well as the installation source.
- The installation source location used with **inst.repo** must contain a valid **.treeinfo** file.
- When you select the RHEL8 installation DVD as the installation source, the **.treeinfo** file points to the BaseOS and the AppStream repositories. You can use a single **inst.repo** option to load both repositories.

9. Create a subdirectory to store the boot image files in the **/var/lib/tftpboot/** directory, and copy the boot image files to the directory. In this example, the directory is **/var/lib/tftpboot/pxelinux/images/RHEL-8.1/**:

```

# mkdir -p /var/lib/tftpboot/pxelinux/images/RHEL-8.1/
# cp /path_to_x86_64_images/pxeboot/{vmlinuz,initrd.img}
/var/lib/tftpboot/pxelinux/images/RHEL-8.1/

```

10. Start and enable the **dhcpd** service:

```

# systemctl start dhcpd
# systemctl enable dhcpd

```

11. Start and enable the **xinetd** service that manages the **tftp** service:

```

# systemctl start xinetd
# systemctl enable xinetd

```

The PXE boot server is now ready to serve PXE clients. You can start the client, which is the system to which you are installing Red Hat Enterprise Linux, select **PXE Boot** when prompted to specify a boot source, and start the network installation.

26.3. CONFIGURING A TFTP SERVER FOR UEFI-BASED CLIENTS

Use this procedure to configure a TFTP server and DHCP server and start the TFTP service on the PXE server for UEFI-based AMD64, Intel 64, and 64-bit ARM systems.



IMPORTANT

- All configuration files in this section are examples. Configuration details vary and are dependent on the architecture and specific requirements.
- Red Hat Enterprise Linux 8 UEFI PXE boot supports a lowercase file format for a MAC-based grub menu file. For example, the MAC address file format for grub2 is **grub.cfg-01-aa-bb-cc-dd-ee-ff**

Procedure

1. As root, install the following packages:

```
# yum install tftp-server dhcp-server xinetd
```

2. Allow incoming connections to the **tftp service** in the firewall:

```
# firewall-cmd --add-service=tftp
```



NOTE

- This command enables temporary access until the next server reboot. To enable permanent access, add the **--permanent** option to the command.
- Depending on the location of the installation ISO file, you might have to allow incoming connections for HTTP or other services.

3. Configure your DHCP server to use the boot images packaged with **shim** as shown in the following example **/etc/dhcp/dhcpd.conf** file:

```
option space pxelinux;
option pxelinux.magic code 208 = string;
option pxelinux.configfile code 209 = text;
option pxelinux.pathprefix code 210 = text;
option pxelinux.reboottime code 211 = unsigned integer 32;
option architecture-type code 93 = unsigned integer 16;

subnet 10.0.0.0 netmask 255.255.255.0 {
    option routers 10.0.0.254;
    range 10.0.0.2 10.0.0.253;

    class "pxeclients" {
        match if substring (option vendor-class-identifier, 0, 9) = "PXEClient";
        next-server 10.0.0.1;

        if option architecture-type = 00:07 {
            filename "BOOTX64.efi";
        } else {
            filename "pxelinux/pxelinux.0";
        }
    }
}
```

4. Access the **BOOTX64.elf** file from the **shim** package, and the **grubx64.elf** file from the **grub2-efi** package in the Binary DVD ISO image file where *my_local_directory* is the name of the directory that you create:

```
# mount -t iso9660 /path_to_image/name_of_image.iso /mount_point -o loop,ro
# cp -pr /mount_point/BaseOS/Packages/shim-version-architecture.rpm /my_local_directory
# cp -pr /mount_point/BaseOS/Packages/grub2-efi-version-architecture.rpm
/my_local_directory
# umount /mount_point
```

5. Extract the packages:

```
# rpm2cpio shim-version-architecture.rpm | cpio -dimv
# rpm2cpio grub2-efi-version-architecture.rpm | cpio -dimv
```

6. Copy the EFI boot images from your boot directory. Replace *ARCH* with *shim* or *grub* followed by the architecture, for example, **grubx64**.

```
# cp my_local_directory/boot/efi/EFI/redhat/ARCH.elf /var/lib/tftpboot/uefi/
# cp my_local_directory/boot/efi/EFI/redhat/ARCH.elf /var/lib/tftpboot/uefi
```

7. Add a configuration file named **grub.cfg** to the **tftpboot/** directory as shown in the following example:

```
set timeout=60
menuentry 'RHEL 8' {
    linuxefi images/RHEL-8.1/vmlinuz ip=dhcp inst.repo=http://10.32.5.1/RHEL-8.1/x86_64/iso-
contents-root/
    initrdefi images/RHEL-8.1/initrd.img
}
```



NOTE

- The installation program cannot boot without its runtime image. Use the **inst.stage2** boot option to specify location of the image. Alternatively, you can use the **inst.repo=** option to specify the image as well as the installation source.
- The installation source location used with **inst.repo** must contain a valid **.treeinfo** file.
- When you select the RHEL8 installation DVD as the installation source, the **.treeinfo** file points to the BaseOS and the AppStream repositories. You can use a single **inst.repo** option to load both repositories.

8. Create a subdirectory to store the boot image files in the **/var/lib/tftpboot/** directory, and copy the boot image files to the directory. In this example, the directory is **/var/lib/tftpboot/images/RHEL-8.1/**:

```
# mkdir -p /var/lib/tftpboot/images/RHEL-8.1/
# cp /path_to_x86_64_images/pxeboot/{vmlinuz,initrd.img} /var/lib/tftpboot/images/RHEL-8.1/
```

9. Start and enable the **dhcpd** service:

```
# systemctl start dhcpd
# systemctl enable dhcpd
```

10. Start and enable the **xinetd** service that manages the **tftp** service:

```
# systemctl start xinetd
# systemctl enable xinetd
```

The PXE boot server is now ready to serve PXE clients. You can start the client, which is the system to which you are installing Red Hat Enterprise Linux, select **PXE Boot** when prompted to specify a boot source, and start the network installation.

Additional resources

- For more information about **shim**, see the upstream documentation: [Using the Shim Program](#).

26.4. CONFIGURING A NETWORK SERVER FOR IBM POWER SYSTEMS

Use this procedure to configure a network boot server for IBM Power systems using GRUB2.



IMPORTANT

All configuration files in this section are examples. Configuration details vary and are dependent on the architecture and specific requirements.

Procedure

1. As root, install the following packages:

```
# yum install tftp-server dhcp-server xinetd
```

2. Allow incoming connections to the **tftp service** in the firewall:

```
# firewall-cmd --add-service=tftp
```



NOTE

- This command enables temporary access until the next server reboot. To enable permanent access, add the **--permanent** option to the command.
- Depending on the location of the installation ISO file, you might have to allow incoming connections for HTTP or other services.

3. Create a **GRUB2** network boot directory inside the tftp root:

```
# grub2-mknetdir --net-directory=/var/lib/tftpboot
Netboot directory for powerpc-ieee1275 created. Configure your DHCP server to point to
/boot/grub2/powerpc-ieee1275/core.elf
```



NOTE

The command output informs you of the file name that needs to be configured in your DHCP configuration, described in this procedure.

- a. If the PXE server runs on an x86 machine, the **grub2-ppc64-modules** must be installed before creating a **GRUB2** network boot directory inside the tftp root:

```
# yum install grub2-ppc64-modules
```

4. Create a **GRUB2** configuration file: **/var/lib/tftpboot/boot/grub2/grub.cfg** as shown in the following example:

```
set default=0
set timeout=5

echo -e "\nWelcome to the Red Hat Enterprise Linux 8 installer!\n\n"

menuentry 'Red Hat Enterprise Linux 8' {
    linux grub2-ppc64/vmlinuz ro ip=dhcp inst.repo=http://10.32.5.1/RHEL-8.1/x86_64/iso-
contents-root/
    initrd grub2-ppc64/initrd.img
}
```



NOTE

- The installation program cannot boot without its runtime image. Use the **inst.stage2** boot option to specify location of the image. Alternatively, you can use the **inst.repo=** option to specify the image as well as the installation source.
- The installation source location used with **inst.repo** must contain a valid **.treeinfo** file.
- When you select the RHEL8 installation DVD as the installation source, the **.treeinfo** file points to the BaseOS and the AppStream repositories. You can use a single **inst.repo** option to load both repositories.

5. Mount the Binary DVD ISO image using the command:

```
# mount -t iso9660 /path_to_image/name_of_iso/ /mount_point -o loop,ro
```

6. Create a directory and copy the **initrd.img** and **vmlinuz** files from Binary DVD ISO image into it, for example:

```
# cp /mount_point/ppc/ppc64/{initrd.img,vmlinuz} /var/lib/tftpboot/grub2-ppc64/
```

7. Configure your DHCP server to use the boot images packaged with **GRUB2** as shown in the following example:

```
subnet 192.168.0.1 netmask 255.255.255.0 {  
    allow bootp;  
    option routers 192.168.0.5;  
    group { #BOOTP POWER clients  
        filename "boot/grub2/powerpc-ieee1275/core.elf";  
        host client1 {  
            hardware ethernet 01:23:45:67:89:ab;  
            fixed-address 192.168.0.112;  
        }  
    }  
}
```

8. Adjust the sample parameters **subnet**, **netmask**, **routers**, **fixed-address** and **hardware ethernet** to fit your network configuration. Note the **file name** parameter; this is the file name that was outputted by the **grub2-mknetdir** command earlier in this procedure.
9. Start and enable the **dhcpcd** service:

```
# systemctl start dhcpcd  
# systemctl enable dhcpcd
```

10. Start and enable **xinetd** service that manages the **tftp** service:

```
# systemctl start xinetd  
# systemctl enable xinetd
```

The PXE boot server is now ready to serve PXE clients. You can start the client, which is the system to which you are installing Red Hat Enterprise Linux, select **PXE Boot** when prompted to specify a boot source, and start the network installation.

CHAPTER 27. BOOT OPTIONS

This section contains information about some of the boot options that you can use to modify the default behavior of the installation program. For a full list of boot options, see the [upstream boot option](#) content.

27.1. TYPES OF BOOT OPTIONS

There are two types of boot options; those with an equals "==" sign, and those without an equals "==" sign. Boot options are appended to the boot command line and multiple options must be separated by a single space. Boot options that are specific to the installation program always start with **inst**.

Options with an equals "==" sign

You must specify a value for boot options that use the **=** symbol. For example, the **inst.vncpassword=** option must contain a value, in this case, a password. The correct syntax for this example is **inst.vncpassword=password**.

Options without an equals "==" sign

This boot option does not accept any values or parameters. For example, the **rd.live.check** option forces the installation program to verify the installation media before starting the installation. If this boot option is present, the verification is performed; if the boot option is not present, the verification is skipped.

27.2. EDITING BOOT OPTIONS

This section contains information about the different ways that you can edit boot options from the boot menu. The boot menu opens after you boot the installation media.

Editing the boot: prompt in BIOS

When using the **boot:** prompt, the first option must always specify the installation program image file that you want to load. In most cases, you can specify the image using the keyword. You can specify additional options according to your requirements.

Prerequisites

- You have created bootable installation media (USB, CD or DVD).
- You have booted the installation from the media, and the installation boot menu is open.

Procedure

1. With the boot menu open, press the **Esc** key on your keyboard.
2. The **boot:** prompt is now accessible.
3. Press the **Tab** key on your keyboard to display the help commands.
4. Press the **Enter** key on your keyboard to start the installation with your options. To return from the **boot:** prompt to the boot menu, restart the system and boot from the installation media again.



NOTE

The **boot:** prompt also accepts **dracut** kernel options. A list of options is available in the **dracut.cmdline(7)** man page.

Editing the > prompt

You can use the **>** prompt to edit predefined boot options. For example, select **Test this media and install Red Hat Enterprise Linux 8.1** from the boot menu to display a full set of options.



NOTE

This procedure is for BIOS-based AMD64 and Intel 64 systems.

Prerequisites

- You have created bootable installation media (USB, CD or DVD).
- You have booted the installation from the media, and the installation boot menu is open.

Procedure

1. From the boot menu, select an option and press the **Tab** key on your keyboard. The **>** prompt is accessible and displays the available options.
2. Append the options that you require to the **>** prompt.
3. Press the **Enter** key on your keyboard to start the installation.
4. Press the **Esc** key on your keyboard to cancel editing and return to the boot menu.

Editing the GRUB2 menu

The GRUB2 menu is available on UEFI-based AMD64, Intel 64, and 64-bit ARM systems.

Prerequisites

- You have created bootable installation media (USB, CD or DVD).
- You have booted the installation from the media, and the installation boot menu is open.

Procedure

1. From the boot menu window, select the required option and press the **e** key on your keyboard.
2. Move the cursor to the kernel command line. On UEFI systems, the kernel command line starts with **linuxefi**.
3. Move the cursor to the end of the **linuxefi** kernel command line.
4. Edit the parameters as required. For example, to configure one or more network interfaces, add the **ip=** parameter at the end of the **linuxefi** kernel command line, followed by the required value.
5. When you finish editing, press **Ctrl+X** on your keyboard to start the installation using the specified options.

27.3. INSTALLATION SOURCE BOOT OPTIONS

This section contains information about the various installation source boot options.

inst.repo=

The **inst.repo=** boot option specifies the installation source, that is, the location providing the package repositories and a valid **.treeinfo** file that describes them. For example: **inst.repo=cdrom**. The target of the **inst.repo=** option must be one of the following installation media:

- an installable tree, which is a directory structure containing the installation program images, packages, and repository data as well as a valid **.treeinfo** file
- a DVD (a physical disk present in the system DVD drive)
- an ISO image of the full Red Hat Enterprise Linux installation DVD, placed on a hard drive or a network location accessible to the system.

You can use the **inst.repo=** boot option to configure different installation methods using different formats. The following table contains details of the **inst.repo=** boot option syntax:

Table 27.1. **inst.repo=** installation source boot options

Source type	Boot option format	Source format
CD/DVD drive	inst.repo=cdrom[:device]	Installation DVD as a physical disk. [a]
Installable tree	inst.repo=hd:device:/path	Image file of the installation DVD, or an installation tree, which is a complete copy of the directories and files on the installation DVD.
NFS Server	inst.repo=nfs:[options:]server:/path	Image file of the installation DVD. [b]
HTTP Server	inst.repo=http://host/path	Installation tree, which is a complete copy of the directories and files on the installation DVD.
HTTPS Server	inst.repo=https://host/path	Installation tree, which is a complete copy of the directories and files on the installation DVD.
FTP Server	inst.repo=ftp://username:password@host/path	
HMC	inst.repo=hmc	

[a] If *device* is left out, installation program automatically searches for a drive containing the installation DVD.

[b] The NFS Server option uses NFS protocol version 3 by default. To use a different version *X*, add **+nfsvers=X** to *options*.



NOTE

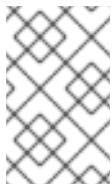
The NFS Server option uses NFS protocol version 3 by default. To use a different version, add `+nfsvers=X` to the option.

You can set disk device names with the following formats:

- Kernel device name, for example `/dev/sda1` or `sdb2`
- File system label, for example `LABEL=Flash` or `LABEL=RHEL8`
- File system UUID, for example `UUID=8176c7bf-04ff-403a-a832-9557f94e61db`
Non-alphanumeric characters must be represented as `\xNN`, where `NN` is the hexadecimal representation of the character. For example, `\x20` is a white space (" ").

`inst.addrepo=`

Use the `inst.addrepo=` boot option to add an additional repository that can be used as another installation source along with the main repository (`inst.repo=`). You can use the `inst.addrepo=` boot option multiple times during one boot. The following table contains details of the `inst.addrepo=` boot option syntax.



NOTE

The `REPO_NAME` is the name of the repository and is required in the installation process. These repositories are only used during the installation process; they are not installed on the installed system.

Table 27.2. `inst.addrepo` installation source boot options

Installation source	Boot option format	Additional information
Installable tree at a URL	<code>inst.addrepo=REPO_NAME, [http,https,ftp]://<host>/<path></code>	Looks for the installable tree at a given URL.
Installable tree at an NFS path	<code>inst.addrepo=REPO_NAME, nfs://<server>/<path></code>	Looks for the installable tree at a given NFS path. A colon is required after the host. The installation program passes every thing after <code>nfs://</code> directly to the mount command instead of parsing URLs according to RFC 2224.

Installation source	Boot option format	Additional information
Installable tree in the installation environment	inst.addrepo=REPO_NAME, file://<path>	Looks for the installable tree at the given location in the installation environment. To use this option, the repository must be mounted before the installation program attempts to load the available software groups. The benefit of this option is that you can have multiple repositories on one bootable ISO, and you can install both the main repository and additional repositories from the ISO. The path to the additional repositories is /run/install/source/REPO_IS_O_PATH . Additionally, you can mount the repository directory in the %pre section in the Kickstart file. The path must be absolute and start with /, for example inst.addrepo=REPO_NAME, file:///<path>
Hard Drive	inst.addrepo=REPO_NAME, hd:<device>:<path>	Mounts the given <device> partition and installs from the ISO that is specified by the <path> . If the <path> is not specified, the installation program looks for a valid installation ISO on the <device> . This installation method requires an ISO with a valid installable tree.

inst.noverifyssl=

The **noverifyssl=** boot option prevents the installation program from verifying the SSL certificate for all HTTPS connections with the exception of the additional Kickstart repositories, where **--noverifyssl** can be set per repository.

inst.stage2=

Use the **inst.stage2=** boot option to specify the location of the installation program runtime image. This option expects a path to a directory containing a valid **.treeinfo** file. The location of the runtime image is read from the **.treeinfo** file. If the **.treeinfo** file is not available, the installation program attempts to load the image from **images/install.img**.

When the **inst.stage2** option is not specified, the installation program attempts to use the location specified with **inst.repo** option.

You should specify this option only for PXE boot. The installation DVD and Boot ISO already contain a correct **inst.stage2** option to boot the installation program from themselves.



NOTE

By default, the **inst.stage2=** boot option is used on the installation media and is set to a specific label, for example, **inst.stage2=hd:LABEL=RHEL-8-0-0-BaseOS-x86_64**. If you modify the default label of the file system containing the runtime image, or if you use a customized procedure to boot the installation system, you must verify that the **inst.stage2=** boot option is set to the correct value.

inst.stage2.all

The **inst.stage2.all** boot option is used to specify several HTTP, HTTPS, or FTP sources. You can use the **inst.stage2=** boot option multiple times with the **inst.stage2.all** option to fetch the image from the sources sequentially until one succeeds. For example:

```
inst.stage2.all
inst.stage2=http://hostname1/path_to_install_tree/
inst.stage2=http://hostname2/path_to_install_tree/
inst.stage2=http://hostname3/path_to_install_tree/
```

inst.dd=

The **inst.dd=** boot option is used to perform a driver update during the installation. See the [Performing an advanced RHEL installation](#) document for information on how to update drivers during installation.

inst.repo=hmc

When booting from a Binary DVD, the installation program prompts you to enter additional kernel parameters. To set the DVD as an installation source, append **inst.repo=hmc** to the kernel parameters. The installation program then enables **SE** and **HMC** file access, fetches the images for stage2 from the DVD, and provides access to the packages on the DVD for software selection. This option eliminates the requirement of an external network setup and expands the installation options.

inst.proxy

The **inst.proxy** boot option is used when performing an installation from a HTTP, HTTPS, FTP source. For example:

```
[PROTOCOL://][USERNAME[:PASSWORD]@]HOST[:PORT]
```

inst.nosave

Use the **inst.nosave** boot option to control which installation logs and related files are not saved to the installed system, for example **input_ks**, **output_ks**, **all_ks**, **logs** and **all**. Multiple values can be combined as a comma-separated list, for example: **input_ks,logs**.



NOTE

The **inst.nosave** boot option is used for excluding files from the installed system that can't be removed by a Kickstart %post script, such as logs and input/output Kickstart results.

Table 27.3. inst.nosave boot options

Option	Description
--------	-------------

Option	Description
input_ks	Disables the ability to save the input Kickstart results.
output_ks	Disables the ability to save the output Kickstart results generated by the installation program.
all_ks	Disables the ability to save the input and output Kickstart results.
logs	Disables the ability to save all installation logs.
all	Disables the ability to save all Kickstart results, and all logs.

inst.multilib

Use the **inst.multilib** boot option to set DNF’s **multilib_policy** to **all**, instead of **best**.

memcheck

The **memcheck** boot option performs a check to verify that the system has enough RAM to complete the installation. If there isn’t enough RAM, the installation process is stopped. The system check is approximate and memory usage during installation depends on the package selection, user interface, for example graphical or text, and other parameters.

nomemcheck

The **nomemcheck** boot option does not perform a check to verify if the system has enough RAM to complete the installation. Any attempt to perform the installation with less than the recommended minimum amount of memory is unsupported, and might result in the installation process failing.

27.4. NETWORK BOOT OPTIONS

This section contains information about commonly used network boot options.



NOTE

Initial network initialization is handled by **dracut**. For a complete list, see the **dracut.cmdline(7)** man page.

ip=

Use the **ip=** boot option to configure one or more network interfaces. To configure multiple interfaces, you can use the **ip** option multiple times, once for each interface; to do so, you must use the **rd.neednet=1** option, and you must specify a primary boot interface using the **bootdev** option. Alternatively, you can use the **ip** option once, and then use Kickstart to set up further interfaces. This option accepts several different formats. The following tables contain information about the most common options.



NOTE

In the following tables:

- The **ip** parameter specifies the client IP address and requires square brackets, for example [2001:db8::99].
- The **gateway** parameter is the default gateway. IPv6 addresses are also accepted.
- The **netmask** parameter is the netmask to be used. This can be either a full netmask (for example, 255.255.255.0) or a prefix (for example, 64).
- The **hostname** parameter is the host name of the client system. This parameter is optional.

Table 27.4. Network interface configuration boot option formats

Configuration method	Boot option format
Automatic configuration of any interface	ip=method
Automatic configuration of a specific interface	ip=interface:method
Static configuration	ip=ip::gateway:netmask:hostname:interface:none
Automatic configuration of a specific interface with an override	ip=ip::gateway:netmask:hostname:interface:method:mtu



NOTE

The method **automatic configuration of a specific interface with an override** brings up the interface using the specified method of automatic configuration, such as **dhcp**, but overrides the automatically-obtained IP address, gateway, netmask, host name or other specified parameters. All parameters are optional, so specify only the parameters that you want to override.

The **method** parameter can be any of the following:

Table 27.5. Automatic interface configuration methods

Automatic configuration method	Value
DHCP	dhcp
IPv6 DHCP	dhcp6
IPv6 automatic configuration	auto6
iSCSI Boot Firmware Table (iBFT)	ibft



NOTE

- If you use a boot option that requires network access, such as **inst.ks=http://host:/path**, without specifying the **ip** option, the installation program uses **ip=dhcp**.
- To connect to an iSCSI target automatically, you must activate a network device for accessing the target. The recommended way to activate a network is to use the **ip=ibft** boot option.

nameserver=

The **nameserver=** option specifies the address of the name server. You can use this option multiple times.



NOTE

The **ip=** parameter requires square brackets. However, an IPv6 address does not work with square brackets. An example of the correct syntax to use for an IPv6 address is **nameserver=2001:db8::1**.

bootdev=

The **bootdev=** option specifies the boot interface. This option is mandatory if you use more than one **ip** option.

ifname=

The **ifname=** options assigns an interface name to a network device with a given MAC address. You can use this option multiple times. The syntax is **ifname=interface:MAC**. For example:

```
ifname=eth0:01:23:45:67:89:ab
```



NOTE

The **ifname=** option is the only supported way to set custom network interface names during installation.

inst.dhcpclass=

The **inst.dhcpclass=** option specifies the DHCP vendor class identifier. The **dhcpd** service sees this value as **vendor-class-identifier**. The default value is **anaconda-\$(uname -srm)**.

inst.waitfornet=

Using the **inst.waitfornet=SECONDS** boot option causes the installation system to wait for network connectivity before installation. The value given in the **SECONDS** argument specifies the maximum amount of time to wait for network connectivity before timing out and continuing the installation process even if network connectivity is not present.

Additional resources

- For more information about networking, see the [Configuring and managing networking](#) document.

27.5. CONSOLE BOOT OPTIONS

This section contains information about configuring boot options for your console, monitor display, and keyboard.

console=

Use the **console=** option to specify a device that you want to use as the primary console. For example, to use a console on the first serial port, use **console=ttyS0**. Use this option in conjunction with the **inst.text** option. You can use the **console=** option multiple times. If you do, the boot message is displayed on all specified consoles, but only the last one is used by the installation program. For example, if you specify **console=ttyS0 console=ttyS1**, the installation program uses **ttyS1**.

inst.lang=

Use the **inst.lang=** option to set the language that you want to use during the installation. The **locale -a | grep _** or **localectl list-locales | grep _** commands return a list of locales.

inst.singlelang

Use the **inst.singlelang** option to install in single language mode, which results in no available interactive options for the installation language and language support configuration. If a language is specified using the **inst.lang** boot option or the **lang** Kickstart command, then it is used. If no language is specified, the installation program defaults to **en_US.UTF-8**.

inst.geoloc=

Use the **inst.geoloc=** option to configure geolocation usage in the installation program. Geolocation is used to preset the language and time zone, and uses the following syntax: **inst.geoloc=value**. The **value** can be any of the following parameters:

Table 27.6. Values for the inst.geoloc boot option

Value	Boot option format
Disable geolocation	inst.geoloc=0
Use the Fedora GeolIP API	inst.geoloc=provider_fedora_geoiip
Use the Hostip.info GeolIP API	inst.geoloc=provider_hostip

If you do not specify the **inst.geoloc=** option, the installation program uses **provider_fedora_geoiip**.

inst.keymap=

Use the **inst.keymap=** option to specify the keyboard layout that you want to use for the installation.

inst.cmdline

Use the **inst.cmdline** option to force the installation program to run in command-line mode. This mode does not allow any interaction, and you must specify all options in a Kickstart file or on the command line.

inst.graphical

Use the **inst.graphical** option to force the installation program to run in graphical mode. This mode is the default.

inst.text

Use the **inst.text** option to force the installation program to run in text mode instead of graphical mode.

inst.noninteractive

Use the **inst.noninteractive** boot option to run the installation program in a non-interactive mode. User interaction is not permitted in the non-interactive mode, and **inst.noninteractive** can be used with a graphical or text installation. When the **inst.noninteractive** option is used in text mode it behaves the same as the **inst.cmdline** option.

inst.resolution=

Use the **inst.resolution=** option to specify the screen resolution in graphical mode. The format is **NxM**, where *N* is the screen width and *M* is the screen height (in pixels). The lowest supported resolution is 1024x768.

inst.vnc=

Use the **inst.vnc=** option to run the graphical installation using VNC. You must use a VNC client application to interact with the installation program. When VNC sharing is enabled, multiple clients can connect. A system installed using VNC starts in text mode.

inst.vncpassword=

Use the **inst.vncpassword=** option to set a password on the VNC server that is used by the installation program.

inst.vncconnect=

Use the **inst.vncconnect=** option to connect to a listening VNC client at the given host location. For example **inst.vncconnect=<host>[:<port>]** The default port is 5900. This option can be used with **vncviewer -listen**.

inst.xdriver=

Use the **inst.xdriver=** option to specify the name of the X driver that you want to use both during installation and on the installed system.

inst.usefbx=

Use the **inst.usefbx** option to prompt the installation program to use the frame buffer X driver instead of a hardware-specific driver. This option is equivalent to **inst.xdriver=fbdev**.

modprobe.blacklist=

Use the **modprobe.blacklist=** option to blacklist or completely disable one or more drivers. Drivers (mods) that you disable using this option cannot load when the installation starts, and after the installation finishes, the installed system retains these settings. You can find a list of the blacklisted drivers in the **/etc/modprobe.d/** directory. Use a comma-separated list to disable multiple drivers. For example:

```
modprobe.blacklist=ahci,firewire_ohci
```

inst.xtimeout=

Use the **inst.xtimeout=** option to specify the timeout in seconds for starting X server.

inst.sshd

Use the **inst.sshd** option to start the **sshd** service during installation, so that you can connect to the system during the installation using SSH, and monitor the installation progress. For more information about SSH, see the **ssh(1)** man page. By default, the **sshd** option is automatically started only on the IBM Z architecture. On other architectures, **sshd** is not started unless you use the **inst.sshd** option.



NOTE

During installation, the root account has no password by default. You can set a root password during installation with the **sshpw** Kickstart command.

inst.kdump-addon=

Use the **inst.kdump-addon=** option to enable or disable the Kdump configuration screen (add-on) in the installation program. This screen is enabled by default; use **inst.kdump-addon=off** to disable it. Disabling the add-on disables the Kdump screens in both the graphical and text-based interface as well as the **%addon com_redhat_kdump** Kickstart command.

27.6. DEBUG BOOT OPTIONS

This section contains information about the options that you can use when debugging issues.

inst.rescue=

Use the **inst.rescue=** option to run the rescue environment. The option is useful for trying to diagnose and fix systems.

inst.updates=

Use the **inst.updates=** option to specify the location of the **updates.img** file that you want to apply during installation. There are a number of sources for the updates.

Table 27.7. inst.updates= source updates

Source	Description	Example
Updates from a network	The easiest way to use inst.updates= is to specify the network location of updates.img . This does not require any modification to the installation tree. To use this method, edit the kernel command line to include inst.updates .	inst.updates=http://some.website.com/path/to/updates.img .
Updates from a disk image	You can save an updates.img on a floppy drive or a USB key. This can be done only with an ext2 filesystem type of updates.img . To save the contents of the image on your floppy drive, insert the floppy disc and run the command.	dd if=updates.img of=/dev/fd0 bs=72k count=20 . To use a USB key or flash media, replace /dev/fd0 with the device name of your USB key.
Updates from an installation tree	If you are using a CD, hard drive, HTTP, or FTP install, you can save the updates.img in the installation tree so that all installations can detect the .img file. Save the file in the images/ directory. The file name must be updates.img .	For NFS installs, there are two options: You can either save the image in the images/ directory, or in the RHupdates/ directory in the installation tree.

inst.loglevel=

Use the **inst.loglevel=** option to specify the minimum level of messages logged on a terminal. This

concerns only terminal logging; log files always contain messages of all levels. Possible values for this option from the lowest to highest level are: **debug**, **info**, **warning**, **error** and **critical**. The default value is **info**, which means that by default, the logging terminal displays messages ranging from **info** to **critical**.

inst.syslog=

When installation starts, the **inst.syslog=** option sends log messages to the **syslog** process on the specified host. The remote **syslog** process must be configured to accept incoming connections.

inst.virtiolog=

Use the **inst.virtiolog=** option to specify the virtio port (a character device at **/dev/virtio-ports/name**) that you want to use for forwarding logs. The default value is **org.fedoraproject.anaconda.log.0**; if this port is present, it is used.

inst.zram

The **inst.zram** option controls the usage of zRAM swap during installation. The option creates a compressed block device inside the system RAM and uses it for swap space instead of the hard drive. This allows the installation program to run with less available memory than is possible without compression, and it might also make the installation faster. By default, swap on zRAM is enabled on systems with 2 GiB or less RAM, and disabled on systems with more than 2 GiB of memory. You can use this option to change this behavior; on a system with more than 2 GiB RAM, use **inst.zram=1** to enable the feature, and on systems with 2 GiB or less memory, use **inst.zram=0** to disable the feature.

rd.live.ram

If the **rd.live.ram** option is specified, the **stage 2** image is copied into RAM. Using this option when the **stage 2** image is on an NFS server increases the minimum required memory by the size of the image by roughly 500 MiB.

inst.nokill

The **inst.nokill** option is a debugging option that prevents the installation program from rebooting when a fatal error occurs, or at the end of the installation process. Use the **inst.nokill** option to capture installation logs which would be lost upon reboot.

inst.noshell

Use **inst.noshell** option if you do not want a shell on terminal session 2 (tty2) during installation.

inst.notmux

Use **inst.notmux** option if you do not want to use tmux during installation. The output is generated without terminal control characters and is meant for non-interactive uses.

remotelog

You can use the **remotelog** option to send all of the logs to a remote **host:port** using a TCP connection. The connection is retired if there is no listener and the installation proceeds as normal.

27.7. STORAGE BOOT OPTIONS

inst.nodmraid

Use the **inst.nodmraid** option to disable **dmraid** support.

**WARNING**

Use this option with caution. If you have a disk that is incorrectly identified as part of a firmware RAID array, it might have some stale RAID metadata on it that must be removed using the appropriate tool, for example, **dmraid** or **wipefs**.

inst.nompath

Use the **inst.nompath** option to disable support for multipath devices. This option can be used for systems on which a false-positive is encountered which incorrectly identifies a normal block device as a multipath device. There is no other reason to use this option.

**WARNING**

Use this option with caution. You should not use this option with multipath hardware. Using this option to attempt to install to a single path of a multipath is not supported.

inst.gpt

The **inst.gpt** boot option forces the installation program to install partition information to a GUID Partition Table (GPT) instead of a Master Boot Record (MBR). This option is not valid on UEFI-based systems, unless they are in BIOS compatibility mode. Normally, BIOS-based systems and UEFI-based systems in BIOS compatibility mode attempt to use the MBR schema for storing partitioning information, unless the disk is 232 sectors in size or larger. Disk sectors are typically 512 bytes in size, meaning that this is usually equivalent to 2 TiB. Using the **inst.gpt** boot option changes this behavior, allowing a GPT to be written to smaller disks.

27.8. KICKSTART BOOT OPTIONS

This section contains information about the Kickstart boot options.

inst.ks=

Use the **inst.ks=** boot option to define the location of a Kickstart file that you want to use to automate the installation. You can then specify locations using any of the **inst.repo** formats. If you specify a device and not a path, the installation program looks for the Kickstart file in **/ks.cfg** on the device that you specify. If you use this option without specifying a device, the installation program uses the following option:

inst.ks=nfs:next-server:/filename

In the previous example, *next-server* is the DHCP next-server option or the IP address of the DHCP server itself, and *filename* is the DHCP filename option, or */kickstart/*. If the given file name ends with the / character, **ip-kickstart** is appended. The following table contains an example.

Table 27.8. Default Kickstart file location

DHCP server address	Client address	Kickstart file location
192.168.122.1	192.168.122.100	192.168.122.1:/kickstart/192.168.122.100-kickstart

If a volume with a label of **OEMDRV** is present, the installation program attempts to load a Kickstart file named **ks.cfg**. If your Kickstart file is in this location, you do not need to use the **inst.ks=** boot option.

inst.ks.all

Specify this option to sequentially try multiple Kickstart file locations provided by multiple **inst.ks** options. The first successful location is used. This applies only to locations of type **http**, **https** or **ftp**, other locations are ignored.

inst.ks.sendmac

Use the **inst.ks.sendmac** option to add headers to outgoing HTTP requests that contain the MAC addresses of all network interfaces. For example:

X-RHN-Provisioning-MAC-0: eth0 01:23:45:67:89:ab

This can be useful when using **inst.ks=http** to provision systems.

inst.ks.sendsn

Use the **inst.ks.sendsn** option to add a header to outgoing HTTP requests. This header contains the system serial number, read from **/sys/class/dmi/id/product_serial**. The header has the following syntax:

X-System-Serial-Number: R8VA23D

Additional resources

- For a full list of boot options, see the [upstream boot option](#) content.

27.9. ADVANCED INSTALLATION BOOT OPTIONS

This section contains information about advanced installation boot options.

inst.kexec

The **inst.kexec** option allows the installation program to use the **kexec** system call at the end of the installation, instead of performing a reboot. The **inst.kexec** option loads the new system immediately, and bypasses the hardware initialization normally performed by the BIOS or firmware.



IMPORTANT

This option is deprecated and available as a Technology Preview only. For information on Red Hat scope of support for Technology Preview features, see the [Technology Preview Features Support Scope](#) document.

When **kexec** is used, device registers which would normally be cleared during a full system reboot, might stay filled with data, which could potentially create issues for some device drivers.

inst.multilib

Use the **inst.multilib** boot option to configure the system for multilib packages, that is, to allow installing 32-bit packages on a 64-bit AMD64 or Intel 64 system. Normally, on an AMD64 or Intel 64 system, only packages for this architecture (marked as x86_64) and packages for all architectures (marked as noarch) are installed. When you use the **inst.multilib** boot option, packages for 32-bit AMD or Intel systems (marked as i686) are automatically installed.

This applies only to packages directly specified in the **%packages** section. If a package is installed as a dependency, only the exact specified dependency is installed. For example, if you are installing the **bash** package which depends on the **glibc** package, the former is installed in multiple variants, while the latter is installed only in variants that the bash package requires.

selinux=0

By default, the **selinux=0** boot option operates in permissive mode in the installation program, and in enforcing mode in the installed system. The **selinux=0** boot option disables the use of SELinux in the installation program and the installed system.



NOTE

The **selinux=0** and **inst.selinux=0** options are not the same. The **selinux=0** option disables the use of SELinux in the installation program and the installed system. The **inst.selinux=0** option disables SELinux only in the installation program. By default, SELinux operates in permissive mode in the installation program, so disabling SELinux has little effect.

inst.nonibftiscsiboot=

Use the **inst.nonibftiscsiboot=** boot option to place the boot loader on iSCSI devices that were not configured in the iSCSI Boot Firmware Table (iBFT).

27.10. DEPRECATED BOOT OPTIONS

This section contains information about deprecated boot options. These options are still accepted by the installation program but they are deprecated and are scheduled to be removed in a future release of Red Hat Enterprise Linux.

method

The **method** option is an alias for **inst.repo**.

repo=nfsiso

The **repo=nfsiso:** option is the same as **inst.repo=nfs:**

dns

Use **nameserver** instead of **dns**. Note that nameserver does not accept comma-separated lists; use multiple nameserver options instead.

netmask, gateway, hostname

The **netmask**, **gateway**, and **hostname** options are provided as part of the **ip** option.

ip=bootif

A PXE-supplied **BOOTIF** option is used automatically, so there is no requirement to use **ip=bootif**.

ksdevice

Table 27.9. Values for the **ksdevice** boot option

Value	Information
Not present	N/A
ksdevice=link	Ignored as this option is the same as the default behavior
ksdevice=bootif	Ignored as this option is the default if BOOTIF= is present
ksdevice=ibft	Replaced with ip=ibft . See ip for details
ksdevice=<MAC>	Replaced with BOOTIF=\${MAC/:/-}
ksdevice=<DEV>	Replaced with bootdev

27.11. REMOVED BOOT OPTIONS

This section contains the boot options that have been removed from Red Hat Enterprise Linux.



NOTE

dracut provides advanced boot options. For more information about **dracut**, see the **dracut.cmdline(7)** man page.

askmethod, asknetwork

initramfs is completely non-interactive, so the **askmethod** and **asknetwork** options have been removed. Instead, use **inst.repo** or specify the appropriate network options.

blacklist, nofirewire

The **modprobe** option handles blacklisting kernel modules; use **modprobe.blacklist=<mod1>, <mod2>**. You can blacklist the firewire module by using **modprobe.blacklist=firewire_ohci**.

inst.headless=

The **headless=** option specified that the system that is being installed to does not have any display hardware, and that the installation program is not required to look for any display hardware.

inst.decorated

The **inst.decorated** option was used to specify the graphical installation in a decorated window. By default, the window is not decorated, so it doesn't have a title bar, resize controls, and so on. This option was no longer required.

serial

Use the **console=ttyS0** option.

updates

Use the **inst.updates** option.

essid, wepkey, wpakey

Dracut does not support wireless networking.

ethtool

This option was no longer required.

gdb

This option was removed as there are many options available for debugging dracut-based **initramfs**.

inst.mediacheck

Use the **dracut option rd.live.check** option.

ks=floppy

Use the **inst.ks=hd:<device>** option.

display

For a remote display of the UI, use the **inst.vnc** option.

utf8

This option was no longer required as the default TERM setting behaves as expected.

noipv6

ipv6 is built into the kernel and cannot be removed by the installation program. You can disable ipv6 using **ipv6.disable=1**. This setting is used by the installed system.

upgradeany

This option was no longer required as the installation program no longer handles upgrades.

CHAPTER 28. KICKSTART REFERENCES

APPENDIX G. KICKSTART SCRIPT FILE FORMAT REFERENCE

This reference describes in detail the kickstart file format.

G.1. KICKSTART FILE FORMAT

Kickstart scripts are plain text files that contain keywords recognized by the installation program, which serve as directions for the installation. Any text editor able to save files as ASCII text, such as **Gedit** or **vim** on Linux systems or **Notepad** on Windows systems, can be used to create and edit Kickstart files. The file name of your Kickstart configuration does not matter; however, it is recommended to use a simple name as you will need to specify this name later in other configuration files or dialogs.

Commands

Commands are keywords that serve as directions for installation. Each command must be on a single line. Commands can take options. Specifying commands and options is similar to using Linux commands in shell.

Sections

Certain special commands that begin with the percent **%** character start a section. Interpretation of commands in sections is different from commands placed outside sections. Every section must be finished with **%end** command.

Section types

The available sections are:

- **Add-on sections.** These sections use the **%addon *addon_name*** command.
- **Package selection sections.** Starts with **%packages**. Use it to list packages for installation, including indirect means such as package groups or modules.
- **Script sections.** These start with **%pre**, **%pre-install**, **%post**, and **%onerror**. These sections are not required.

Command section

The command section is a term used for the commands in the Kickstart file that are not part of any script section or **%packages** section.

Script section count and ordering

All sections except the command section are optional and can be present multiple times. When a particular type of script section is to be evaluated, all sections of that type present in the Kickstart are evaluated in order of appearance: two **%post** sections are evaluated one after another, in the order as they appear. However, you do not have to specify the various types of script sections in any order: it does not matter if there are **%post** sections before **%pre** sections.

Comments

Kickstart comments are lines starting with the hash **#** character. These lines are ignored by the installation program.

Items that are not required can be omitted. Omitting any required item results in the installation program changing to the interactive mode so that the user can provide an answer to the related item, just as during a regular interactive installation. It is also possible to declare the kickstart script as non-interactive with the **cmdline** command. In non-interactive mode, any missing answer aborts the installation process.

G.2. PACKAGE SELECTION IN KICKSTART

Kickstart uses sections started by the **%packages** command for selecting packages to install. You can install packages, groups, environments, module streams, and module profiles this way.

G.2.1. Package selection section

Use the **%packages** command to begin a Kickstart section which describes the software packages to be installed. The **%packages** section must end with the **%end** command.

You can specify packages by environment, group, module stream, module profile, or by their package names. Several environments and groups that contain related packages are defined. See the **repository/repo-data/*-comps-repository.architecture.xml** file on the Red Hat Enterprise Linux 8 Installation DVD for a list of environments and groups.

The ***-comps-repository.architecture.xml** file contains a structure describing available environments (marked by the **<environment>** tag) and groups (the **<group>** tag). Each entry has an ID, user visibility value, name, description, and package list. If the group is selected for installation, the packages marked **mandatory** in the package list are always installed, the packages marked **default** are installed if they are not specifically excluded elsewhere, and the packages marked **optional** must be specifically included elsewhere even when the group is selected.

You can specify a package group or environment using either its ID (the **<id>** tag) or name (the **<name>** tag).

If you are not sure what package should be installed, Red Hat recommends you to select the **Minimal Install** environment. **Minimal Install** provides only the packages which are essential for running Red Hat Enterprise Linux 8. This will substantially reduce the chance of the system being affected by a vulnerability. If necessary, additional packages can be added later after the installation. For more details on **Minimal Install**, see the [Installing the Minimum Amount of Packages Required](#) section of the *Security Hardening* document. Note that **Initial Setup** can not run after a system is installed from a Kickstart file unless a desktop environment and the X Window System were included in the installation and graphical login was enabled.



IMPORTANT

To install a 32-bit package on a 64-bit system:

- specify the **--multilib** option for the **%packages** section
- append the package name with the 32-bit architecture for which the package was built; for example, **glibc.i686**

G.2.2. Package selection commands

These commands can be used within the **%packages** section of a Kickstart file.

Specifying an environment

Specify an entire environment to be installed as a line starting with the **@^** symbols:

```
%packages
@^Infrastructure Server
%end
```

This installs all packages which are part of the **Infrastructure Server** environment. All available environments are described in the **repository/repodata/*-comps-repository.architecture.xml** file on the Red Hat Enterprise Linux 8 Installation DVD.

Only a single environment should be specified in the Kickstart file. If more environments are specified, only the last specified environment is used.

Specifying groups

Specify groups, one entry to a line, starting with an @ symbol, and then the full group name or group id as given in the ***-comps-repository.architecture.xml** file. For example:

```
%packages
@X Window System
@Desktop
@Sound and Video
%end
```

The **Core** group is always selected – it is not necessary to specify it in the **%packages** section.

Specifying individual packages

Specify individual packages by name, one entry to a line. You can use the asterisk character (*) as a wildcard in package names. For example:

```
%packages
sqlite
curl
aspell
docbook*
%end
```

The **docbook*** entry includes the packages **docbook-dtds** and **docbook-style** that match the pattern represented with the wildcard.

Specifying profiles of module streams

Specify profiles for module streams, one entry to a line, using the syntax for profiles:

```
%packages
@module:stream/profile
%end
```

This installs all packages listed in the specified profile of the module stream.

- When a module has a default stream specified, you can leave it out. When the default stream is not specified, you must specify it.
- When a module stream has a default profile specified, you can leave it out. When the default profile is not specified, you must specify it.
- Installing a module multiple times with different streams is not possible.
- Installing multiple profiles of the same module and stream is possible.

Modules and groups use the same syntax starting with the @ symbol. When a module and a package group exist with the same name, the module takes precedence.

In Red Hat Enterprise Linux 8, modules are present only in the AppStream repository. To list available modules, use the **yum module list** command on an installed Red Hat Enterprise Linux 8 system.

It is also possible to enable module streams using the **module** Kickstart command and then install packages contained in the module stream by naming them directly.

Excluding environments, groups, or packages

Use a leading dash (-) to specify packages or groups to exclude from the installation. For example:

```
%packages
-@Graphical Administration Tools
-autofs
-ipa*compat
%end
```



IMPORTANT

Installing all available packages using only * in a Kickstart file is not supported.

You can change the default behavior of the **%packages** section by using several options. Some options work for the entire package selection, others are used with only specific groups.

Additional resources

- For more information about handling packages, see the [Installing software](#) chapter of the [Configuring basic system settings](#) document.
- For more information about modules and streams, see the [Installing, managing, and removing user-space components](#) document.

G.2.3. Common package selection options

The following options are available for the **%packages** sections. To use an option, append it to the start of the package selection section. For example:

```
%packages --multilib --ignoremissing
```

--default

Install the default set of packages. This corresponds to the package set which would be installed if no other selections were made in the **Package Selection** screen during an interactive installation.

--excludedocs

Do not install any documentation contained within packages. In most cases, this excludes any files normally installed in the **/usr/share/doc** directory, but the specific files to be excluded depend on individual packages.

--ignoremissing

Ignore any packages, groups, module streams, module profiles, and environments missing in the installation source, instead of halting the installation to ask if the installation should be aborted or continued.

--instLangs=

Specify a list of languages to install. Note that this is different from package group level selections. This option does not describe which package groups should be installed; instead, it sets RPM macros controlling which translation files from individual packages should be installed.

--multilib

Configure the installed system for multilib packages, to allow installing 32-bit packages on a 64-bit system, and install packages specified in this section as such.

Normally, on an AMD64 and Intel 64 system, you can install only the x86_64 and the noarch packages. However, with the **--multilib** option, you can automatically install the 32-bit AMD and the i686 Intel system packages available, if any.

This only applies to packages explicitly specified in the **%packages** section. Packages which are only being installed as dependencies without being specified in the Kickstart file are only installed in architecture versions in which they are needed, even if they are available for more architectures.

This option only works during the installation. Already installed systems are not configured for multilib packages installation using the **dnf** command.

--nocore

Disables installation of the **@Core** package group which is otherwise always installed by default.

Disabling the **@Core** package group with **--nocore** should be only used for creating lightweight containers; installing a desktop or server system with **--nocore** will result in an unusable system.



NOTES

- Using **-@Core** to exclude packages in the **@Core** package group does not work. The only way to exclude the **@Core** package group is with the **--nocore** option.
- The **@Core** package group is defined as a minimal set of packages needed for installing a working system. It is not related in any way to core packages as defined in the [Package Manifest](#) and [Scope of Coverage Details](#).

--excludeWeakdeps

Disables installation of packages from weak dependencies. These are packages linked to the selected package set by Recommends and Supplements flags. By default weak dependencies will be installed.

--retries=

Sets the number of times Yum will attempt to download packages (retries). The default value is 10. This option only applies during the installation, and will not affect Yum configuration on the installed system.

--timeout=

Sets the Yum timeout in seconds. The default value is 30. This option only applies during the installation, and will not affect Yum configuration on the installed system.

G.2.4. Options for specific package groups

The options in this list only apply to a single package group. Instead of using them at the **%packages** command in the Kickstart file, append them to the group name. For example:

```
%packages
@Graphical Administration Tools --optional
%end
```

--nodefaults

Only install the group's mandatory packages, not the default selections.

--optional

Install packages marked as optional in the group definition in the ***-comps-repository.architecture.xml** file, in addition to installing the default selections.

Note that some package groups, such as **Scientific Support**, do not have any mandatory or default packages specified - only optional packages. In this case the **--optional** option must always be used, otherwise no packages from this group will be installed.

G.3. SCRIPTS IN KICKSTART FILE

A kickstart file can include the following scripts:

- **%pre**
- **%pre-install**
- **%post**

This section provides the following details about the scripts:

- Execution time
- Types of commands that can be included in the script
- Purpose of the script
- Script options

G.3.1. **%pre** script

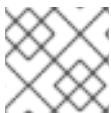
The **%pre** scripts are run on the system immediately after the Kickstart file has been loaded, but before it is completely parsed and installation begins. Each of these sections must start with **%pre** and end with **%end**.

The **%pre** script can be used for activation and configuration of networking and storage devices. It is also possible to run scripts, using interpreters available in the installation environment. Adding a **%pre** script can be useful if you have networking and storage that needs special configuration before proceeding with the installation, or have a script that, for example, sets up additional logging parameters or environment variables.

Debugging problems with **%pre** scripts can be difficult, so it is recommended only to use a **%pre** script when necessary.

Commands related to networking, storage, and file systems are available to use in the **%pre** script, in addition to most of the utilities in the installation environment **/sbin** and **/bin** directories.

You can access the network in the **%pre** section. However, the name service has not been configured at this point, so only IP addresses work, not URLs.



NOTE

The pre script does not run in the chroot environment.

G.3.1.1. %pre script section options

The following options can be used to change the behavior of pre-installation scripts. To use an option, append it to the **%pre** line at the beginning of the script. For example:

```
%pre --interpreter=/usr/libexec/platform-python
-- Python script omitted --
%end
```

--interpreter=

Allows you to specify a different scripting language, such as Python. Any scripting language available on the system can be used; in most cases, these are **/usr/bin/sh**, **/usr/bin/bash**, and **/usr/libexec/platform-python**.

Note that the **platform-python** interpreter uses Python version 3.6. You must change your Python scripts from previous RHEL versions for the new path and version. Additionally, **platform-python** is meant for system tools: Use the **python36** package outside the installation environment. For more details about Python in Red Hat Enterprise Linux 8, see [Introduction to Python](#) in [Configuring basic system settings](#).

--erroronfail

Display an error and halt the installation if the script fails. The error message will direct you to where the cause of the failure is logged.

--log=

Logs the script's output into the specified log file. For example:

```
%pre --log=/tmp/ks-pre.log
```

G.3.2. %pre-install script

The commands in the **pre-install** script are run after the following tasks are complete:

- System is partitioned
- Filesystems are created and mounted under /mnt/sysimage
- Network has been configured according to any boot options and kickstart commands

Each of the **%pre-install** sections must start with **%pre-install** and end with **%end**.

The **%pre-install** scripts can be used to modify the installation, and to add users and groups with guaranteed IDs before package installation.

It is recommended to use the **%post** scripts for any modifications required in the installation. Use the **%pre-install** script only if the **%post** script falls short for the required modifications.

Note: The **pre-install** script does not run in chroot environment.

G.3.2.1. %pre-install script section options

The following options can be used to change the behavior of **pre-install** scripts. To use an option, append it to the **%pre-install** line at the beginning of the script. For example:

```
%pre-install --interpreter=/usr/libexec/platform-python
-- Python script omitted --
%end
```

Note that you can have multiple **%pre-install** sections, with same or different interpreters. They are evaluated in their order of appearance in the Kickstart file.

--interpreter=

Allows you to specify a different scripting language, such as Python. Any scripting language available on the system can be used; in most cases, these are **/usr/bin/sh**, **/usr/bin/bash**, and **/usr/libexec/platform-python**.

Note that the **platform-python** interpreter uses Python version 3.6. You must change your Python scripts from previous RHEL versions for the new path and version. Additionally, **platform-python** is meant for system tools: Use the **python36** package outside the installation environment. For more details about Python in Red Hat Enterprise Linux 8, see [Introduction to Python](#) in [Configuring basic system settings](#).

--erroronfail

Display an error and halt the installation if the script fails. The error message will direct you to where the cause of the failure is logged.

--log=

Logs the script's output into the specified log file. For example:

```
%pre-install --log=/mnt/sysimage/root/ks-pre.log
```

G.3.3. **%post** script

The **%post** script is a post-installation script that is run after the installation is complete, but before the system is rebooted for the first time. You can use this section to run tasks such as system subscription.

You have the option of adding commands to run on the system once the installation is complete, but before the system is rebooted for the first time. This section must start with **%post** and end with **%end**.

The **%post** section is useful for functions such as installing additional software or configuring an additional name server. The post-install script is run in a **chroot** environment, therefore, performing tasks such as copying scripts or RPM packages from the installation media do not work by default. You can change this behavior using the **--nochroot** option as described below. Then the **%post** script will run in the installation environment, not in **chroot** on the installed target system.

Because post-install script runs in a **chroot** environment, most **systemctl** commands will refuse to perform any action. For more information, see the [Behavior of systemctl in a chroot Environment](#) section of the [Configuring and managing system administration](#) document.

Note that during execution of the **%post** section, the installation media must be still inserted.

G.3.3.1. **%post** script section options

The following options can be used to change the behavior of post-installation scripts. To use an option, append it to the **%post** line at the beginning of the script. For example:

```
%post --interpreter=/usr/libexec/platform-python
-- Python script omitted --
%end
```

--interpreter=

Allows you to specify a different scripting language, such as Python. For example:

```
%post --interpreter=/usr/libexec/platform-python
```

Any scripting language available on the system can be used; in most cases, these are **/usr/bin/sh**, **/usr/bin/bash**, and **/usr/libexec/platform-python**.

Note that the **platform-python** interpreter uses Python version 3.6. You must change your Python scripts from previous RHEL versions for the new path and version. Additionally, **platform-python** is meant for system tools: Use the **python36** package outside the installation environment. For more details about Python in Red Hat Enterprise Linux 8, see [Introduction to Python](#) in *Configuring basic system settings*.

--nochroot

Allows you to specify commands that you would like to run outside of the chroot environment. The following example copies the file `/etc/resolv.conf` to the file system that was just installed.

```
%post --nochroot
cp /etc/resolv.conf /mnt/sysimage/etc/resolv.conf
%end
```

--erroronfail

Display an error and halt the installation if the script fails. The error message will direct you to where the cause of the failure is logged.

--log=

Logs the script's output into the specified log file. Note that the path of the log file must take into account whether or not you use the **--nochroot** option. For example, without **--nochroot**:

```
%post --log=/root/ks-post.log
```

and with **--nochroot**:

```
%post --nochroot --log=/mnt/sysimage/root/ks-post.log
```

G.3.3.2. Example: Mounting NFS in a post-install script

This example of a **%post** section mounts an NFS share and executes a script named **runme** located at **/usr/new-machines/** on the share. Note that NFS file locking is not supported while in Kickstart mode, therefore the **-o nolock** option is required.

```
# Start of the %post section with logging into /root/ks-post.log
%post --log=/root/ks-post.log

# Mount an NFS share
mkdir /mnt/temp
```

```

mount -o nolock 10.10.0.2:/usr/new-machines /mnt/temp
openvt -s -w -- /mnt/temp/runme
umount /mnt/temp

# End of the %post section
%end

```

G.3.3.3. Example: Running subscription-manager as a post-install script

One of the most common uses of post-installation scripts in Kickstart installations is automatic registration of the installed system using Red Hat Subscription Manager. The following is an example of automatic subscription in a **%post** script:

```

%post --log=/root/ks-post.log
subscription-manager register --username=admin@example.com --password=secret --auto-attach
%end

```

The subscription-manager command-line script registers a system to a Red Hat Subscription Management server (Customer Portal Subscription Management, Satellite 6, or CloudForms System Engine). This script can also be used to assign or attach subscriptions automatically to the system that best-match that system. When registering to the Customer Portal, use the Red Hat Network login credentials. When registering to Satellite 6 or CloudForms System Engine, you may also need to specify more subscription-manager options like **--serverurl**, **--org**, **--environment** as well as credentials provided by your local administrator. Note that credentials in the form of an **--org --activationkey** combination is a good way to avoid exposing **--username --password** values in shared kickstart files.

Additional options can be used with the registration command to set a preferred service level for the system and to restrict updates and errata to a specific minor release version of RHEL for customers with Extended Update Support subscriptions that need to stay fixed on an older stream.

See also the [How do I use subscription-manager in a kickstart file?](#) article on the Red Hat Customer Portal for additional information about using **subscription-manager** in a Kickstart **%post** section.

G.4. ANACONDA CONFIGURATION SECTION

Additional installation options can be configured in the **%anaconda** section of your Kickstart file. This section controls the behavior of the user interface of the installation system.

This section must be placed towards the end of the Kickstart file, after Kickstart commands, and must start with **%anaconda** and end with **%end**.

Currently, the only command that can be used in the **%anaconda** section is **pwpolicy**.

Example G.1. Sample %anaconda script

The following is an example %anaconda section:

```

%anaconda
pwpolicy root --minlen=10 --strict
%end

```

This example **%anaconda** section sets a password policy which requires that the root password be at least 10 characters long, and strictly forbids passwords which do not match this requirement.

G.5. KICKSTART ERROR HANDLING SECTION

Starting with Red Hat Enterprise Linux 7, Kickstart installations can contain custom scripts which are run when the installation program encounters a fatal error. For example, an error in a package that has been requested for installation, failure to start VNC when specified, or an error when scanning storage devices. Installation cannot continue after such an error has occurred. The installation program will run all **%onerror** scripts in the order they are provided in the Kickstart file. In addition, **%onerror** scripts will be run in the event of a traceback.

Each **%onerror** script is required to end with **%end**.

Error handling sections accept the following options:

--erroronfail

Display an error and halt the installation if the script fails. The error message will direct you to where the cause of the failure is logged.

--interpreter=

Allows you to specify a different scripting language, such as Python. For example:

```
%onerror --interpreter=/usr/libexec/platform-python
```

Any scripting language available on the system can be used; in most cases, these are **/usr/bin/sh**, **/usr/bin/bash**, and **/usr/libexec/platform-python**.

Note that the **platform-python** interpreter uses Python version 3.6. You must change your Python scripts from previous RHEL versions for the new path and version. Additionally, **platform-python** is meant for system tools: Use the **python36** package outside the installation environment. For more details about Python in Red Hat Enterprise Linux 8, see [Introduction to Python](#) in [Configuring basic system settings](#).

--log=

Logs the script's output into the specified log file.

G.6. KICKSTART ADD-ON SECTIONS

Starting with Red Hat Enterprise Linux 7, Kickstart installations support add-ons. These add-ons can expand the basic Kickstart (and Anaconda) functionality in many ways.

To use an add-on in your Kickstart file, use the **%addon addon_name options** command, and finish the command with an **%end** statement, similar to pre-installation and post-installation script sections. For example, if you want to use the Kdump add-on, which is distributed with Anaconda by default, use the following commands:

```
%addon com_redhat_kdump --enable --reserve-mb=auto  
%end
```

The **%addon** command does not include any options of its own – all options are dependent on the actual add-on.

APPENDIX H. KICKSTART COMMANDS AND OPTIONS REFERENCE

This reference is a complete list of all Kickstart commands supported by the Red Hat Enterprise Linux installation program. The commands are sorted alphabetically in a few broad categories. If a command can fall under multiple categories, it is listed in all of them.

H.1. KICKSTART CHANGES

The following sections describe the changes in Kickstart commands and options in Red Hat Enterprise Linux 8.

H.1.1. auth or authconfig is deprecated in RHEL 8

The **auth** or **authconfig** Kickstart command is deprecated in Red Hat Enterprise Linux 8 because the **authconfig** tool and package have been removed.

Similarly to **authconfig** commands issued on command line, **authconfig** commands in Kickstart scripts now use the **authselect-compat** tool to run the new **authselect** tool. For a description of this compatibility layer and its known issues, see the manual page **authselect-migration(7)**. The installation program will automatically detect use of the deprecated commands and install on the system the **authselect-compat** package to provide the compatibility layer.

H.1.2. Kickstart no longer supports Btrfs

The Btrfs file system is not supported in Red Hat Enterprise Linux 8. As a result, the Graphical User Interface (GUI) and the Kickstart commands no longer support Btrfs.

H.1.3. Using Kickstart files from previous RHEL releases

If you are using Kickstart files from previous RHEL releases, see the *Repositories* section of the [Considerations in adopting RHEL 8](#) document for more information about the Red Hat Enterprise Linux 8 BaseOS and AppStream repositories.

H.1.4. Deprecated Kickstart commands and options

The following Kickstart commands and options have been deprecated in Red Hat Enterprise Linux 8.

Where only specific options are listed, the base command and its other options are still available and not deprecated.

- **auth** or **authconfig** – use **authselect** instead
- **device**
- **deviceprobe**
- **dmraid**
- **install** – use the subcommands or methods directly as commands
- **multipath**
- **bootloader --upgrade**

- **ignoredisk --interactive**
- **partition --active**
- **reboot --kexec**

Except the **auth** or **authconfig** command, using the commands in Kickstart files prints a warning in the logs.

You can turn the deprecated command warnings into errors with the **inst.ksstrict** boot option, except for the **auth** or **authconfig** command.

H.1.5. Removed Kickstart commands and options

The following Kickstart commands and options have been completely removed in Red Hat Enterprise Linux 8. Using them in Kickstart files will cause an error.

- **upgrade** (This command had already previously been deprecated.)
- **btrfs**
- **part/partition btrfs**
- **part --fstype btrfs** or **partition --fstype btrfs**
- **logvol --fstype btrfs**
- **raid --fstype btrfs**
- **unsupported_hardware**

Where only specific options and values are listed, the base command and its other options are still available and not removed.

H.1.6. New Kickstart commands and options

The following commands and options were added in Red Hat Enterprise Linux 8.2.

RHEL 8.2

- **rhsm**
- **zipl**

The following commands and options were added in Red Hat Enterprise Linux 8.

RHEL 8.0

- **authselect**
- **module**

H.2. KICKSTART COMMANDS FOR INSTALLATION PROGRAM CONFIGURATION AND FLOW CONTROL

The Kickstart commands in this list control the mode and course of installation, and what happens at its end.

H.2.1. autostep

The **autostep** Kickstart command is optional. This option makes the installation program step through every screen, displaying each briefly. Normally, Kickstart installations skip unnecessary screens.

Syntax

```
autostep [--autoscreenshot]
```

Options

- **--autoscreenshot** – Take a screenshot at every step during installation. These screenshots are stored in **/tmp/anaconda-screenshots** during the installation, and after the installation finishes you can find them in **/root/anaconda-screenshots**.

Each screen is only captured right before the installation program switches to the next one. This is important, because if you do not use all required Kickstart options and the installation therefore does not begin automatically, you can go to the screens which were not automatically configured, perform any configuration you want. Then, when you click **Done** to continue, the screen is captured including the configuration you just provided.

Notes

- This option should not be used when deploying a system because it can disrupt package installation.

H.2.2. cdrom

The **cdrom** Kickstart command is optional. It performs the installation from the first optical drive on the system.

Syntax

```
cdrom
```

Notes

- Previously, the **cdrom** command had to be used together with the **install** command. The **install** command has been deprecated and **cdrom** can be used on its own, because it implies **install**.
- This command has no options.
- To actually run the installation, one of **cdrom**, **harddrive**, **hmc**, **nfs**, **liveimg**, or **url** must be specified.

H.2.3. cmdline

The **cmdline** Kickstart command is optional. It performs the installation in a completely non-interactive command line mode. Any prompt for interaction halts the installation.

Syntax

cmdline**Notes**

- For a fully automatic installation, you must either specify one of the available modes (**graphical**, **text**, or **cmdline**) in the Kickstart file, or you must use the **console=** boot option. If no mode is specified, the system will use graphical mode if possible, or prompt you to choose from VNC and text mode.
- This command has no options.
- This mode is useful on IBM Z systems with the x3270 terminal.

H.2.4. driverdisk

The **driverdisk** Kickstart command is optional. Use it to provide additional drivers to the installation program.

Driver disks can be used during Kickstart installations to provide additional drivers not included by default. You must copy the driver disks contents to the root directory of a partition on the system's hard drive. Then, you must use the **driverdisk** command to specify that the installation program should look for a driver disk and its location.

Syntax

```
driverdisk [partition]--source=url--biospart=biospart
```

Options

You must specify the location of driver disk in one way out of these:

- partition* - Partition containing the driver disk. Note that the partition must be specified as a full path (for example, **/dev/sdb1**), *not* just the partition name (for example, **sdb1**).
- source=** - URL for the driver disk. Examples include:

```
driverdisk --source=ftp://path/to/dd.img
driverdisk --source=http://path/to/dd.img
driverdisk --source=nfs:host:/path/to/dd.img
```

- biospart=** - BIOS partition containing the driver disk (for example, **82p2**).

Notes

Driver disks can also be loaded from a hard disk drive or a similar device instead of being loaded over the network or from **initrd**. Follow this procedure:

- Load the driver disk on a hard disk drive, a USB or any similar device.
- Set the label, for example, *DD*, to this device.
- Add the following line to your Kickstart file:

```
driverdisk LABEL=DD:/e1000.rpm
```

Replace *DD* with a specific label and replace *dd.rpm* with a specific name. Use anything supported by the **inst.repo** command instead of *LABEL* to specify your hard disk drive.

H.2.5. eula

The **eula** Kickstart command is optional. Use this option to accept the End User License Agreement (EULA) without user interaction. Specifying this option prevents Initial Setup from prompting you to accept the license agreement after you finish the installation and reboot the system for the first time. See the [Completing initial setup](#) section of the *Performing a standard RHEL installation* document for more information.

Syntax

```
  eula
```

Options

- **--agreed** (required) – Accept the EULA. This option must always be used, otherwise the **eula** command is meaningless.
- This command has no options.

H.2.6. firstboot

The **firstboot** Kickstart command is optional. It determines whether the **Initial Setup** application starts the first time the system is booted. If enabled, the **initial-setup** package must be installed. If not specified, this option is disabled by default.

Syntax

```
  firstboot OPTIONS
```

Options

- **--enable** or **--enabled** – Initial Setup is started the first time the system boots.
- **--disable** or **--disabled** – Initial Setup is not started the first time the system boots.
- **--reconfig** – Enable the Initial Setup to start at boot time in reconfiguration mode. This mode enables the language, mouse, keyboard, root password, security level, time zone and networking configuration options in addition to the default ones.

H.2.7. graphical

The **graphical** Kickstart command is optional. It performs the installation in graphical mode. This is the default.

Syntax

```
  graphical [--non-interactive]
```

Options

- **--non-interactive** - Performs the installation in a completely non-interactive mode. This mode will terminate the installation when user interaction is required.

Notes

- For a fully automatic installation, you must either specify one of the available modes (**graphical**, **text**, or **cmdline**) in the Kickstart file, or you must use the **console=** boot option. If no mode is specified, the system will use graphical mode if possible, or prompt you to choose from VNC and text mode.

H.2.8. halt

The **halt** Kickstart command is optional.

Halt the system after the installation has successfully completed. This is similar to a manual installation, where Anaconda displays a message and waits for the user to press a key before rebooting. During a Kickstart installation, if no completion method is specified, this option is used as the default.

Syntax

```
halt
```

Notes

- The **halt** command is equivalent to the **shutdown -H** command. For more details, see the *shutdown(8)* man page.
- For other completion methods, see the **poweroff**, **reboot**, and **shutdown** commands.
- This command has no options.

H.2.9. harddrive

The **harddrive** Kickstart command is optional. It performs the installation from a Red Hat installation tree or full installation ISO image on a local drive. The drive must contain a file system the installation program can mount: **ext2**, **ext3**, **ext4**, **vfat**, or **xfs**.

Syntax

```
harddrive OPTIONS
```

Options

- **--biospart=** - BIOS partition to install from (such as **82**).
- **--partition=** - Partition to install from (such as **sdb2**).
- **--dir=** - Directory containing the **variant** directory of the installation tree, or the ISO image of the full installation DVD.

Example

```
harddrive --partition=hdb2 --dir=/tmp/install-tree
```

Notes

- Previously, the **harddrive** command had to be used together with the **install** command. The **install** command has been deprecated and **harddrive** can be used on its own, because it implies **install**.
- To actually run the installation, one of **cdrom**, **harddrive**, **hmc**, **nfs**, **liveimg**, or **url** must be specified.

H.2.10. **install** (deprecated)



IMPORTANT

The **install** Kickstart command is deprecated in Red Hat Enterprise Linux 8. Use its methods as separate commands.

The **install** Kickstart command is optional. It specifies the default installation mode.

Syntax

```
install
  installation_method
```

Notes

- The **install** command must be followed by an installation method command. The installation method command must be on a separate line.
- The methods include:
 - **cdrom**
 - **harddrive**
 - **hmc**
 - **nfs**
 - **liveimg**
 - **url**

For details about the methods, see their separate reference pages.

H.2.11. **liveimg**

The **liveimg** Kickstart command is optional. It performs the installation from a disk image instead of packages.

Syntax

```
liveimg --url=SOURCE [OPTIONS]
```

Mandatory options

- **--url=** - The location to install from. Supported protocols are **HTTP**, **HTTPS**, **FTP**, and **file**.

Optional options

- **--url=** - The location to install from. Supported protocols are **HTTP**, **HTTPS**, **FTP**, and **file**.
- **--proxy=** - Specify an **HTTP**, **HTTPS** or **FTP** proxy to use while performing the installation.
- **--checksum=** - An optional argument with the **SHA256** checksum of the image file, used for verification.
- **--noverifyssl** - Disable SSL verification when connecting to an **HTTPS** server.

Example

```
liveimg --url=file:///images/install/squashfs.img --
checksum=03825f567f17705100de3308a20354b4d81ac9d8bed4bb4692b2381045e56197 --
noverifyssl
```

Notes

- The image can be the **squashfs.img** file from a live ISO image, a compressed tar file (**.tar**, **.tbz**, **.tgz**, **.txz**, **.tar.bz2**, **.tar.gz**, or **.tar.xz**), or any file system that the installation media can mount. Supported file systems are **ext2**, **ext3**, **ext4**, **vfat**, and **xfs**.
- When using the **liveimg** installation mode with a driver disk, drivers on the disk will not automatically be included in the installed system. If necessary, these drivers should be installed manually, or in the **%post** section of a kickstart script.
- Previously, the **liveimg** command had to be used together with the **install** command. The **install** command has been deprecated and **liveimg** can be used on its own, because it implies **install**.
- To actually run the installation, one of **cdrom**, **harddrive**, **hmc**, **nfs**, **liveimg**, or **url** must be specified.

H.2.12. logging

The **logging** Kickstart command is optional. It controls the error logging of Anaconda during installation. It has no effect on the installed system.



NOTE

Logging is supported over TCP only. For remote logging, ensure that the port number that you specify in **--port=** option is open on the remote server. The default port is 514.

Syntax

```
logging OPTIONS
```

Optional options

- **--host=** - Send logging information to the given remote host, which must be running a syslogd process configured to accept remote logging.
- **--port=** - If the remote syslogd process uses a port other than the default, set it using this option.
- **--level=** - Specify the minimum level of messages that appear on tty3. All messages are still sent to the log file regardless of this level, however. Possible values are **debug**, **info**, **warning**, **error**, or **critical**.

H.2.13. mediacheck

The **mediacheck** Kickstart command is optional. This command forces the installation program to perform a media check before starting the installation. This command requires that installations be attended, so it is disabled by default.

Syntax

```
mediacheck
```

Notes

- This Kickstart command is equivalent to the **rd.live.check** boot option.
- This command has no options.

H.2.14. nfs

The **nfs** Kickstart command is optional. It performs the installation from a specified NFS server.

Syntax

```
nfs OPTIONS
```

Options

- **--server=** - Server from which to install (host name or IP).
- **--dir=** - Directory containing the **variant** directory of the installation tree.
- **--opts=** - Mount options to use for mounting the NFS export. (optional)

Example

```
nfs --server=nfsserver.example.com --dir=/tmp/install-tree
```

Notes

- Previously, the **nfs** command had to be used together with the **install** command. The **install** command has been deprecated and **nfs** can be used on its own, because it implies **install**.
- To actually run the installation, one of **cdrom**, **harddrive**, **hmc**, **nfs**, **liveimg**, or **url** must be specified.

H.2.15. `ostreesetup`

The **`ostreesetup`** Kickstart command is optional. It is used to set up OSTree-based installations.

Syntax

```
ostreesetup --osname=OSNAME [--remote=REMOTE] --url=URL --ref=REF [--nogpg]
```

Mandatory options:

- **--osname=OSNAME** - Management root for OS installation.
- **--url=URL** - URL of the repository to install from.
- **--ref=REF** - Name of the branch from the repository to be used for installation.

Optional options:

- **--remote=REMOTE** - Management root for OS installation.
- **--nogpg** - Disable GPG key verification.

Notes

- For more information about the OSTree tools, see the upstream documentation: <https://ostree.readthedocs.io/en/latest/>

H.2.16. `poweroff`

The **`poweroff`** Kickstart command is optional. It shuts down and powers off the system after the installation has successfully completed. Normally during a manual installation, Anaconda displays a message and waits for the user to press a key before rebooting.

Syntax

```
poweroff
```

Notes

- The **`poweroff`** option is equivalent to the **`shutdown -P`** command. For more details, see the `shutdown(8)` man page.
- For other completion methods, see the **`halt`**, **`reboot`**, and **`shutdown`** Kickstart commands. The **`halt`** option is the default completion method if no other methods are explicitly specified in the Kickstart file.
- The **`poweroff`** command is highly dependent on the system hardware in use. Specifically, certain hardware components such as the BIOS, APM (advanced power management), and ACPI (advanced configuration and power interface) must be able to interact with the system kernel. Consult your hardware documentation for more information on your system's APM/ACPI abilities.
- This command has no options.

H.2.17. reboot

The **reboot** Kickstart command is optional. It instructs the installation program to reboot after the installation is successfully completed (no arguments). Normally, Kickstart displays a message and waits for the user to press a key before rebooting.

Syntax

```
reboot OPTIONS
```

Options

- **--eject** – Attempt to eject the bootable media (DVD, USB, or other media) before rebooting.
- **--kexec** – Uses the **kexec** system call instead of performing a full reboot, which immediately loads the installed system into memory, bypassing the hardware initialization normally performed by the BIOS or firmware.

IMPORTANT

This option is deprecated and available as a Technology Preview only. For information on Red Hat scope of support for Technology Preview features, see the [Technology Preview Features Support Scope](#) document.

When **kexec** is used, device registers (which would normally be cleared during a full system reboot) might stay filled with data, which could potentially create issues for some device drivers.

Notes

- Use of the **reboot** option *might* result in an endless installation loop, depending on the installation media and method.
- The **reboot** option is equivalent to the **shutdown -r** command. For more details, see the [shutdown\(8\)](#) man page.
- Specify **reboot** to automate installation fully when installing in command line mode on IBM Z.
- For other completion methods, see the **halt**, **poweroff**, and **shutdown** Kickstart options. The **halt** option is the default completion method if no other methods are explicitly specified in the Kickstart file.

H.2.18. rhsm

The **rhsm** Kickstart command is optional. It instructs the installation program to register and install RHEL from the CDN.



NOTE

The **rhsm** Kickstart command removes the requirement of using custom **%post** scripts when registering the system.

Options

- **--organization=** – Uses the organization id to register and install RHEL from the CDN.
- **--activation-key=** – Uses the activation key to register and install RHEL from the CDN. Multiple keys can be used, as long as the activation keys are registered to your subscription.
- **--connect-to-insights** – Connects the target system to Red Hat Insights.
- **--proxy=** – Sets the HTTP proxy.
- **--server-hostname=** – Sets the server hostname. Use this option if you are running Satellite Server or performing internal testing.
- **--rhsm-baseurl=** – Sets the rhsm baseurl option. Use this option if you are running Satellite Server or performing internal testing.



NOTE

The Server hostname does not require the HTTP protocol, for example **--server-hostname="nameofhost.com"**. The rhsm baseurl does require the HTTP protocol, for example **--rhsm-baseurl="http://nameofhost.com"**.

H.2.19. shutdown

The **shutdown** Kickstart command is optional. It shuts down the system after the installation has successfully completed.

Syntax

```
shutdown
```

Notes

- The **shutdown** Kickstart option is equivalent to the **shutdown** command. For more details, see the *shutdown(8)* man page.
- For other completion methods, see the **halt**, **poweroff**, and **reboot** Kickstart options. The **halt** option is the default completion method if no other methods are explicitly specified in the Kickstart file.
- This command has no options.

H.2.20. sshpw

The **sshpw** Kickstart command is optional.

During the installation, you can interact with the installation program and monitor its progress over an **SSH** connection. Use the **sshpw** command to create temporary accounts through which to log on. Each instance of the command creates a separate account that exists only in the installation environment. These accounts are not transferred to the installed system.

Syntax

```
sshpw --username=name [OPTIONS] password
```

Mandatory options

- **--username**=*name* - Provides the name of the user. This option is required.
- *password* - The password to use for the user. This option is required.

Optional options

- **--iscrypted** - If this option is present, the password argument is assumed to already be encrypted. This option is mutually exclusive with **--plaintext**. To create an encrypted password, you can use Python:

```
$ python3 -c 'import crypt,getpass;pw=getpass.getpass();print(crypt.crypt(pw) if (pw==getpass.getpass("Confirm: ")) else exit())'
```

This generates a sha512 crypt-compatible hash of your password using a random salt.

- **--plaintext** - If this option is present, the password argument is assumed to be in plain text. This option is mutually exclusive with **--iscrypted**
- **--lock** - If this option is present, this account is locked by default. This means that the user will not be able to log in from the console.
- **--sshkey** - If this option is present, then the *<password>* string is interpreted as an ssh key value.

Notes

- By default, the **ssh** server is not started during the installation. To make **ssh** available during the installation, boot the system with the kernel boot option **inst.sshd**.
- If you want to disable root **ssh** access, while allowing another user **ssh** access, use the following:

```
sshpw --username=example_username example_password --plaintext
sshpw --username=root example_password --lock
```

- To simply disable root **ssh** access, use the following:

```
sshpw --username=root example_password --lock
```

H.2.21. text

The **text** Kickstart command is optional. It performs the Kickstart installation in text mode. Kickstart installations are performed in graphical mode by default.

Syntax

```
text [--non-interactive]
```

Options

- **--non-interactive** - Performs the installation in a completely non-interactive mode. This mode will terminate the installation when user interaction is required.

Notes

- Note that for a fully automatic installation, you must either specify one of the available modes (**graphical**, **text**, or **cmdline**) in the Kickstart file, or you must use the **console=** boot option. If no mode is specified, the system will use graphical mode if possible, or prompt you to choose from VNC and text mode.

H.2.22. url

The **url** Kickstart command is optional. It performs the installation from an installation tree image on a remote server using **FTP**, **HTTP**, or **HTTPS**.

Syntax

```
url --url=FROM [OPTIONS]
```

Mandatory options

- **--url=FROM** - The location to install from. Supported protocols are **HTTP**, **HTTPS**, **FTP**, and **file**.

Optional options

- **--mirrorlist=** - The mirror URL to install from.
- **--proxy=** - Specify an **HTTP**, **HTTPS** or **FTP** proxy to use while performing the installation.
- **--noverifyssl** - Disable SSL verification when connecting to an **HTTPS** server.
- **--metalink=URL** - Specify the metalink URL to install from. Variable substitution is done for **\$releasever** and **\$basearch** in the **URL**.

Examples

- To install from a **HTTP** server:

```
url --url=http://server/path
```

- To install from a **FTP** server:

```
url --url=ftp://username:password@server/path
```

- To install from a local file:

```
liveimg --url=file:///images/install/squashfs.img --noverifyssl
```

Notes

- Previously, the **url** command had to be used together with the **install** command. The **install** command has been deprecated and **url** can be used on its own, because it implies **install**.
- To actually run the installation, one of **cdrom**, **harddrive**, **hmc**, **nfs**, **liveimg**, or **url** must be specified.

H.2.23. vnc

The **vnc** Kickstart command is optional. It allows the graphical installation to be viewed remotely through VNC.

This method is usually preferred over text mode, as there are some size and language limitations in text installations. With no additional options, this command starts a VNC server on the installation system with no password and displays the details required to connect to it.

Syntax

```
vnc [--host=host_name] [--port=port] [--password=password]
```

Options

- **--host=** - Connect to the VNC viewer process listening on the given host name.
- **--port=** - Provide a port that the remote VNC viewer process is listening on. If not provided, Anaconda uses the VNC default port of 5900.
- **--password=** - Set a password which must be provided to connect to the VNC session. This is optional, but recommended.

Additional resources

- For more information about VNC installations, including instructions on how to connect to the installation system, see [Performing a remote RHEL installation using VNC](#) .

H.2.24. %include

The **%include** Kickstart command is optional.

Use the **%include** command to include the contents of another file in the Kickstart file as if the contents were at the location of the **%include** command in the Kickstart file.

This inclusion is evaluated only after the **%pre** script sections and can thus be used to include files generated by scripts in the **%pre** sections. To include files before evaluation of **%pre** sections, use the **%ksappend** command.

Syntax

```
%include path/to/file
```

H.2.25. %ksappend

The **%ksappend** Kickstart command is optional.

Use the **%ksappend** command to include the contents of another file in the Kickstart file as if the contents were at the location of the **%ksappend** command in the Kickstart file.

This inclusion is evaluated before the **%pre** script sections, unlike inclusion with the **%include** command.

Syntax

```
%ksappend path/to/file
```

H.3. KICKSTART COMMANDS FOR SYSTEM CONFIGURATION

The Kickstart commands in this list configure further details on the resulting system such as users, repositories, or services.

H.3.1. auth or authconfig (deprecated)



IMPORTANT

Use the new **authselect** command instead of the deprecated **auth** or **authconfig** Kickstart command. **auth** and **authconfig** are available only for limited backwards compatibility.

The **auth** or **authconfig** Kickstart command is optional. It sets up the authentication options for the system using the **authconfig** tool, which can also be run on the command line after the installation finishes.

Syntax

```
authconfig [OPTIONS]
```

Notes

- Previously, the **auth** or **authconfig** Kickstart commands called the **authconfig** tool. This tool has been deprecated in Red Hat Enterprise Linux 8. These Kickstart commands now use the **authselect-compat** tool to call the new **authselect** tool. For a description of the compatibility layer and its known issues, see the manual page *authselect-migration(7)*. The installation program will automatically detect use of the deprecated commands and install on the system the **authselect-compat** package to provide the compatibility layer.
- Passwords are shadowed by default.
- When using OpenLDAP with the **SSL** protocol for security, make sure that the **SSLv2** and **SSLv3** protocols are disabled in the server configuration. This is due to the POODLE SSL vulnerability (CVE-2014-3566). See <https://access.redhat.com/solutions/1234843> for details.

H.3.2. authselect

The **authselect** Kickstart command is optional. It sets up the authentication options for the system using the **authselect** command, which can also be run on the command line after the installation finishes.

Syntax

```
authselect [OPTIONS]
```

Notes

- This command passes all options to the **authselect** command. Refer to the *authselect(8)* manual page and the **authselect --help** command for more details.

- This command replaces the deprecated **auth** or **authconfig** commands deprecated in Red Hat Enterprise Linux 8 together with the **authconfig** tool.
- Passwords are shadowed by default.
- When using OpenLDAP with the **SSL** protocol for security, make sure that the **SSLv2** and **SSLv3** protocols are disabled in the server configuration. This is due to the POODLE SSL vulnerability (CVE-2014-3566). See <https://access.redhat.com/solutions/1234843> for details.

H.3.3. firewall

The **firewall** Kickstart command is optional. It specifies the firewall configuration for the installed system.

Syntax

```
firewall --enabled|--disabled [incoming] [OPTIONS]
```

Mandatory options

- **--enabled** or **--enable** - Reject incoming connections that are not in response to outbound requests, such as DNS replies or DHCP requests. If access to services running on this machine is needed, you can choose to allow specific services through the firewall.
- **--disabled** or **--disable** - Do not configure any iptables rules.

Optional options

- **--trust** - Listing a device here, such as **em1**, allows all traffic coming to and from that device to go through the firewall. To list more than one device, use the option more times, such as **--trust em1 --trust em2**. Do not use a comma-separated format such as **--trust em1, em2**.
- **--remove-service** - Do not allow services through the firewall.
- *incoming* - Replace with one or more of the following to allow the specified services through the firewall.
 - **--ssh**
 - **--smtp**
 - **--http**
 - **--ftp**
- **--port=** - You can specify that ports be allowed through the firewall using the `port:protocol` format. For example, to allow IMAP access through your firewall, specify **imap:tcp**. Numeric ports can also be specified explicitly; for example, to allow UDP packets on port 1234 through, specify **1234:udp**. To specify multiple ports, separate them by commas.
- **--service=** - This option provides a higher-level way to allow services through the firewall. Some services (like **cups**, **avahi**, and so on.) require multiple ports to be open or other special configuration in order for the service to work. You can specify each individual port with the **--port** option, or specify **--service=** and open them all at once.

Valid options are anything recognized by the **firewall-offline-cmd** program in the **firewalld** package. If the **firewalld** service is running, **firewall-cmd --get-services** provides a list of known service names.

- **--use-system-defaults** - Do not configure the firewall at all. This option instructs anaconda to do nothing and allows the system to rely on the defaults that were provided with the package or ostree. If this option is used with other options then all other options will be ignored.

H.3.4. group

The **group** Kickstart command is optional. It creates a new user group on the system.

```
group --name=name [--gid=gid]
```

Mandatory options

- **--name=** - Provides the name of the group.

Optional options

- **--gid=** - The group's GID. If not provided, defaults to the next available non-system GID.

Notes

- If a group with the given name or GID already exists, this command fails.
- The **user** command can be used to create a new group for the newly created user.

H.3.5. keyboard (required)

The **keyboard** Kickstart command is required. It sets one or more available keyboard layouts for the system.

Syntax

```
keyboard --vckeymap|--xlayouts OPTIONS
```

Options

- **--vckeymap=** - Specify a **VConsole** keymap which should be used. Valid names correspond to the list of files in the **/usr/lib/kbd/keymaps/xkb/** directory, without the **.map.gz** extension.
- **--xlayouts=** - Specify a list of X layouts that should be used as a comma-separated list without spaces. Accepts values in the same format as **setxkbmap(1)**, either in the **layout** format (such as **cz**), or in the **layout (variant)** format (such as **cz (qwerty)**).
All available layouts can be viewed on the **xkeyboard-config(7)** man page under **Layouts**.
- **--switch=** - Specify a list of layout-switching options (shortcuts for switching between multiple keyboard layouts). Multiple options must be separated by commas without spaces. Accepts values in the same format as **setxkbmap(1)**.
Available switching options can be viewed on the **xkeyboard-config(7)** man page under **Options**.

Notes

- Either the **--vckeymap=** or the **--xlayouts=** option must be used.

Example

The following example sets up two keyboard layouts (**English (US)** and **Czech (qwerty)**) using the **--xlayouts=** option, and allows to switch between them using **Alt+Shift**:

```
keyboard --xlayouts=us,'cz (qwerty)' --switch=grp:alt_shift_toggle
```

H.3.6. lang (required)

The **lang** Kickstart command is required. It sets the language to use during installation and the default language to use on the installed system.

Syntax

```
lang language [--addsupport=language,...]
```

Mandatory options

- language*** - Install support for this language and set it as system default.

Optional options

- addsupport=** - Add support for additional languages. Takes the form of comma-separated list without spaces. For example:

```
lang en_US --addsupport=cs_CZ,de_DE,en_UK
```

Notes

- The **locale -a | grep _** or **localectl list-locales | grep _** commands return a list of supported locales.
- Certain languages (for example, Chinese, Japanese, Korean, and Indic languages) are not supported during text-mode installation. If you specify one of these languages with the **lang** command, the installation process continues in English, but the installed system uses your selection as its default language.

Example

To set the language to English, the Kickstart file should contain the following line:

```
lang en_US
```

H.3.7. module

The **module** Kickstart command is optional. Use this command to enable a package module stream within kickstart script.

Syntax

-

```
module --name=NAME [--stream=STREAM]
```

Mandatory options

- **--name=** - Specifies the name of the module to enable. Replace *NAME* with the actual name.

Optional options

- **--stream=** - Specifies the name of the module stream to enable. Replace *STREAM* with the actual name.

You do not need to specify this option for modules with a default stream defined. For modules without a default stream, this option is mandatory and leaving it out results in an error. Enabling a module multiple times with different streams is not possible.

Notes

- Using a combination of this command and the **%packages** section allows you to install packages provided by the enabled module and stream combination, without specifying the module and stream explicitly. Modules must be enabled before package installation. After enabling a module with the **module** command, you can install the packages enabled by this module by listing them in the **%packages** section.
- A single **module** command can enable only a single module and stream combination. To enable multiple modules, use multiple **module** commands. Enabling a module multiple times with different streams is not possible.
- In Red Hat Enterprise Linux 8, modules are present only in the AppStream repository. To list available modules, use the **yum module list** command on an installed Red Hat Enterprise Linux 8 system with a valid subscription.

Additional resources

- For more information about modules and streams, see the [Installing, managing, and removing user-space components](#) document.

H.3.8. repo

The **repo** Kickstart command is optional. It configures additional yum repositories that can be used as sources for package installation. You can add multiple **repo** lines.

Syntax

```
repo --name=repoid [--baseurl=url|--mirrorlist=url|--metalink=url] [OPTIONS]
```

Mandatory options

- **--name=** - The repository id. This option is required. If a repository has a name which conflicts with another previously added repository, it is ignored. Because the installation program uses a list of preset repositories, this means that you cannot add repositories with the same names as the preset ones.

URL options

These options are mutually exclusive and optional. The variables that can be used in yum repository configuration files are not supported here. You can use the strings **\$releasever** and **\$basearch** which are replaced by the respective values in the URL.

- **--baseurl=** - The URL to the repository.
- **--mirrorlist=** - The URL pointing at a list of mirrors for the repository.
- **--metalink=** - The URL with metalink for the repository.

Optional options

- **--install** - Save the provided repository configuration on the installed system in the **/etc/yum.repos.d/** directory. Without using this option, a repository configured in a Kickstart file will only be available during the installation process, not on the installed system.
- **--cost=** - An integer value to assign a cost to this repository. If multiple repositories provide the same packages, this number is used to prioritize which repository will be used before another. Repositories with a lower cost take priority over repositories with higher cost.
- **--excludedpkgs=** - A comma-separated list of package names that must *not* be pulled from this repository. This is useful if multiple repositories provide the same package and you want to make sure it comes from a particular repository. Both full package names (such as **publican**) and globs (such as **gnome-***) are accepted.
- **--includedpkgs=** - A comma-separated list of package names and globs that are allowed to be pulled from this repository. Any other packages provided by the repository will be ignored. This is useful if you want to install just a single package or set of packages from a repository while excluding all other packages the repository provides.
- **--proxy=[protocol://][username[:password]@]host[:port]** - Specify an HTTP/HTTPS/FTP proxy to use just for this repository. This setting does not affect any other repositories, nor how the **install.img** is fetched on HTTP installations.
- **--noverifyssl** - Disable SSL verification when connecting to an **HTTPS** server.

Notes

- Repositories used for installation must be stable. The installation can fail if a repository is modified before the installation concludes.

H.3.9. **rootpw** (required)

The **rootpw** Kickstart command is required. It sets the system's root password to the *password* argument.

Syntax

```
rootpw [--iscrypted|--plaintext] [--lock] password
```

Mandatory options

- *password* - Password specification. Either plain text or encrypted string. See **--iscrypted** and **--plaintext** below.

Options

- **--iscrypted** - If this option is present, the password argument is assumed to already be encrypted. This option is mutually exclusive with **--plaintext**. To create an encrypted password, you can use python:

```
$ python -c 'import crypt,getpass;pw=getpass.getpass();print(crypt.crypt(pw) if (pw==getpass.getpass("Confirm: ")) else exit())'
```

This generates a sha512 crypt-compatible hash of your password using a random salt.

- **--plaintext** - If this option is present, the password argument is assumed to be in plain text. This option is mutually exclusive with **--iscrypted**.
- **--lock** - If this option is present, the root account is locked by default. This means that the root user will not be able to log in from the console. This option will also disable the **Root Password** screens in both the graphical and text-based manual installation.

H.3.10. selinux

The **selinux** Kickstart command is optional. It sets the state of SELinux on the installed system. The default SELinux policy is **enforcing**.

Syntax

```
selinux [--disabled|--enforcing|--permissive]
```

Options

- **--enforcing** - Enables SELinux with the default targeted policy being **enforcing**.
- **--permissive** - Outputs warnings based on the SELinux policy, but does not actually enforce the policy.
- **--disabled** - Disables SELinux completely on the system.

Additional resources

For more information regarding SELinux, see the [Using SELinux](#) document.

H.3.11. services

The **services** Kickstart command is optional. It modifies the default set of services that will run under the default systemd target. The list of disabled services is processed before the list of enabled services. Therefore, if a service appears on both lists, it will be enabled.

Syntax

```
services [--disabled=/list] [--enabled=/list]
```

Options

- **--disabled=** - Disable the services given in the comma separated list.
- **--enabled=** - Enable the services given in the comma separated list.

Notes

- Do not include spaces in the list of services. If you do, Kickstart will enable or disable only the services up to the first space. For example:

```
services --disabled=auditd, cups,smartd, nfslock
```

That disables only the **auditd** service. To disable all four services, this entry must include no spaces:

```
services --disabled=auditd,cups,smartd,nfslock
```

H.3.12. skipx

The **skipx** Kickstart command is optional. If present, X is not configured on the installed system.

If you install a display manager among your package selection options, this package creates an X configuration, and the installed system defaults to **graphical.target**. That overrides the effect of the **skipx** option.

Syntax

```
skipx
```

Notes

- This command has no options.

H.3.13. sshkey

The **sshkey** Kickstart command is optional. It adds a SSH key to the **authorized_keys** file of the specified user on the installed system.

Syntax

```
sshkey --username=user KEY
```

Mandatory options

- **--username=** - The user for which the key will be installed.
- **KEY** - The SSH key.

H.3.14. syspurpose

The **syspurpose** Kickstart command is optional. Use it to set the system purpose which describes how the system will be used after installation. This information helps apply the correct subscription entitlement to the system.

Syntax

```
syspurpose [OPTIONS]
```

Options

- **--role=** - Set the intended system role. Available values are:
 - Red Hat Enterprise Linux Server
 - Red Hat Enterprise Linux Workstation
 - Red Hat Enterprise Linux Compute Node
- **--sla=** - Set the Service Level Agreement. Available values are:
 - Premium
 - Standard
 - Self-Support
- **--usage=** - The intended usage of the system. Available values are:
 - Production
 - Disaster Recovery
 - Development/Test
- **--addon=** - Specifies additional layered products or features. You can use this option multiple times.

Notes

- Enter the values with spaces and enclose them in double quotes:


```
syspurpose --role="Red Hat Enterprise Linux Server"
```
- While it is strongly recommended that you configure System Purpose, it is an optional feature of the Red Hat Enterprise Linux installation program. If you want to enable System Purpose after the installation completes, you can do so using the **syspurpose** command-line tool.

H.3.15. **timezone** (required)

The **timezone** Kickstart command is required. It sets the system time zone.

Syntax

```
timezone timezone [OPTIONS]
```

Mandatory options

- *timezone* - the time zone to set for the system.

Optional options

- **--utc** - If present, the system assumes the hardware clock is set to UTC (Greenwich Mean) time.
- **--nontp** - Disable the NTP service automatic starting.

- **--ntpservers=** - Specify a list of NTP servers to be used as a comma-separated list without spaces.

Notes

In Red Hat Enterprise Linux 8, time zone names are validated using the **pytz.all_timezones** list, provided by the **pytz** package. In previous releases, the names were validated against **pytz.common_timezones**, which is a subset of the currently used list. Note that the graphical and text mode interfaces still use the more restricted **pytz.common_timezones** list; you must use a Kickstart file to use additional time zone definitions.

H.3.16. user

The **user** Kickstart command is optional. It creates a new user on the system.

Syntax

```
user --name=username [OPTIONS]
```

Mandatory options

- **--name=** - Provides the name of the user. This option is required.

Optional options

- **--gecos=** - Provides the GECOS information for the user. This is a string of various system-specific fields separated by a comma. It is frequently used to specify the user's full name, office number, and so on. See the **passwd(5)** man page for more details.
- **--groups=** - In addition to the default group, a comma separated list of group names the user should belong to. The groups must exist before the user account is created. See the **group** command.
- **--homedir=** - The home directory for the user. If not provided, this defaults to **/home/username**.
- **--lock** - If this option is present, this account is locked by default. This means that the user will not be able to log in from the console. This option will also disable the **Create User** screens in both the graphical and text-based manual installation.
- **--password=** - The new user's password. If not provided, the account will be locked by default.
- **--iscrypted** - If this option is present, the password argument is assumed to already be encrypted. This option is mutually exclusive with **--plaintext**. To create an encrypted password, you can use python:

```
$ python -c 'import crypt,getpass;pw=getpass.getpass();print(crypt.crypt(pw) if (pw==getpass.getpass("Confirm: ")) else exit())'
```

This generates a sha512 crypt-compatible hash of your password using a random salt.

- **--plaintext** - If this option is present, the password argument is assumed to be in plain text. This option is mutually exclusive with **--iscrypted**
- **--shell=** - The user's login shell. If not provided, the system default is used.

- **--uid=** - The user’s UID (User ID). If not provided, this defaults to the next available non-system UID.
- **--gid=** - The GID (Group ID) to be used for the user’s group. If not provided, this defaults to the next available non-system group ID.

Notes

- Consider using the **--uid** and **--gid** options to set IDs of regular users and their default groups at range starting at **5000** instead of **1000**. That is because the range reserved for system users and groups, **0-999**, might increase in the future and thus overlap with IDs of regular users. For changing the minimum UID and GID limits after the installation, which ensures that your chosen UID and GID ranges are applied automatically on user creation, see the [Setting default permissions for new files using umask](#) section of the *Configuring basic system settings* document.
- Files and directories are created with various permissions, dictated by the application used to create the file or directory. For example, the **mkdir** command creates directories with all permissions enabled. However, applications are prevented from granting certain permissions to newly created files, as specified by the **user file-creation mask** setting. The **user file-creation mask** can be controlled with the **umask** command. The default setting of the **user file-creation mask** for new users is defined by the **UMASK** variable in the **/etc/login.defs** configuration file on the installed system. If unset, it defaults to **022**. This means that by default when an application creates a file, it is prevented from granting write permission to users other than the owner of the file. However, this can be overridden by other settings or scripts. More information can be found in the [Setting default permissions for new files using umask](#) section of the *Configuring basic system settings* document.

H.3.17. xconfig

The **xconfig** Kickstart command is optional. It configures the X Window System.

Syntax

```
xconfig [--startxonboot]
```

Options

- **--startxonboot** - Use a graphical login on the installed system.

Notes

- Because Red Hat Enterprise Linux 8 does not include the KDE Desktop Environment, do not use the **--defaultdesktop=** documented in upstream.

H.4. KICKSTART COMMANDS FOR NETWORK CONFIGURATION

The Kickstart commands in this list let you configure networking on the system.

H.4.1. network

The **network** Kickstart command is optional. It configures network information for the target system and activates network devices in the installation environment.

The device specified in the first **network** command is activated automatically. Activation of the device can be also explicitly required by the **--activate** option.

Syntax

```
network OPTIONS
```

Options

- **--activate** - activate this device in the installation environment.

If you use the **--activate** option on a device that has already been activated (for example, an interface you configured with boot options so that the system could retrieve the Kickstart file) the device is reactivated to use the details specified in the Kickstart file.

Use the **--nodefroute** option to prevent the device from using the default route.

- **--no-activate** - do not activate this device in the installation environment.

By default, Anaconda activates the first network device in the Kickstart file regardless of the **--activate** option. You can disable the default setting by using the **--no-activate** option.

- **--bootproto=** - One of **dhcp**, **bootp**, **ibft**, or **static**. The default option is **dhcp**; the **dhcp** and **bootp** options are treated the same. To disable **ipv4** configuration of the device, use **--noipv4** option.



NOTE

This option configures ipv4 configuration of the device. For ipv6 configuration use **--ipv6** and **--ipv6gateway** options.

The DHCP method uses a DHCP server system to obtain its networking configuration. The BOOTP method is similar, requiring a BOOTP server to supply the networking configuration. To direct a system to use DHCP:

```
network --bootproto=dhcp
```

To direct a machine to use BOOTP to obtain its networking configuration, use the following line in the Kickstart file:

```
network --bootproto=bootp
```

To direct a machine to use the configuration specified in iBFT, use:

```
network --bootproto=ibft
```

The **static** method requires that you specify at least the IP address and netmask in the Kickstart file. This information is static and is used during and after the installation.

All static networking configuration information must be specified on one line; you cannot wrap lines using a backslash (\) as you can on a command line.

```
network --bootproto=static --ip=10.0.2.15 --netmask=255.255.255.0 --gateway=10.0.2.254 --nameserver=10.0.2.1
```

You can also configure multiple nameservers at the same time. To do so, use the **--nameserver=** option once, and specify each of their IP addresses, separated by commas:

```
network --bootproto=static --ip=10.0.2.15 --netmask=255.255.255.0 --gateway=10.0.2.254 --
nameserver=192.168.2.1,192.168.3.1
```

- **--device=** - specifies the device to be configured (and eventually activated in Anaconda) with the **network** command.

If the **--device=** option is missing on the *first* use of the **network** command, the value of the **ksdevice=** Anaconda boot option is used, if available. Note that this is considered deprecated behavior; in most cases, you should always specify a **--device=** for every **network** command.

The behavior of any subsequent **network** command in the same Kickstart file is unspecified if its **--device=** option is missing. Make sure you specify this option for any **network** command beyond the first.

You can specify a device to be activated in any of the following ways:

- the device name of the interface, for example, **em1**
- the MAC address of the interface, for example, **01:23:45:67:89:ab**
- the keyword **link**, which specifies the first interface with its link in the **up** state
- the keyword **bootif**, which uses the MAC address that pxelinux set in the **BOOTIF** variable. Set **IPAPPEND 2** in your **pxelinux.cfg** file to have pxelinux set the **BOOTIF** variable.

For example:

```
network --bootproto=dhcp --device=em1
```

- **--ip=** - IP address of the device.
- **--ipv6=** - IPv6 address of the device, in the form of *address[/prefix length]* - for example, **3ffe:ffff:0:1::1/128**. If **prefix is omitted**, **'64** is used. You can also use **auto** for automatic configuration, or **dhcp** for DHCPv6-only configuration (no router advertisements).
- **--gateway=** - Default gateway as a single IPv4 address.
- **--ipv6gateway=** - Default gateway as a single IPv6 address.
- **--nodefroute** - Prevents the interface being set as the default route. Use this option when you activate additional devices with the **--activate=** option, for example, a NIC on a separate subnet for an iSCSI target.
- **--nameserver=** - DNS name server, as an IP address. To specify more than one name server, use this option once, and separate each IP address with a comma.
- **--netmask=** - Network mask for the installed system.
- **--hostname=** - The host name for the installed system. The host name can either be a fully-qualified domain name (FQDN) in the format **host_name.domainname**, or a short host name with no domain. Many networks have a Dynamic Host Configuration Protocol (DHCP) service which automatically supplies connected systems with a domain name; to allow DHCP to assign the domain name, only specify a short host name.



IMPORTANT

If your network does *not* provide a DHCP service, always use the FQDN as the system's host name.

- **--ethtool=** - Specifies additional low-level settings for the network device which will be passed to the ethtool program.
- **--onboot=** - Whether or not to enable the device at boot time.
- **--dhcpclass=** - The DHCP class.
- **--mtu=** - The MTU of the device.
- **--noipv4** - Disable IPv4 on this device.
- **--noipv6** - Disable IPv6 on this device.
- **--bondslaves=** - When this option is used, the bond device specified by the **--device=** option is created using slaves defined in the **--bondslaves=** option. For example:

```
network --device=bond0 --bondslaves=em1,em2
```

The above command creates a bond device named **bond0** using the **em1** and **em2** interfaces as its slaves.

- **--bondopts=** - a list of optional parameters for a bonded interface, which is specified using the **--bondslaves=** and **--device=** options. Options in this list must be separated by commas (",") or semicolons (";"). If an option itself contains a comma, use a semicolon to separate the options. For example:

```
network --bondopts=mode=active-backup,balance-rr;primary=eth1
```



IMPORTANT

The **--bondopts=mode=** parameter only supports full mode names such as **balance-rr** or **broadcast**, not their numerical representations such as **0** or **3**.

- **--vlanid=** - Specifies virtual LAN (VLAN) ID number (802.1q tag) for the device created using the device specified in **--device=** as a parent. For example, **network --device=em1 --vlanid=171** creates a virtual LAN device **em1.171**.
- **--interfacename=** - Specify a custom interface name for a virtual LAN device. This option should be used when the default name generated by the **--vlanid=** option is not desirable. This option must be used along with **--vlanid=**. For example:

```
network --device=em1 --vlanid=171 --interfacename=vlan171
```

The above command creates a virtual LAN interface named **vlan171** on the **em1** device with an ID of **171**.

The interface name can be arbitrary (for example, **my-vlan**), but in specific cases, the following conventions must be followed:

- If the name contains a dot (.), it must take the form of **NAME.ID**. The *NAME* is arbitrary, but the *ID* must be the VLAN ID. For example: **em1.171** or **my-vlan.171**.
- Names starting with **vlan** must take the form of **vlan***ID* - for example, **vlan171**.
- **--teamslaves=** - Team device specified by the **--device=** option will be created using slaves specified in this option. Slaves are separated by commas. A slave can be followed by its configuration, which is a single-quoted JSON string with double quotes escaped by the \ character. For example:

```
network --teamslaves="p3p1'{"prio": -10, "sticky": true}',p3p2'{"prio": 100}"
```

See also the **--teamconfig=** option.

- **--teamconfig=** - Double-quoted team device configuration which is a JSON string with double quotes escaped by the \ character. The device name is specified by **--device=** option and its slaves and their configuration by **--teamslaves=** option. For example:

```
network --device team0 --activate --bootproto static --ip=10.34.102.222 --
netmask=255.255.255.0 --gateway=10.34.102.254 --nameserver=10.34.39.2 --
teamslaves="p3p1'{"prio": -10, "sticky": true}',p3p2'{"prio": 100}" --teamconfig="{"runner": {"name": "activebackup"}}
```

- **--bridgeslaves=** - When this option is used, the network bridge with device name specified using the **--device=** option will be created and devices defined in the **--bridgeslaves=** option will be added to the bridge. For example:

```
network --device=bridge0 --bridgeslaves=em1
```

- **--bridgeopts=** - An optional comma-separated list of parameters for the bridged interface. Available values are **stp**, **priority**, **forward-delay**, **hello-time**, **max-age**, and **ageing-time**. For information about these parameters, see the *bridge setting* table in the **nm-settings(5)** man page or at <https://developer.gnome.org/NetworkManager/0.9/ref-settings.html>. Also see the [Configuring and managing networking](#) document for general information about network bridging.
- **--bindto=mac** - Bind the device configuration (**ifcfg**) file on the installed system to the device MAC address (**HWADDR**) instead of the default binding to the interface name (**DEVICE**). Note that this option is independent of the **--device=** option - **--bindto=mac** will be applied even if the same **network** command also specifies a device name, **link**, or **bootif**.

Notes

- The **ethN** device names such as **eth0** are no longer available in Red Hat Enterprise Linux 8 due to changes in the naming scheme. For more information about the device naming scheme, see the upstream document [Predictable Network Interface Names](#) .
- If you used a Kickstart option or a boot option to specify an installation repository on a network, but no network is available at the start of the installation, the installation program displays the **Network Configuration** window to set up a network connection prior to displaying the **Installation Summary** window. For more details, see the [Configuring network and host name options](#) section of the *Performing a standard RHEL installation* document.

H.4.2. realm

The **realm** Kickstart command is optional. Use it to join an Active Directory or IPA domain. For more information about this command, see the **join** section of the **realm(8)** man page.

Syntax

```
realm join [OPTIONS] domain
```

Mandatory options

- **domain** - The domain to join.

Options

- **--computer-ou=OU=** - Provide the distinguished name of an organizational unit in order to create the computer account. The exact format of the distinguished name depends on the client software and membership software. The root DSE portion of the distinguished name can usually be left out.
- **--no-password** - Join automatically without a password.
- **--one-time-password=** - Join using a one-time password. This is not possible with all types of realm.
- **--client-software=** - Only join realms which can run this client software. Valid values include **sssd** and **winbind**. Not all realms support all values. By default, the client software is chosen automatically.
- **--server-software=** - Only join realms which can run this server software. Possible values include **active-directory** or **freeipa**.
- **--membership-software=** - Use this software when joining the realm. Valid values include **samba** and **adcli**. Not all realms support all values. By default, the membership software is chosen automatically.

H.5. KICKSTART COMMANDS FOR HANDLING STORAGE

The Kickstart commands in this section configure aspects of storage such as devices, disks, partitions, LVM, and filesystems.

H.5.1. device (deprecated)

The **device** Kickstart command is optional. Use it to load additional kernel modules.

On most PCI systems, the installation program automatically detects Ethernet and SCSI cards. However, on older systems and some PCI systems, Kickstart requires a hint to find the proper devices. The **device** command, which tells the installation program to install extra modules, uses the following format:

Syntax

```
device moduleName --opts=options
```

Options

- *moduleName* - Replace with the name of the kernel module which should be installed.
- **--opts=** - Options to pass to the kernel module. For example:

device --opts="aic152x=0x340 io=11"

H.5.2. autopart

The **autopart** Kickstart command is optional. It automatically creates partitions.

The automatically created partitions are: a root (/) partition (1 GB or larger), a **swap** partition, and an appropriate **/boot** partition for the architecture. On large enough drives (50 GB and larger), this also creates a **/home** partition.

Syntax

autopart *OPTIONS*

Options

- **--type=** - Selects one of the predefined automatic partitioning schemes you want to use. Accepts the following values:
 - **lvm**: The LVM partitioning scheme.
 - **plain**: Regular partitions with no LVM.
 - **thinp**: The LVM Thin Provisioning partitioning scheme.
- **--fstype=** - Selects one of the available file system types. The available values are **ext2**, **ext3**, **ext4**, **xfs**, and **vfat**. The default file system is **xfs**.
- **--nohome** - Disables automatic creation of the **/home** partition.
- **--nolvm** - Do not use LVM for automatic partitioning. This option is equal to **--type=plain**.
- **--noboot** - Do not create a **/boot** partition.
- **--noswap** - Do not create a swap partition.
- **--encrypted** - Encrypts all partitions with Linux Unified Key Setup (LUKS). This is equivalent to checking the **Encrypt partitions** check box on the initial partitioning screen during a manual graphical installation.



NOTE

When encrypting one or more partitions, Anaconda attempts to gather 256 bits of entropy to ensure the partitions are encrypted securely. Gathering entropy can take some time - the process will stop after a maximum of 10 minutes, regardless of whether sufficient entropy has been gathered.

The process can be sped up by interacting with the installation system (typing on the keyboard or moving the mouse). If you are installing in a virtual machine, you can also attach a **virtio-rng** device (a virtual random number generator) to the guest.

- **--luks-version=LUKS_VERSION** - Specifies which version of LUKS format should be used to encrypt the filesystem. This option is only meaningful if **--encrypted** is specified.
- **--passphrase=** - Provides a default system-wide passphrase for all encrypted devices.
- **--escrowcert=URL_of_X.509_certificate** - Stores data encryption keys of all encrypted volumes as files in **/root**, encrypted using the X.509 certificate from the URL specified with *URL_of_X.509_certificate*. The keys are stored as a separate file for each encrypted volume. This option is only meaningful if **--encrypted** is specified.
- **--backuptoolsphrase** - Adds a randomly-generated passphrase to each encrypted volume. Store these passphrases in separate files in **/root**, encrypted using the X.509 certificate specified with **--escrowcert**. This option is only meaningful if **--escrowcert** is specified.
- **--cipher=** - Specifies the type of encryption to use if the Anaconda default **aes-xts-plain64** is not satisfactory. You must use this option together with the **--encrypted** option; by itself it has no effect. Available types of encryption are listed in the [Security hardening](#) document, but Red Hat strongly recommends using either **aes-xts-plain64** or **aes-cbc-essiv:sha256**.
- **--pbkdf=PBKDF** - Sets Password-Based Key Derivation Function (PBKDF) algorithm for LUKS keyslot. See also the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified.
- **--pbkdf-memory=PBKDF_MEMORY** - Sets the memory cost for PBKDF. See also the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified.
- **--pbkdf-time=PBKDF_TIME** - Sets the number of milliseconds to spend with PBKDF passphrase processing. See also **--iter-time** in the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified, and is mutually exclusive with **--pbkdf-iterations**.
- **--pbkdf-iterations=PBKDF_ITERATIONS** - Sets the number of iterations directly and avoids PBKDF benchmark. See also **--pbkdf-force-iterations** in the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified, and is mutually exclusive with **--pbkdf-time**.

Notes

- The **autopart** option cannot be used together with the **part/partition**, **raid**, **logvol**, or **volgroup** options in the same Kickstart file.
- The **autopart** command is not mandatory, but you must include it if there are no **part** or **mount** commands in your Kickstart script.
- It is recommended to use the **autopart --nohome** Kickstart option when installing on a single FBA DASD of the CMS type. This ensures that the installation program does not create a separate **/home** partition. The installation then proceeds successfully.
- If you lose the LUKS passphrase, any encrypted partitions and their data is completely inaccessible. There is no way to recover a lost passphrase. However, you can save encryption passphrases with the **--escrowcert** and create backup encryption passphrases with the **--backuptoolsphrase** options.

H.5.3. bootloader (required)

The **bootloader** Kickstart command is required. It specifies how the boot loader should be installed.

Syntax

```
bootloader [OPTIONS]
```

Options

- **--append=** - Specifies additional kernel parameters. To specify multiple parameters, separate them with spaces. For example:

```
bootloader --location=mbr --append="hdd=ide-scsi ide=nodma"
```

The **rhgb** and **quiet** parameters are automatically added when the **plymouth** package is installed, even if you do not specify them here or do not use the **--append=** command at all. To disable this behavior, explicitly disallow installation of **plymouth**:

```
%packages
-plymouth
%end
```

This option is useful for disabling mechanisms which were implemented to mitigate the Meltdown and Spectre speculative execution vulnerabilities found in most modern processors (CVE-2017-5754, CVE-2017-5753, and CVE-2017-5715). In some cases, these mechanisms may be unnecessary, and keeping them enabled causes decreased performance with no improvement in security. To disable these mechanisms, add the options to do so into your Kickstart file - for example, **bootloader --append="nopti noibrs noibpb"** on AMD64/Intel 64 systems.



WARNING

Ensure your system is not at risk of attack before disabling any of the vulnerability mitigation mechanisms. See the [Red Hat vulnerability response article](#) for information about the Meltdown and Spectre vulnerabilities.

- **--boot-drive=** - Specifies which drive the boot loader should be written to, and therefore which drive the computer will boot from. If you use a multipath device as the boot drive, specify only one member of the device.



IMPORTANT

The **--boot-drive=** option is currently being ignored in Red Hat Enterprise Linux installations on IBM Z systems using the **zipl** boot loader. When **zipl** is installed, it determines the boot drive on its own.

- **--leavebootorder** - The installation program will add Red Hat Enterprise Linux 8 to the top of the list of installed systems in the boot loader, and preserve all existing entries as well as their order.

- **--driveorder=** - Specifies which drive is first in the BIOS boot order. For example:

```
bootloader --driveorder=sda,hda
```

- **--location=** - Specifies where the boot record is written. Valid values are the following:

- **mbr** - The default option. Depends on whether the drive uses the Master Boot Record (MBR) or GUID Partition Table (GPT) scheme:
On a GPT-formatted disk, this option installs stage 1.5 of the boot loader into the BIOS boot partition.

On an MBR-formatted disk, stage 1.5 is installed into the empty space between the MBR and the first partition.

- **partition** - Install the boot loader on the first sector of the partition containing the kernel.
- **none** - Do not install the boot loader.

In most cases, this option does not need to be specified.

- **--nombr** - Do not install the boot loader to the MBR.

- **--password=** - If using GRUB2, sets the boot loader password to the one specified with this option. This should be used to restrict access to the GRUB2 shell, where arbitrary kernel options can be passed.

If a password is specified, GRUB2 also asks for a user name. The user name is always **root**.

- **--iscrypted** - Normally, when you specify a boot loader password using the **--password=** option, it is stored in the Kickstart file in plain text. If you want to encrypt the password, use this option and an encrypted password.

To generate an encrypted password, use the **grub2-mkpasswd-pbkdf2** command, enter the password you want to use, and copy the command's output (the hash starting with **grub.pbkdf2**) into the Kickstart file. An example **bootloader** Kickstart entry with an encrypted password looks similar to the following:

```
bootloader --iscrypted --
password=grub.pbkdf2.sha512.10000.5520C6C9832F3AC3D149AC0B24BE69E2D4FB0DBE
EDBD29CA1D30A044DE2645C4C7A291E585D4DC43F8A4D82479F8B95CA4BA4381F8550
510B75E8E0BB2938990.C688B6F0EF935701FF9BD1A8EC7FE5BD2333799C98F28420C5
CC8F1A2A233DE22C83705BB614EA17F3FDFDF4AC2161CEA3384E56EB38A2E39102F53
34C47405E
```

- **--timeout=** - Specifies the amount of time the boot loader waits before booting the default option (in seconds).
- **--default=** - Sets the default boot image in the boot loader configuration.
- **--extlinux** - Use the extlinux boot loader instead of GRUB2. This option only works on systems supported by extlinux.
- **--disabled** - This option is a stronger version of **--location=none**. While **--location=none** simply disables boot loader installation, **--disabled** disables boot loader installation and also disables installation of the package containing the boot loader, thus saving space.

Notes

- Red Hat recommends setting up a boot loader password on every system. An unprotected boot loader can allow a potential attacker to modify the system's boot options and gain unauthorized access to the system.
- In some cases, a special partition is required to install the boot loader on AMD64, Intel 64, and 64-bit ARM systems. The type and size of this partition depends on whether the disk you are installing the boot loader to uses the Master Boot Record (MBR) or a GUID Partition Table (GPT) schema. For more information, see the [Configuring boot loader](#) section of the *Performing a standard RHEL installation* document.
- Device names in the **sdX** (or **/dev/sdX**) format are not guaranteed to be consistent across reboots, which can complicate usage of some Kickstart commands. When a command calls for a device node name, you can instead use any item from **/dev/disk**. For example, instead of:

```
part / --fstype=xfs --onpart=sda1
```

You can use an entry similar to one of the following:

```
part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
```

```
part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1
```

This way the command will always target the same storage device. This is especially useful in large storage environments. See the chapter [Overview of persistent naming attributes](#) in the *Managing storage devices* document for more in-depth information about different ways to consistently refer to storage devices.

- The **--upgrade** option is deprecated in Red Hat Enterprise Linux 8.

H.5.4. **zipl**

The **zipl** Kickstart command is optional. It specifies the ZIPL configuration for IBM Z.

Options

- secure-boot** - Enables secure boot if it is supported by the installing system.



NOTE

When installed on a system that is later than IBM z14, the installed system cannot be booted from an IBM z14 or earlier model.

- force-secure-boot** - Enables secure boot unconditionally.



NOTE

Installation is not supported on IBM z14 and earlier models.

- no-secure-boot** - Disables secure boot.



NOTE

Secure Boot is not supported on IBM z14 and earlier models. Use **--no-secure-boot** if you intend to boot the installed system on IBM z14 and earlier models.

H.5.5. clearpart

The **clearpart** Kickstart command is optional. It removes partitions from the system, prior to creation of new partitions. By default, no partitions are removed.

Syntax

```
clearpart OPTIONS
```

Options

- **--all** - Erases all partitions from the system.

This option will erase all disks which can be reached by the installation program, including any attached network storage. Use this option with caution.

You can prevent **clearpart** from wiping storage you want to preserve by using the **--drives=** option and specifying only the drives you want to clear, by attaching network storage later (for example, in the **%post** section of the Kickstart file), or by blacklisting the kernel modules used to access network storage.

- **--drives=** - Specifies which drives to clear partitions from. For example, the following clears all the partitions on the first two drives on the primary IDE controller:

```
clearpart --drives=hda,hdb --all
```

To clear a multipath device, use the format **disk/by-id/scsi-*WWID***, where *WWID* is the world-wide identifier for the device. For example, to clear a disk with *WWID*

58095BEC5510947BE8C0360F604351918, use:

```
clearpart --drives=disk/by-id/scsi-58095BEC5510947BE8C0360F604351918
```

This format is preferable for all multipath devices, but if errors arise, multipath devices that do not use logical volume management (LVM) can also be cleared using the format **disk/by-id/dm-*uuid-mpath-*WWID****, where *WWID* is the world-wide identifier for the device. For example, to clear a disk with *WWID* **2416CD96995134CA5D787F00A5AA11017**, use:

```
clearpart --drives=disk/by-id/dm-uuid-mpath-2416CD96995134CA5D787F00A5AA11017
```

Never specify multipath devices by device names like **mpatha**. Device names such as this are not specific to a particular disk. The disk named **/dev/mpatha** during installation might not be the one that you expect it to be. Therefore, the **clearpart** command could target the wrong disk.

- **--initlabel** - Initializes a disk (or disks) by creating a default disk label for all disks in their respective architecture that have been designated for formatting (for example, msdos for x86). Because **--initlabel** can see all disks, it is important to ensure only those drives that are to be formatted are connected.

```
clearpart --initlabel --drives=names_of_disks
```

For example:

```
clearpart --initlabel --drives=dasda,dasdb,dasdc
```

- **--list=** - Specifies which partitions to clear. This option overrides the **--all** and **--linux** options if used. Can be used across different drives. For example:

```
clearpart --list=sda2,sda3,sdb1
```

- **--disklabel=LABEL** - Set the default disklabel to use. Only disklabels supported for the platform will be accepted. For example, on the 64-bit Intel and AMD architectures, the **msdos** and **gpt** disklabels are accepted, but **dasd** is not accepted.
- **--linux** - Erases all Linux partitions.
- **--none** (default) - Do not remove any partitions.
- **--cdl** - Reformat any LDL DASDs to CDL format.

Notes

- Device names in the **sdX** (or **/dev/sdX**) format are not guaranteed to be consistent across reboots, which can complicate usage of some Kickstart commands. When a command calls for a device node name, you can instead use any item from **/dev/disk**. For example, instead of:

```
part / --fstype=xfs --onpart=sda1
```

You could use an entry similar to one of the following:

```
part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
```

```
part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1
```

This way the command will always target the same storage device. This is especially useful in large storage environments. See the chapter [Overview of persistent naming attributes](#) in the *Managing storage devices* document for more in-depth information about different ways to consistently refer to storage devices.

- If the **clearpart** command is used, then the **part --onpart** command cannot be used on a logical partition.

H.5.6. fcoe

The **fcoe** Kickstart command is optional. It specifies which FCoE devices should be activated automatically in addition to those discovered by Enhanced Disk Drive Services (EDD).

Syntax

```
fcoe --nic=name [OPTIONS]
```

Options

- **--nic=** (required) - The name of the device to be activated.

- **--dcb=** - Establish Data Center Bridging (DCB) settings.
- **--autovlan** - Discover VLANs automatically. This option is enabled by default.

H.5.7. ignoredisk

The **ignoredisk** Kickstart command is optional. It causes the installation program to ignore the specified disks.

This is useful if you use automatic partitioning and want to be sure that some disks are ignored. For example, without **ignoredisk**, attempting to deploy on a SAN-cluster the Kickstart would fail, as the installation program detects passive paths to the SAN that return no partition table.

Syntax

```
ignoredisk --drives=drive1,drive2,... | --only-use=drive
```

Options

- **--drives=driveN,...** - Replace *driveN* with one of **sda**, **sdb**,..., **hda**,... and so on.
- **--only-use=driveN,...** - Specifies a list of disks for the installation program to use. All other disks are ignored. For example, to use disk **sda** during installation and ignore all other disks:

```
ignoredisk --only-use=sda
```

To include a multipath device that does not use LVM:

```
ignoredisk --only-use=disk/by-id/dm-uuid-mpath-2416CD96995134CA5D787F00A5AA11017
```

To include a multipath device that uses LVM:

```
ignoredisk --only-use=disk/by-id/scsi-58095BEC5510947BE8C0360F604351918
```

You must specify one of the **--drives** and **--only-use**.

Notes

- The **--interactive** option is deprecated in Red Hat Enterprise Linux 8. This option allowed users to manually navigate the advanced storage screen.
- To ignore a multipath device that does not use logical volume management (LVM), use the format **disk/by-id/dm-uuid-mpath-*WWID***, where *WWID* is the world-wide identifier for the device. For example, to ignore a disk with **WWID 2416CD96995134CA5D787F00A5AA11017**, use:


```
ignoredisk --drives=disk/by-id/dm-uuid-mpath-2416CD96995134CA5D787F00A5AA11017
```
- Multipath devices that use LVM are not assembled until after Anaconda has parsed the Kickstart file. Therefore, you cannot specify these devices in the format **dm-uuid-mpath**. Instead, to ignore a multipath device that uses LVM, use the format **disk/by-id/scsi-*WWID***, where *WWID* is the world-wide identifier for the device. For example, to ignore a disk with *WWID 58095BEC5510947BE8C0360F604351918*, use:

```
ignoredisk --drives=disk/by-id/scsi-58095BEC5510947BE8C0360F604351918
```

- Never specify multipath devices by device names like **mpatha**. Device names such as this are not specific to a particular disk. The disk named **/dev/mpatha** during installation might not be the one that you expect it to be. Therefore, the **clearpart** command could target the wrong disk.
- Device names in the **sdX** (or **/dev/sdX**) format are not guaranteed to be consistent across reboots, which can complicate usage of some Kickstart commands. When a command calls for a device node name, you can instead use any item from **/dev/disk**. For example, instead of:

```
part / --fstype=xfs --onpart=sda1
```

You can use an entry similar to one of the following:

```
part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
```

```
part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1
```

This way the command will always target the same storage device. This is especially useful in large storage environments. See the chapter [Overview of persistent naming attributes](#) in the *Managing storage devices* document for more in-depth information about different ways to consistently refer to storage devices.

H.5.8. iscsi

The **iscsi** Kickstart command is optional. It specifies additional iSCSI storage to be attached during installation.

Syntax

```
iscsi --ipaddr=address [OPTIONS]
```

Mandatory options

- ipaddr=** (required) - the IP address of the target to connect to.

Optional options

- port=** (required) - the port number. If not present, **--port=3260** is used automatically by default.
- target=** - the target IQN (iSCSI Qualified Name).
- iface=** - bind the connection to a specific network interface instead of using the default one determined by the network layer. Once used, it must be specified in all instances of the **iscsi** command in the entire Kickstart file.
- user=** - the user name required to authenticate with the target
- password=** - the password that corresponds with the user name specified for the target
- reverse-user=** - the user name required to authenticate with the initiator from a target that uses reverse CHAP authentication

- **--reverse-password=** - the password that corresponds with the user name specified for the initiator

Notes

- If you use the **iscsi** command, you must also assign a name to the iSCSI node, using the **iscsiname** command. The **iscsiname** command must appear before the **iscsi** command in the Kickstart file.
- Wherever possible, configure iSCSI storage in the system BIOS or firmware (iBFT for Intel systems) rather than use the **iscsi** command. Anaconda automatically detects and uses disks configured in BIOS or firmware and no special configuration is necessary in the Kickstart file.
- If you must use the **iscsi** command, ensure that networking is activated at the beginning of the installation, and that the **iscsi** command appears in the Kickstart file *before* you refer to iSCSI disks with commands such as **clearpart** or **ignoredisk**.

H.5.9. iscsiname

The **iscsiname** Kickstart command is optional. It assigns a name to an iSCSI node specified by the **iscsi** command.

Syntax

```
iscsiname iqname
```

Options

- ***iqname*** - Name to assign to the iSCSI node.

Notes

- If you use the **iscsi** command in your Kickstart file, you must specify **iscsiname** *earlier* in the Kickstart file.

H.5.10. logvol

The **logvol** Kickstart command is optional. It creates a logical volume for Logical Volume Management (LVM).

Syntax

```
logvol mntpoint --vgname=name --name=name [OPTIONS]
```

Mandatory options

- ***mntpoint*** - The mount point where the partition is mounted. Must be of one of the following forms:
 - **/path**
For example, **/** or **/home**
 - **swap**
The partition is used as swap space.

To determine the size of the swap partition automatically, use the **--recommended** option:

```
swap --recommended
```

To determine the size of the swap partition automatically and also allow extra space for your system to hibernate, use the **--hibernation** option:

```
swap --hibernation
```

The size assigned will be equivalent to the swap space assigned by **--recommended** plus the amount of RAM on your system.

For the swap sizes assigned by these commands, see [Section C.4, “Recommended partitioning scheme”](#) for AMD64, Intel 64, and 64-bit ARM systems.

- **--vgname=name** - name of the volume group.
- **--name=name** - name of the logical volume.

Optional options

- **--noformat** - Use an existing logical volume and do not format it.
- **--useexisting** - Use an existing logical volume and reformat it.
- **--fstype=** - Sets the file system type for the logical volume. Valid values are **xfs**, **ext2**, **ext3**, **ext4**, **swap**, and **vfat**.
- **--fsoptions=** - Specifies a free form string of options to be used when mounting the filesystem. This string will be copied into the **/etc/fstab** file of the installed system and should be enclosed in quotes.
- **--mkfsoptions=** - Specifies additional parameters to be passed to the program that makes a filesystem on this partition. No processing is done on the list of arguments, so they must be supplied in a format that can be passed directly to the **mkfs** program. This means multiple options should be comma-separated or surrounded by double quotes, depending on the filesystem.
- **--fsprofile=** - Specifies a usage type to be passed to the program that makes a filesystem on this partition. A usage type defines a variety of tuning parameters to be used when making a filesystem. For this option to work, the filesystem must support the concept of usage types and there must be a configuration file that lists valid types. For **ext2**, **ext3**, and **ext4**, this configuration file is **/etc/mke2fs.conf**.
- **--label=** - Sets a label for the logical volume.
- **--grow** - Extends the logical volume to occupy the available space (if any), or up to the maximum size specified, if any. The option must be used only if you have pre-allocated a minimum storage space in the disk image, and would want the volume to grow and occupy the available space. In a physical environment, this is an one-time-action. However, in a virtual environment, the volume size increases as and when the virtual machine writes any data to the virtual disk.
- **--size=** - The size of the logical volume in MiB. This option cannot be used together with the **--percent=** option.

- **--percent=** - The size of the logical volume, as a percentage of the free space in the volume group after any statically-sized logical volumes are taken into account. This option cannot be used together with the **--size=** option.



IMPORTANT

When creating a new logical volume, you must either specify its size statically using the **--size=** option, or as a percentage of remaining free space using the **--percent=** option. You cannot use both of these options on the same logical volume.

- **--maxsize=** - The maximum size in MiB when the logical volume is set to grow. Specify an integer value here such as **500** (do not include the unit).
- **--recommended** - Use this option when creating a logical volume to determine the size of this volume automatically, based on your system's hardware. For details about the recommended scheme, see [Section C.4, “Recommended partitioning scheme”](#) for AMD64, Intel 64, and 64-bit ARM systems.
- **--resize** - Resize a logical volume. If you use this option, you must also specify **--useexisting** and **--size**.
- **--encrypted** - Specifies that this logical volume should be encrypted with Linux Unified Key Setup (LUKS), using the passphrase provided in the **--passphrase=** option. If you do not specify a passphrase, the installation program uses the default, system-wide passphrase set with the **autopart --passphrase** command, or stops the installation and prompts you to provide a passphrase if no default is set.



NOTE

When encrypting one or more partitions, Anaconda attempts to gather 256 bits of entropy to ensure the partitions are encrypted securely. Gathering entropy can take some time - the process will stop after a maximum of 10 minutes, regardless of whether sufficient entropy has been gathered.

The process can be sped up by interacting with the installation system (typing on the keyboard or moving the mouse). If you are installing in a virtual machine, you can also attach a **virtio-rng** device (a virtual random number generator) to the guest.

- **--passphrase=** - Specifies the passphrase to use when encrypting this logical volume. You must use this option together with the **--encrypted** option; it has no effect by itself.
- **--cipher=** - Specifies the type of encryption to use if the Anaconda default **aes-xts-plain64** is not satisfactory. You must use this option together with the **--encrypted** option; by itself it has no effect. Available types of encryption are listed in the [Security hardening](#) document, but Red Hat strongly recommends using either **aes-xts-plain64** or **aes-cbc-essiv:sha256**.
- **--escrowcert=URL_of_X.509_certificate** - Store data encryption keys of all encrypted volumes as files in **/root**, encrypted using the X.509 certificate from the URL specified with **URL_of_X.509_certificate**. The keys are stored as a separate file for each encrypted volume. This option is only meaningful if **--encrypted** is specified.
- **--luks-version=LUKS_VERSION** - Specifies which version of LUKS format should be used to encrypt the filesystem. This option is only meaningful if **--encrypted** is specified.

- **--backuppassphrase** - Add a randomly-generated passphrase to each encrypted volume. Store these passphrases in separate files in **/root**, encrypted using the X.509 certificate specified with **--escrowcert**. This option is only meaningful if **--escrowcert** is specified.
- **--pbkdf=PBKDF** - Sets Password-Based Key Derivation Function (PBKDF) algorithm for LUKS keyslot. See also the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified.
- **--pbkdf-memory=PBKDF_MEMORY** - Sets the memory cost for PBKDF. See also the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified.
- **--pbkdf-time=PBKDF_TIME** - Sets the number of milliseconds to spend with PBKDF passphrase processing. See also **--iter-time** in the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified, and is mutually exclusive with **--pbkdf-iterations**.
- **--pbkdf-iterations=PBKDF_ITERATIONS** - Sets the number of iterations directly and avoids PBKDF benchmark. See also **--pbkdf-force-iterations** in the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified, and is mutually exclusive with **--pbkdf-time**.
- **--thinpool** - Creates a thin pool logical volume. (Use a mount point of **none**)
- **--metadatasize=size** - Specify the metadata area size (in MiB) for a new thin pool device.
- **--chunksize=size** - Specify the chunk size (in KiB) for a new thin pool device.
- **--thin** - Create a thin logical volume. (Requires use of **--poolname**)
- **--poolname=name** - Specify the name of the thin pool in which to create a thin logical volume. Requires the **--thin** option.
- **--profile=name** - Specify the configuration profile name to use with thin logical volumes. If used, the name will also be included in the metadata for the given logical volume. By default, the available profiles are **default** and **thin-performance** and are defined in the **/etc/lvm/profile/** directory. See the **lvm(8)** man page for additional information.
- **--cachepvs=** - A comma-separated list of physical volumes which should be used as a cache for this volume.
- **--cachemode=** - Specify which mode should be used to cache this logical volume - either **writeback** or **writethrough**.



NOTE

For more information about cached logical volumes and their modes, see the **lvmcache(7)** man page.

- **--cachesize=** - Size of cache attached to the logical volume, specified in MiB. This option requires the **--cachepvs=** option.

Notes

- Do not use the dash (-) character in logical volume and volume group names when installing Red Hat Enterprise Linux using Kickstart. If this character is used, the installation finishes normally, but the **/dev/mapper/** directory will list these volumes and volume groups with every dash

doubled. For example, a volume group named **volgrp-01** containing a logical volume named **logvol-01** will be listed as **/dev/mapper/volgrp-01-logvol-01**.

This limitation only applies to newly created logical volume and volume group names. If you are reusing existing ones using the **--noformat** option, their names will not be changed.

- If you lose the LUKS passphrase, any encrypted partitions and their data is completely inaccessible. There is no way to recover a lost passphrase. However, you can save encryption passphrases with the **--escrowcert** and create backup encryption passphrases with the **--backuptoolsphrase** options.

Examples

- Create the partition first, create the logical volume group, and then create the logical volume:

```
part pv.01 --size 3000
volgroup myvg pv.01
logvol / --vgname=myvg --size=2000 --name=rootvol
```

- Create the partition first, create the logical volume group, and then create the logical volume to occupy 90% of the remaining space in the volume group:

```
part pv.01 --size 1 --grow
volgroup myvg pv.01
logvol / --vgname=myvg --name=rootvol --percent=90
```

Additional resources

- For more information regarding LVM, see the [Configuring and managing logical volumes](#) document.

H.5.11. mount

The **mount** Kickstart command is optional. It assigns a mount point to an existing block device, and optionally reformats it to a given format.

Syntax

```
mount [OPTIONS] device mountpoint
```

Mandatory options:

- **device** - The block device to mount.
- **mountpoint** - Where to mount the **device**. It must be a valid mount point, such as **/** or **/usr**, or **none** if the device is unmountable (for example **swap**).

Optional options:

- **--reformat=** - Specifies a new format (such as **ext4**) to which the device should be reformatted.
- **--mkfsoptions=** - Specifies additional options to be passed to the command which creates the new file system specified in **--reformat=**. The list of options provided here is not processed, so they must be specified in a format that can be passed directly to the **mkfs** program. The list of

options should be either comma-separated or surrounded by double quotes, depending on the file system. See the **mkfs** man page for the file system you want to create (for example **mkfs.ext4(8)** or **mkfs.xfs(8)**) for specific details.

- **--mountoptions=** - Specifies a free form string that contains options to be used when mounting the file system. The string will be copied to the **/etc/fstab** file on the installed system and should be enclosed in double quotes. See the **mount(8)** man page for a full list of mount options, and **fstab(5)** for basics.

Notes

- Unlike most other storage configuration commands in Kickstart, **mount** does not require you to describe the entire storage configuration in the Kickstart file. You only need to ensure that the described block device exists on the system. However, if you want to *create* the storage stack with all the devices mounted, you must use other commands such as **part** to do so.
- You can not use **mount** together with other storage-related commands such as **part**, **logvol**, or **autopart** in the same Kickstart file.

H.5.12. nvdimm

The **nvdimm** Kickstart command is optional. It performs an action on Non-Volatile Dual In-line Memory Module (NVDIMM) devices.

Syntax

```
nvdimm action [OPTIONS]
```

Actions

- **reconfigure** - Reconfigure a specific NVDIMM device into a given mode. Additionally, the specified device is implicitly marked as to be used, so a subsequent **nvdimm use** command for the same device is redundant. This action uses the following format:

```
nvdimm reconfigure [--namespace=NAMESPACE] [--mode=MODE] [--sector-size=SECTOR_SIZE]
```

- **--namespace=** - The device specification by namespace. For example:

```
nvdimm reconfigure --namespace=namespace0.0 --mode=sector --sector-size=512
```

- **--mode=** - The mode specification. Currently, only the value **sector** is available.

- **--sector-size=** - Size of a sector for sector mode. For example:

```
nvdimm reconfigure --namespace=namespace0.0 --mode=sector --sector-size=512
```

The supported sector sizes are 512 and 4096 bytes.

- **use** - Specify a NVDIMM device as a target for installation. The device must be already configured to the sector mode by the **nvdimm reconfigure** command. This action uses the following format:

```
nvdimm use [--namespace=NAMESPACE] [--block-devs=DEVICES]
```

- **--namespace=** - Specifies the device by namespace. For example:

```
nvdimm use --namespace=namespace0.0
```

- **--blockdevs=** - Specifies a comma-separated list of block devices corresponding to the NVDIMM devices to be used. The asterisk * wildcard is supported. For example:

```
nvdimm use --blockdevs=pmem0s,pmem1s
nvdimm use --blockdevs=pmem*
```

Notes

- By default, all NVDIMM devices are ignored by the installation program. You must use the **nvdimm** command to enable installation on these devices.

H.5.13. part or partition

The **part** or **partition** Kickstart command is required. It creates a partition on the system.

Syntax

```
part|partition mntpoint --name=name --device=device --rule=rule [OPTIONS]
```

Options

- *mntpoint* - Where the partition is mounted. The value must be of one of the following forms:
 - **/path**
For example, **/**, **/usr**, **/home**
 - **swap**
The partition is used as swap space.

To determine the size of the swap partition automatically, use the **--recommended** option:

```
swap --recommended
```

The size assigned will be effective but not precisely calibrated for your system.

To determine the size of the swap partition automatically but also allow extra space for your system to hibernate, use the **--hibernation** option:

```
swap --hibernation
```

The size assigned will be equivalent to the swap space assigned by **--recommended** plus the amount of RAM on your system.

For the swap sizes assigned by these commands, see [Section C.4, "Recommended partitioning scheme"](#) for AMD64, Intel 64, and 64-bit ARM systems.

- **raid.id**

The partition is used for software RAID (see **raid**).

- **pv.id**

The partition is used for LVM (see **logvol**).

- **biosboot**

The partition will be used for a BIOS Boot partition. A 1 MiB BIOS boot partition is necessary on BIOS-based AMD64 and Intel 64 systems using a GUID Partition Table (GPT); the boot loader will be installed into it. It is not necessary on UEFI systems. See also the **bootloader** command.

- **/boot/efi**

An EFI System Partition. A 50 MiB EFI partition is necessary on UEFI-based AMD64, Intel 64, and 64-bit ARM; the recommended size is 200 MiB. It is not necessary on BIOS systems. See also the **bootloader** command.

- **--size=** - The minimum partition size in MiB. Specify an integer value here such as **500** (do not include the unit).



IMPORTANT

If the **--size** value is too small, the installation fails. Set the **--size** value as the minimum amount of space you require. For size recommendations, see [Section C.4, "Recommended partitioning scheme"](#).

- **--grow** - Tells the partition to grow to fill available space (if any), or up to the maximum size setting, if one is specified.



NOTE

If you use **--grow=** without setting **--maxsize=** on a swap partition, Anaconda limits the maximum size of the swap partition. For systems that have less than 2 GB of physical memory, the imposed limit is twice the amount of physical memory. For systems with more than 2 GB, the imposed limit is the size of physical memory plus 2GB.

- **--maxsize=** - The maximum partition size in MiB when the partition is set to grow. Specify an integer value here such as **500** (do not include the unit).
- **--noformat** - Specifies that the partition should not be formatted, for use with the **--onpart** command.
- **--onpart=** or **--usepart=** - Specifies the device on which to place the partition. Uses an existing blank device and format it to the new specified type. For example:

```
partition /home --onpart=hda1
```

puts **/home** on **/dev/hda1**.

These options can also add a partition to a logical volume. For example:

```
partition pv.1 --onpart=hda2
```

The device must already exist on the system; the **--onpart** option will not create it.

It is also possible to specify an entire drive, rather than a partition, in which case Anaconda will

format and use the drive without creating a partition table. Note, however, that installation of GRUB2 is not supported on a device formatted in this way, and must be placed on a drive with a partition table.

```
partition pv.1 --onpart=hdb
```

- **--ondisk=** or **--ondrive=** - Creates a partition (specified by the **part** command) on an existing disk. This command always creates a partition. Forces the partition to be created on a particular disk. For example, **--ondisk=sdb** puts the partition on the second SCSI disk on the system. To specify a multipath device that does not use logical volume management (LVM), use the format **disk/by-id/dm-uuid-mpath-*WWID***, where *WWID* is the world-wide identifier for the device. For example, to specify a disk with *WWID* **2416CD96995134CA5D787F00A5AA11017**, use:

```
part / --fstype=xfs --grow --asprimary --size=8192 --ondisk=disk/by-id/dm-uuid-mpath-2416CD96995134CA5D787F00A5AA11017
```

Multipath devices that use LVM are not assembled until after Anaconda has parsed the Kickstart file. Therefore, you cannot specify these devices in the format **dm-uuid-mpath**. Instead, to specify a multipath device that uses LVM, use the format **disk/by-id/scsi-*WWID***, where *WWID* is the world-wide identifier for the device. For example, to specify a disk with *WWID* **58095BEC5510947BE8C0360F604351918**, use:

```
part / --fstype=xfs --grow --asprimary --size=8192 --ondisk=disk/by-id/scsi-58095BEC5510947BE8C0360F604351918
```



WARNING

Never specify multipath devices by device names like **mpatha**. Device names such as this are not specific to a particular disk. The disk named **/dev/mpatha** during installation might not be the one that you expect it to be. Therefore, the **clearpart** command could target the wrong disk.

- **--asprimary** - Forces the partition to be allocated as a *primary* partition. If the partition cannot be allocated as primary (usually due to too many primary partitions being already allocated), the partitioning process fails. This option only makes sense when the disk uses a Master Boot Record (MBR); for GUID Partition Table (GPT)-labeled disks this option has no meaning.
- **--fsprofile=** - Specifies a usage type to be passed to the program that makes a filesystem on this partition. A usage type defines a variety of tuning parameters to be used when making a filesystem. For this option to work, the filesystem must support the concept of usage types and there must be a configuration file that lists valid types. For **ext2**, **ext3**, **ext4**, this configuration file is **/etc/mke2fs.conf**.
- **--mkfsoptions=** - Specifies additional parameters to be passed to the program that makes a filesystem on this partition. This is similar to **--fsprofile** but works for all filesystems, not just the ones that support the profile concept. No processing is done on the list of arguments, so they

must be supplied in a format that can be passed directly to the `mkfs` program. This means multiple options should be comma-separated or surrounded by double quotes, depending on the filesystem.

- **--fstype=** - Sets the file system type for the partition. Valid values are **xfs**, **ext2**, **ext3**, **ext4**, **swap**, **vfat**, **efi** and **biosboot**.
- **--fsoptions** - Specifies a free form string of options to be used when mounting the filesystem. This string will be copied into the `/etc/fstab` file of the installed system and should be enclosed in quotes.
- **--label=** - assign a label to an individual partition.
- **--recommended** - Determine the size of the partition automatically. For details about the recommended scheme, see [Section C.4, "Recommended partitioning scheme"](#) for AMD64, Intel 64, and 64-bit ARM.



IMPORTANT

This option can only be used for partitions which result in a file system such as the **/boot** partition and **swap** space. It cannot be used to create LVM physical volumes or RAID members.

- **--onbiosdisk** - Forces the partition to be created on a particular disk as discovered by the BIOS.
- **--encrypted** - Specifies that this partition should be encrypted with Linux Unified Key Setup (LUKS), using the passphrase provided in the **--passphrase** option. If you do not specify a passphrase, Anaconda uses the default, system-wide passphrase set with the **autopart --passphrase** command, or stops the installation and prompts you to provide a passphrase if no default is set.



NOTE

When encrypting one or more partitions, Anaconda attempts to gather 256 bits of entropy to ensure the partitions are encrypted securely. Gathering entropy can take some time - the process will stop after a maximum of 10 minutes, regardless of whether sufficient entropy has been gathered.

The process can be sped up by interacting with the installation system (typing on the keyboard or moving the mouse). If you are installing in a virtual machine, you can also attach a **virtio-rng** device (a virtual random number generator) to the guest.

- **--luks-version=LUKS_VERSION** - Specifies which version of LUKS format should be used to encrypt the filesystem. This option is only meaningful if **--encrypted** is specified.
- **--passphrase=** - Specifies the passphrase to use when encrypting this partition. You must use this option together with the **--encrypted** option; by itself it has no effect.
- **--cipher=** - Specifies the type of encryption to use if the Anaconda default **aes-xts-plain64** is not satisfactory. You must use this option together with the **--encrypted** option; by itself it has no effect. Available types of encryption are listed in the [Security hardening](#) document, but Red Hat strongly recommends using either **aes-xts-plain64** or **aes-cbc-essiv:sha256**.
- **--escrowcert=URL_of_X.509_certificate** - Store data encryption keys of all encrypted

partitions as files in **/root**, encrypted using the X.509 certificate from the URL specified with *URL_of_X.509_certificate*. The keys are stored as a separate file for each encrypted partition. This option is only meaningful if **--encrypted** is specified.

- **--backuppassphrase** - Add a randomly-generated passphrase to each encrypted partition. Store these passphrases in separate files in **/root**, encrypted using the X.509 certificate specified with **--escrowcert**. This option is only meaningful if **--escrowcert** is specified.
- **--pbkdf=PBKDF** - Sets Password-Based Key Derivation Function (PBKDF) algorithm for LUKS keyslot. See also the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified.
- **--pbkdf-memory=PBKDF_MEMORY** - Sets the memory cost for PBKDF. See also the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified.
- **--pbkdf-time=PBKDF_TIME** - Sets the number of milliseconds to spend with PBKDF passphrase processing. See also **--iter-time** in the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified, and is mutually exclusive with **--pbkdf-iterations**.
- **--pbkdf-iterations=PBKDF_ITERATIONS** - Sets the number of iterations directly and avoids PBKDF benchmark. See also **--pbkdf-force-iterations** in the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified, and is mutually exclusive with **--pbkdf-time**.
- **--resize=** - Resize an existing partition. When using this option, specify the target size (in MiB) using the **--size=** option and the target partition using the **--onpart=** option.

Notes

- The **part** command is not mandatory, but you must include either **part**, **autopart** or **mount** in your Kickstart script.
- The **--active** option is deprecated in Red Hat Enterprise Linux 8.
- If partitioning fails for any reason, diagnostic messages appear on virtual console 3.
- All partitions created are formatted as part of the installation process unless **--noformat** and **--onpart** are used.
- Device names in the **sdX** (or **/dev/sdX**) format are not guaranteed to be consistent across reboots, which can complicate usage of some Kickstart commands. When a command calls for a device node name, you can instead use any item from **/dev/disk**. For example, instead of:

```
part / --fstype=xfs --onpart=sda1
```

You could use an entry similar to one of the following:

```
part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
```

```
part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1
```

This way the command will always target the same storage device. This is especially useful in large storage environments. See the chapter [Overview of persistent naming attributes](#) in the *Managing storage devices* document for more in-depth information about different ways to consistently refer to storage devices.

- If you lose the LUKS passphrase, any encrypted partitions and their data is completely inaccessible. There is no way to recover a lost passphrase. However, you can save encryption passphrases with the **--escrowcert** and create backup encryption passphrases with the **--backuppassphrase** options.

H.5.14. raid

The **raid** Kickstart command is optional. It assembles a software RAID device.

Syntax

```
raid mntpoint --level=level --device=device-name partitions*
```

Options

- *mntpoint* - Location where the RAID file system is mounted. If it is `/`, the RAID level must be 1 unless a boot partition (`/boot`) is present. If a boot partition is present, the `/boot` partition must be level 1 and the root (`/`) partition can be any of the available types. The *partitions** (which denotes that multiple partitions can be listed) lists the RAID identifiers to add to the RAID array.



IMPORTANT

On IBM Power Systems, if a RAID device has been prepared and has not been reformatted during the installation, ensure that the RAID metadata version is **0.90** if you intend to put the `/boot` and **PReP** partitions on the RAID device.

The default Red Hat Enterprise Linux 7 **mdadm** metadata version is not supported for the boot device.

- **--level=** - RAID level to use (0, 1, 4, 5, 6, or 10). See [Section C.3, “Supported RAID types”](#) for information about various available RAID levels.
- **--device=** - Name of the RAID device to use - for example, **--device=root**.



IMPORTANT

Do not use **mdraid** names in the form of `md0` - these names are not guaranteed to be persistent. Instead, use meaningful names such as **root** or **swap**. Using meaningful names creates a symbolic link from `/dev/md/name` to whichever `/dev/mdX` node is assigned to the array.

If you have an old (v0.90 metadata) array that you cannot assign a name to, you can specify the array by a filesystem label or UUID (for example, **--device=rhel7-root --label=rhel7-root**).

- **--chunksize=** - Sets the chunk size of a RAID storage in KiB. In certain situations, using a different chunk size than the default (**512 KiB**) can improve the performance of the RAID.
- **--spares=** - Specifies the number of spare drives allocated for the RAID array. Spare drives are used to rebuild the array in case of drive failure.
- **--fsprofile=** - Specifies a usage type to be passed to the program that makes a filesystem on this partition. A usage type defines a variety of tuning parameters to be used when making a filesystem. For this option to work, the filesystem must support the concept of usage types and

there must be a configuration file that lists valid types. For ext2, ext3, and ext4, this configuration file is **/etc/mke2fs.conf**.

- **--fstype=** - Sets the file system type for the RAID array. Valid values are **xfs**, **ext2**, **ext3**, **ext4**, **swap**, and **vfat**.
- **--fsoptions=** - Specifies a free form string of options to be used when mounting the filesystem. This string will be copied into the **/etc/fstab** file of the installed system and should be enclosed in quotes.
- **--mkfsoptions=** - Specifies additional parameters to be passed to the program that makes a filesystem on this partition. No processing is done on the list of arguments, so they must be supplied in a format that can be passed directly to the mkfs program. This means multiple options should be comma-separated or surrounded by double quotes, depending on the filesystem.
- **--label=** - Specify the label to give to the filesystem to be made. If the given label is already in use by another filesystem, a new label will be created.
- **--noformat** - Use an existing RAID device and do not format the RAID array.
- **--useexisting** - Use an existing RAID device and reformat it.
- **--encrypted** - Specifies that this RAID device should be encrypted with Linux Unified Key Setup (LUKS), using the passphrase provided in the **--passphrase** option. If you do not specify a passphrase, Anaconda uses the default, system-wide passphrase set with the **autopart --passphrase** command, or stops the installation and prompts you to provide a passphrase if no default is set.



NOTE

When encrypting one or more partitions, Anaconda attempts to gather 256 bits of entropy to ensure the partitions are encrypted securely. Gathering entropy can take some time - the process will stop after a maximum of 10 minutes, regardless of whether sufficient entropy has been gathered.

The process can be sped up by interacting with the installation system (typing on the keyboard or moving the mouse). If you are installing in a virtual machine, you can also attach a **virtio-rng** device (a virtual random number generator) to the guest.

- **--luks-version=LUKS_VERSION** - Specifies which version of LUKS format should be used to encrypt the filesystem. This option is only meaningful if **--encrypted** is specified.
- **--cipher=** - Specifies the type of encryption to use if the Anaconda default **aes-xts-plain64** is not satisfactory. You must use this option together with the **--encrypted** option; by itself it has no effect. Available types of encryption are listed in the [Security hardening](#) document, but Red Hat strongly recommends using either **aes-xts-plain64** or **aes-cbc-essiv:sha256**.
- **--passphrase=** - Specifies the passphrase to use when encrypting this RAID device. You must use this option together with the **--encrypted** option; by itself it has no effect.
- **--escrowcert=URL_of_X.509_certificate** - Store the data encryption key for this device in a file in **/root**, encrypted using the X.509 certificate from the URL specified with **URL_of_X.509_certificate**. This option is only meaningful if **--encrypted** is specified.

- **--backuppassphrase** - Add a randomly-generated passphrase to this device. Store the passphrase in a file in **/root**, encrypted using the X.509 certificate specified with **--escrowcert**. This option is only meaningful if **--escrowcert** is specified.
- **--pbkdf=PBKDF** - Sets Password-Based Key Derivation Function (PBKDF) algorithm for LUKS keyslot. See also the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified.
- **--pbkdf-memory=PBKDF_MEMORY** - Sets the memory cost for PBKDF. See also the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified.
- **--pbkdf-time=PBKDF_TIME** - Sets the number of milliseconds to spend with PBKDF passphrase processing. See also **--iter-time** in the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified, and is mutually exclusive with **--pbkdf-iterations**.
- **--pbkdf-iterations=PBKDF_ITERATIONS** - Sets the number of iterations directly and avoids PBKDF benchmark. See also **--pbkdf-force-iterations** in the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified, and is mutually exclusive with **--pbkdf-time**.

Example

The following example shows how to create a RAID level 1 partition for **/**, and a RAID level 5 for **/home**, assuming there are three SCSI disks on the system. It also creates three swap partitions, one on each drive.

```
part raid.01 --size=6000 --ondisk=sda
part raid.02 --size=6000 --ondisk=sdb
part raid.03 --size=6000 --ondisk=sdc
part swap --size=512 --ondisk=sda
part swap --size=512 --ondisk=sdb
part swap --size=512 --ondisk=sdc
part raid.11 --size=1 --grow --ondisk=sda
part raid.12 --size=1 --grow --ondisk=sdb
part raid.13 --size=1 --grow --ondisk=sdc
raid / --level=1 --device=rhel8-root --label=rhel8-root raid.01 raid.02 raid.03
raid /home --level=5 --device=rhel8-home --label=rhel8-home raid.11 raid.12 raid.13
```

Notes

- If you lose the LUKS passphrase, any encrypted partitions and their data is completely inaccessible. There is no way to recover a lost passphrase. However, you can save encryption passphrases with the **--escrowcert** and create backup encryption passphrases with the **--backuppassphrase** options.

H.5.15. **reqpart**

The **reqpart** Kickstart command is optional. It automatically creates partitions required by your hardware platform. These include a **/boot/efi** partition for systems with UEFI firmware, a **biosboot** partition for systems with BIOS firmware and GPT, and a **PRePBoot** partition for IBM Power Systems.

Syntax

```
reqpart [--add-boot]
```

Options

- **--add-boot** - Creates a separate **/boot** partition in addition to the platform-specific partition created by the base command.

Notes

- This command cannot be used together with **autopart**, because **autopart** does everything the **reqpart** command does and, in addition, creates other partitions or logical volumes such as **/** and **swap**. In contrast with **autopart**, this command only creates platform-specific partitions and leaves the rest of the drive empty, allowing you to create a custom layout.

H.5.16. **snapshot**

The **snapshot** Kickstart command is optional. Use it to create LVM thin volume snapshots during the installation process. This enables you to back up a logical volume before or after the installation.

To create multiple snapshots, add the **snapshot** Kickstart command multiple times.

Syntax

```
snapshot vg_name/lv_name --name=snapshot_name --when=pre-install/post-install
```

Options

- ***vg_name/lv_name*** - Sets the name of the volume group and logical volume to create the snapshot from.
- **--name=*snapshot_name*** - Sets the name of the snapshot. This name must be unique within the volume group.
- **--when=*pre-install/post-install*** - Sets if the snapshot is created before the installation begins or after the installation is completed.

H.5.17. **volgroup**

The **volgroup** Kickstart command is optional. It creates a Logical Volume Management (LVM) group.

Syntax

```
volgroup name [OPTIONS] [partition*]
```

Mandatory options

- *name* - Name of the new volume group.

Options

- *partition* - Physical volume partitions to use as backing storage for the volume group.
- **--noformat** - Use an existing volume group and do not format it.
- **--useexisting** - Use an existing volume group and reformat it. If you use this option, do not specify a *partition*. For example:

```
volgroup rhel00 --useexisting --noformat
```

- **--pesize=** - Set the size of the volume group's physical extents in KiB. The default value is 4096 (4 MiB), and the minimum value is 1024 (1 MiB).
- **--reserved-space=** - Specify an amount of space to leave unused in a volume group in MiB. Applicable only to newly created volume groups.
- **--reserved-percent=** - Specify a percentage of total volume group space to leave unused. Applicable only to newly created volume groups.

Notes

- Create the partition first, then create the logical volume group, and then create the logical volume. For example:

```
part pv.01 --size 10000
volgroup my_volgrp pv.01
logvol / --vgname=my_volgrp --size=2000 --name=root
```

- Do not use the dash (-) character in logical volume and volume group names when installing Red Hat Enterprise Linux using Kickstart. If this character is used, the installation finishes normally, but the **/dev/mapper/** directory will list these volumes and volume groups with every dash doubled. For example, a volume group named **volgrp-01** containing a logical volume named **logvol-01** will be listed as **/dev/mapper/volgrp--01-logvol--01**.

This limitation only applies to newly created logical volume and volume group names. If you are reusing existing ones using the **--noformat** option, their names will not be changed.

H.5.18. zerombr

The **zerombr** Kickstart command is optional. The **zerombr** initializes any invalid partition tables that are found on disks and destroys all of the contents of disks with invalid partition tables. This command is required when performing an installation on an IBM Z system with unformatted Direct Access Storage Device (DASD) disks, otherwise the unformatted disks are not formatted and used during the installation.

Syntax

```
zerombr
```

Notes

- On IBM Z, if **zerombr** is specified, any Direct Access Storage Device (DASD) visible to the installation program which is not already low-level formatted is automatically low-level formatted with dasdfmt. The command also prevents user choice during interactive installations.
- If **zerombr** is not specified and there is at least one unformatted DASD visible to the installation program, a non-interactive Kickstart installation exits unsuccessfully.
- If **zerombr** is not specified and there is at least one unformatted DASD visible to the installation program, an interactive installation exits if the user does not agree to format all visible and unformatted DASDs. To circumvent this, only activate those DASDs that you will use during installation. You can always add more DASDs after installation is complete.

- This command has no options.

H.5.19. zfcp

The **zfcp** Kickstart command is optional. It defines a Fibre channel device.

This option only applies on IBM Z. All of the options described below must be specified.

Syntax

```
zfcp --devnum=devnum --wwpn=wwpn --fcplun=lun
```

Options

- **--devnum=** - The device number (zFCP adapter device bus ID).
- **--wwpn=** - The device's World Wide Port Name (WWPN). Takes the form of a 16-digit number, preceded by **0x**.
- **--fcplun=** - The device's Logical Unit Number (LUN). Takes the form of a 16-digit number, preceded by **0x**.

Example

```
zfcp --devnum=0.0.4000 --wwpn=0x5005076300C213e9 --fcplun=0x5022000000000000
```

H.6. KICKSTART COMMANDS FOR ADDONS SUPPLIED WITH THE RHEL INSTALLATION PROGRAM

The Kickstart commands in this section are related to add-ons supplied by default with the Red Hat Enterprise Linux installation program: Kdump and OpenSCAP.

H.6.1. %addon com_redhat_kdump

The **%addon com_redhat_kdump** Kickstart command is optional. This command configures the kdump kernel crash dumping mechanism.

Syntax

```
%addon com_redhat_kdump [OPTIONS]  
%end
```



NOTE

The syntax for this command is unusual because it is an add-on rather than a built-in Kickstart command.

Notes

Kdump is a kernel crash dumping mechanism that allows you to save the contents of the system's memory for later analysis. It relies on **kexec**, which can be used to boot a Linux kernel from the context of another kernel without rebooting the system, and preserve the contents of the first kernel's memory

that would otherwise be lost.

In case of a system crash, **kexec** boots into a second kernel (a capture kernel). This capture kernel resides in a reserved part of the system memory. Kdump then captures the contents of the crashed kernel's memory (a crash dump) and saves it to a specified location. The location cannot be configured using this Kickstart command; it must be configured after the installation by editing the **/etc/kdump.conf** configuration file.

For more information about Kdump, see the [Installing and configuring kdump](#) chapter of the *Managing, monitoring and updating the kernel* document.

Options

- **--enable** - Enable kdump on the installed system.
- **--disable** - Disable kdump on the installed system.
- **--reserve-mb=** - The amount of memory you want to reserve for kdump, in MiB. For example:

```
%addon com_redhat_kdump --enable --reserve-mb=128
%end
```

You can also specify **auto** instead of a numeric value. In that case, the installation program will determine the amount of memory automatically based on the criteria described in the [Memory requirements for kdump](#) section of the *Managing, monitoring and updating the kernel* document.

If you enable kdump and do not specify a **--reserve-mb=** option, the value **auto** will be used.

- **--enablefadump** - Enable firmware-assisted dumping on systems which allow it (notably, IBM Power Systems servers).

H.6.2. %addon org_fedora_oscrap

The **%addon org_fedora_oscrap** Kickstart command is optional.

The OpenSCAP installation program add-on is used to apply SCAP (Security Content Automation Protocol) content - security policies - on the installed system. This add-on has been enabled by default since Red Hat Enterprise Linux 7.2. When enabled, the packages necessary to provide this functionality will automatically be installed. However, by default, no policies are enforced, meaning that no checks are performed during or after installation unless specifically configured.



IMPORTANT

Applying a security policy is not necessary on all systems. This command should only be used when a specific policy is mandated by your organization rules or government regulations.

Unlike most other commands, this add-on does not accept regular options, but uses key-value pairs in the body of the **%addon** definition instead. These pairs are whitespace-agnostic. Values can be optionally enclosed in single quotes ('') or double quotes ("").

Syntax

```
%addon org_fedora_oscap
key = value
%end
```

Keys

The following keys are recognized by the add-on:

- **content-type** - Type of the security content. Possible values are **datastream**, **archive**, **rpm**, and **scap-security-guide**. If the **content-type** is **scap-security-guide**, the add-on will use content provided by the **scap-security-guide** package, which is present on the boot media. This means that all other keys except **profile** will have no effect.
- **content-url** - Location of the security content. The content must be accessible using HTTP, HTTPS, or FTP; local storage is currently not supported. A network connection must be available to reach content definitions in a remote location.
- **datastream-id** - ID of the data stream referenced in the **content-url** value. Used only if **content-type** is **datastream**.
- **xccdf-id** - ID of the benchmark you want to use.
- **content-path** - Path to the datastream or the XCCDF file which should be used, given as a relative path in the archive.
- **profile** - ID of the profile to be applied. Use **default** to apply the default profile.
- **fingerprint** - A MD5, SHA1 or SHA2 checksum of the content referenced by **content-url**.
- **tailoring-path** - Path to a tailoring file which should be used, given as a relative path in the archive.

Examples

- The following is an example **%addon org_fedora_oscap** section which uses content from the **scap-security-guide** on the installation media:

Example H.1. Sample OpenSCAP Add-on Definition Using SCAP Security Guide

```
%addon org_fedora_oscap
content-type = scap-security-guide
profile = pci-dss
%end
```

- The following is a more complex example which loads a custom profile from a web server:

Example H.2. Sample OpenSCAP Add-on Definition Using a Datastream

```
%addon org_fedora_oscap
content-type = datastream
content-url = http://www.example.com/scap/testing\_ds.xml
datastream-id = scap_example.com_datastream_testing
xccdf-id = scap_example.com_cref_xccdf.xml
```

```
profile = xccdf_example.com_profile_my_profile
fingerprint = 240f2f18222faa98856c3b4fc50c4195
%end
```

Additional resources

- Additional information about the OpenSCAP installation program add-on is available at <https://www.open-scap.org/tools/oscap-anaconda-addon/>.
- For more information about the profiles available in the SCAP Security Guide and what they do, see the [OpenSCAP Portal](#).

H.7. COMMANDS USED IN ANACONDA

The **pwpolicy** command is an Anaconda UI specific command that can be used only in the **%anaconda** section of the kickstart file.

H.7.1. pwpolicy

The **pwpolicy** Kickstart command is optional. Use this command to enforce a custom password policy during installation. The policy requires you to create passwords for the root, users, or the luks user accounts. The factors such as password length and strength decide the validity of a password.

Syntax

```
pwpolicy name [--minlen=length] [--minquality=quality] [--strict|--nostrict] [--emptyok|--noempty] [--changesok|--nochanges]
```

Mandatory options

- *name* - Replace with either **root**, **user** or **luks** to enforce the policy for the **root** password, user passwords, or LUKS passphrase, respectively.

Optional options

- **--minlen=** - Sets the minimum allowed password length, in characters. The default is **6**.
- **--minquality=** - Sets the minimum allowed password quality as defined by the **libpwquality** library. The default value is **1**.
- **--strict** - Enables strict password enforcement. Passwords which do not meet the requirements specified in **--minquality=** and **--minlen=** will not be accepted. This option is disabled by default.
- **--nostrict** - Passwords which do *not* meet the minimum quality requirements specified by the **--minquality=** and **-minlen=** options will be allowed, after **Done** is clicked twice in the GUI. For text mode interface, a similar mechanism is used.
- **--emptyok** - Allows the use of empty passwords. Enabled by default for user passwords.
- **--noempty** - Disallows the use of empty passwords. Enabled by default for the root password and the LUKS passphrase.

- **--changesok** - Allows changing the password in the user interface, even if the Kickstart file already specifies a password. Disabled by default.
- **--nochanges** - Disallows changing passwords which are already set in the Kickstart file. Enabled by default.

Notes

- The **pwpolicy** command is an Anaconda-UI specific command that can be used only in the **%anaconda** section of the kickstart file.
- The **libpwquality** library is used to check minimum password requirements (length and quality). You can use the **pwscore** and **pwmake** commands provided by the **libpwquality** package to check the quality score of a password, or to create a random password with a given score. See the **pwscore(1)** and **pwmake(1)** man page for details about these commands.

H.8. KICKSTART COMMANDS FOR SYSTEM RECOVERY

The Kickstart command in this section repairs an installed system.

H.8.1. rescue

The **rescue** Kickstart command is optional. It provides a shell environment with root privileges and a set of system management tools to repair the installation and to troubleshoot the issues like:

- Mount file systems as read-only
- Blacklist or add a driver provided on a driver disc
- Install or upgrade system packages
- Manage partitions



NOTE

The Kickstart rescue mode is different from the rescue mode and emergency mode, which are provided as part of the `systemd` and service manager.

The **rescue** command does not modify the system on its own. It only sets up the rescue environment by mounting the system under `/mnt/sysimage` in a read-write mode. You can choose not to mount the system, or to mount it in read-only mode.

Syntax

```
rescue [--nomount|--romount]
```

Options

- **--nomount** or **--romount** - Controls how the installed system is mounted in the rescue environment. By default, the installation program finds your system and mount it in read-write mode, telling you where it has performed this mount. You can optionally select to not mount anything (the **--nomount** option) or mount in read-only mode (the **--romount** option). Only one of these two options can be used.

Notes

To run a rescue mode, make a copy of the Kickstart file, and include the **rescue** command in it.

Using the **rescue** command causes the installer to perform the following steps:

1. Run the **%pre** script.

2. Set up environment for rescue mode.

The following kickstart commands take effect:

a. updates

b. sshpw

c. logging

d. lang

e. network

3. Set up advanced storage environment.

The following kickstart commands take effect:

a. fcoe

b. iscsi

c. iscsiname

d. nvdimm

e. zfcp

4. Mount the system

rescue [--nomount|--romount]

5. Run %post script

This step is run only if the installed system is mounted in read-write mode.

6. Start shell

7. Reboot system

PART II. DESIGN OF SECURITY

CHAPTER 29. OVERVIEW OF SECURITY HARDENING IN RHEL

Due to the increased reliance on powerful, networked computers to help run businesses and keep track of our personal information, entire industries have been formed around the practice of network and computer security. Enterprises have solicited the knowledge and skills of security experts to properly audit systems and tailor solutions to fit the operating requirements of their organization. Because most organizations are increasingly dynamic in nature, their workers are accessing critical company IT resources locally and remotely, hence the need for secure computing environments has become more pronounced.

Unfortunately, many organizations, as well as individual users, regard security as more of an afterthought, a process that is overlooked in favor of increased power, productivity, convenience, ease of use, and budgetary concerns. Proper security implementation is often enacted postmortem – after an unauthorized intrusion has already occurred. Taking the correct measures prior to connecting a site to an untrusted network, such as the Internet, is an effective means of thwarting many attempts at intrusion.

29.1. WHAT IS COMPUTER SECURITY?

Computer security is a general term that covers a wide area of computing and information processing. Industries that depend on computer systems and networks to conduct daily business transactions and access critical information regard their data as an important part of their overall assets. Several terms and metrics have entered our daily business vocabulary, such as total cost of ownership (TCO), return on investment (ROI), and quality of service (QoS). Using these metrics, industries can calculate aspects such as data integrity and high-availability (HA) as part of their planning and process management costs. In some industries, such as electronic commerce, the availability and trustworthiness of data can mean the difference between success and failure.

29.2. STANDARDIZING SECURITY

Enterprises in every industry rely on regulations and rules that are set by standards-making bodies such as the American Medical Association (AMA) or the Institute of Electrical and Electronics Engineers (IEEE). The same concepts hold true for information security. Many security consultants and vendors agree upon the standard security model known as CIA, or *Confidentiality, Integrity, and Availability*. This three-tiered model is a generally accepted component to assessing risks of sensitive information and establishing security policy. The following describes the CIA model in further detail:

- Confidentiality – Sensitive information must be available only to a set of pre-defined individuals. Unauthorized transmission and usage of information should be restricted. For example, confidentiality of information ensures that a customer's personal or financial information is not obtained by an unauthorized individual for malicious purposes such as identity theft or credit fraud.
- Integrity – Information should not be altered in ways that render it incomplete or incorrect. Unauthorized users should be restricted from the ability to modify or destroy sensitive information.
- Availability – Information should be accessible to authorized users any time that it is needed. Availability is a warranty that information can be obtained with an agreed-upon frequency and timeliness. This is often measured in terms of percentages and agreed to formally in Service Level Agreements (SLAs) used by network service providers and their enterprise clients.

29.3. CRYPTOGRAPHIC SOFTWARE AND CERTIFICATIONS

Red Hat Enterprise Linux undergoes several security certifications, such as **FIPS 140-2** or **Common Criteria** (CC), to ensure that industry best practices are followed.

The [RHEL 8 core crypto components](#) Knowledgebase article provides an overview of the Red Hat Enterprise Linux 8 core crypto components, documenting which are they, how are they selected, how are they integrated into the operating system, how do they support hardware security modules and smart cards, and how do crypto certifications apply to them.

29.4. SECURITY CONTROLS

Computer security is often divided into three distinct master categories, commonly referred to as **controls**:

- Physical
- Technical
- Administrative

These three broad categories define the main objectives of proper security implementation. Within these controls are sub-categories that further detail the controls and how to implement them.

29.4.1. Physical controls

Physical control is the implementation of security measures in a defined structure used to deter or prevent unauthorized access to sensitive material. Examples of physical controls are:

- Closed-circuit surveillance cameras
- Motion or thermal alarm systems
- Security guards
- Picture IDs
- Locked and dead-bolted steel doors
- Biometrics (includes fingerprint, voice, face, iris, handwriting, and other automated methods used to recognize individuals)

29.4.2. Technical controls

Technical controls use technology as a basis for controlling the access and usage of sensitive data throughout a physical structure and over a network. Technical controls are far-reaching in scope and encompass such technologies as:

- Encryption
- Smart cards
- Network authentication
- Access control lists (ACLs)
- File integrity auditing software

29.4.3. Administrative controls

Administrative controls define the human factors of security. They involve all levels of personnel within an organization and determine which users have access to what resources and information by such means as:

- Training and awareness
- Disaster preparedness and recovery plans
- Personnel recruitment and separation strategies
- Personnel registration and accounting

29.5. VULNERABILITY ASSESSMENT

Given time, resources, and motivation, an attacker can break into nearly any system. All of the security procedures and technologies currently available cannot guarantee that any systems are completely safe from intrusion. Routers help secure gateways to the Internet. Firewalls help secure the edge of the network. Virtual Private Networks safely pass data in an encrypted stream. Intrusion detection systems warn you of malicious activity. However, the success of each of these technologies is dependent upon a number of variables, including:

- The expertise of the staff responsible for configuring, monitoring, and maintaining the technologies.
- The ability to patch and update services and kernels quickly and efficiently.
- The ability of those responsible to keep constant vigilance over the network.

Given the dynamic state of data systems and technologies, securing corporate resources can be quite complex. Due to this complexity, it is often difficult to find expert resources for all of your systems. While it is possible to have personnel knowledgeable in many areas of information security at a high level, it is difficult to retain staff who are experts in more than a few subject areas. This is mainly because each subject area of information security requires constant attention and focus. Information security does not stand still.

A vulnerability assessment is an internal audit of your network and system security; the results of which indicate the confidentiality, integrity, and availability of your network. Typically, vulnerability assessment starts with a reconnaissance phase, during which important data regarding the target systems and resources is gathered. This phase leads to the system readiness phase, whereby the target is essentially checked for all known vulnerabilities. The readiness phase culminates in the reporting phase, where the findings are classified into categories of high, medium, and low risk; and methods for improving the security (or mitigating the risk of vulnerability) of the target are discussed.

If you were to perform a vulnerability assessment of your home, you would likely check each door to your home to see if they are closed and locked. You would also check every window, making sure that they closed completely and latch correctly. This same concept applies to systems, networks, and electronic data. Malicious users are the thieves and vandals of your data. Focus on their tools, mentality, and motivations, and you can then react swiftly to their actions.

29.5.1. Defining assessment and testing

Vulnerability assessments may be broken down into one of two types: *outside looking in* and *inside looking around*.

When performing an outside-looking-in vulnerability assessment, you are attempting to compromise your systems from the outside. Being external to your company provides you with the cracker's point of view. You see what a cracker sees – publicly-routable IP addresses, systems on your *DMZ*, external interfaces of your firewall, and more. *DMZ* stands for "demilitarized zone", which corresponds to a computer or small subnetwork that sits between a trusted internal network, such as a corporate private LAN, and an untrusted external network, such as the public Internet. Typically, the *DMZ* contains devices accessible to Internet traffic, such as web (HTTP) servers, FTP servers, SMTP (e-mail) servers and DNS servers.

When you perform an inside-looking-around vulnerability assessment, you are at an advantage since you are internal and your status is elevated to trusted. This is the point of view you and your co-workers have once logged on to your systems. You see print servers, file servers, databases, and other resources.

There are striking distinctions between the two types of vulnerability assessments. Being internal to your company gives you more privileges than an outsider. In most organizations, security is configured to keep intruders out. Very little is done to secure the internals of the organization (such as departmental firewalls, user-level access controls, and authentication procedures for internal resources). Typically, there are many more resources when looking around inside as most systems are internal to a company. Once you are outside the company, your status is untrusted. The systems and resources available to you externally are usually very limited.

Consider the difference between vulnerability assessments and *penetration tests*. Think of a vulnerability assessment as the first step to a penetration test. The information gleaned from the assessment is used for testing. Whereas the assessment is undertaken to check for holes and potential vulnerabilities, the penetration testing actually attempts to exploit the findings.

Assessing network infrastructure is a dynamic process. Security, both information and physical, is dynamic. Performing an assessment shows an overview, which can turn up false positives and false negatives. A false positive is a result, where the tool finds vulnerabilities which in reality do not exist. A false negative is when it omits actual vulnerabilities.

Security administrators are only as good as the tools they use and the knowledge they retain. Take any of the assessment tools currently available, run them against your system, and it is almost a guarantee that there are some false positives. Whether by program fault or user error, the result is the same. The tool may find false positives, or, even worse, false negatives.

Now that the difference between a vulnerability assessment and a penetration test is defined, take the findings of the assessment and review them carefully before conducting a penetration test as part of your new best practices approach.



WARNING

Do not attempt to exploit vulnerabilities on production systems. Doing so can have adverse effects on productivity and efficiency of your systems and network.

The following list examines some of the benefits of performing vulnerability assessments.

- Creates proactive focus on information security.
- Finds potential exploits before crackers find them.
- Results in systems being kept up to date and patched.

- Promotes growth and aids in developing staff expertise.
- Abates financial loss and negative publicity.

29.5.2. Establishing a methodology for vulnerability assessment

To aid in the selection of tools for a vulnerability assessment, it is helpful to establish a vulnerability assessment methodology. Unfortunately, there is no predefined or industry approved methodology at this time; however, common sense and best practices can act as a sufficient guide.

What is the target? Are we looking at one server, or are we looking at our entire network and everything within the network? Are we external or internal to the company? The answers to these questions are important as they help determine not only which tools to select but also the manner in which they are used.

To learn more about establishing methodologies, see the following website:

- <https://www.owasp.org/> – The Open Web Application Security Project

29.5.3. Vulnerability assessment tools

An assessment can start by using some form of an information-gathering tool. When assessing the entire network, map the layout first to find the hosts that are running. Once located, examine each host individually. Focusing on these hosts requires another set of tools. Knowing which tools to use may be the most crucial step in finding vulnerabilities.

The following tools are just a small sampling of the available tools:

- **Nmap** is a popular tool that can be used to find host systems and open ports on those systems. To install **Nmap** from the **AppStream** repository, enter the **yum install nmap** command as the **root** user. See the **nmap(1)** man page for more information.
- The tools from the **OpenSCAP** suite, such as the **oscap** command-line utility and the **scap-workbench** graphical utility, provides a fully automated compliance audit. See [Scanning the system for security compliance and vulnerabilities](#) for more information.
- Advanced Intrusion Detection Environment (**AIDE**) is a utility that creates a database of files on the system, and then uses that database to ensure file integrity and detect system intrusions. See [Checking integrity with AIDE](#) for more information.

29.6. SECURITY THREATS

29.6.1. Threats to network security

Bad practices when configuring the following aspects of a network can increase the risk of an attack.

Insecure architectures

A misconfigured network is a primary entry point for unauthorized users. Leaving a trust-based, open local network vulnerable to the highly-insecure Internet is much like leaving a door ajar in a crime-ridden neighborhood – nothing may happen for an arbitrary amount of time, but someone exploits the opportunity *eventually*.

Broadcast networks

System administrators often fail to realize the importance of networking hardware in their security

schemes. Simple hardware, such as hubs and routers, relies on the broadcast or non-switched principle; that is, whenever a node transmits data across the network to a recipient node, the hub or router sends a broadcast of the data packets until the recipient node receives and processes the data. This method is the most vulnerable to address resolution protocol (ARP) or media access control (MAC) address spoofing by both outside intruders and unauthorized users on local hosts.

Centralized servers

Another potential networking pitfall is the use of centralized computing. A common cost-cutting measure for many businesses is to consolidate all services to a single powerful machine. This can be convenient as it is easier to manage and costs considerably less than multiple-server configurations. However, a centralized server introduces a single point of failure on the network. If the central server is compromised, it may render the network completely useless or worse, prone to data manipulation or theft. In these situations, a central server becomes an open door that allows access to the entire network.

29.6.2. Threats to server security

Server security is as important as network security because servers often hold a great deal of an organization's vital information. If a server is compromised, all of its contents may become available for the cracker to steal or manipulate at will. The following sections detail some of the main issues.

Unused services and open ports

A full installation of Red Hat Enterprise Linux 8 contains more than 1000 applications and library packages. However, most server administrators do not opt to install every single package in the distribution, preferring instead to install a base installation of packages, including several server applications.

A common occurrence among system administrators is to install the operating system without paying attention to what programs are actually being installed. This can be problematic because unneeded services may be installed, configured with the default settings, and possibly turned on. This can cause unwanted services, such as Telnet, DHCP, or DNS, to run on a server or workstation without the administrator realizing it, which in turn can cause unwanted traffic to the server or even a potential pathway into the system for crackers.

Unpatched services

Most server applications that are included in a default installation are solid, thoroughly tested pieces of software. Having been in use in production environments for many years, their code has been thoroughly refined and many of the bugs have been found and fixed.

However, there is no such thing as perfect software and there is always room for further refinement. Moreover, newer software is often not as rigorously tested as one might expect, because of its recent arrival to production environments or because it may not be as popular as other server software.

Developers and system administrators often find exploitable bugs in server applications and publish the information on bug tracking and security-related websites such as the Bugtraq mailing list (<http://www.securityfocus.com>) or the Computer Emergency Response Team (CERT) website (<http://www.cert.org>). Although these mechanisms are an effective way of alerting the community to security vulnerabilities, it is up to system administrators to patch their systems promptly. This is particularly true because crackers have access to these same vulnerability tracking services and will use the information to crack unpatched systems whenever they can. Good system administration requires vigilance, constant bug tracking, and proper system maintenance to ensure a more secure computing environment.

Inattentive administration

Administrators who fail to patch their systems are one of the greatest threats to server security. This applies as much to inexperienced administrators as it does to overconfident or unmotivated administrators.

Some administrators fail to patch their servers and workstations, while others fail to watch log messages from the system kernel or network traffic. Another common error is when default passwords or keys to services are left unchanged. For example, some databases have default administration passwords because the database developers assume that the system administrator changes these passwords immediately after installation. If a database administrator fails to change this password, even an inexperienced cracker can use a widely-known default password to gain administrative privileges to the database. These are only a few examples of how inattentive administration can lead to compromised servers.

Inherently insecure services

Even the most vigilant organization can fall victim to vulnerabilities if the network services they choose are inherently insecure. For instance, there are many services developed under the assumption that they are used over trusted networks; however, this assumption fails as soon as the service becomes available over the Internet – which is itself inherently untrusted.

One category of insecure network services are those that require unencrypted user names and passwords for authentication. Telnet and FTP are two such services. If packet sniffing software is monitoring traffic between the remote user and such a service user names and passwords can be easily intercepted.

Inherently, such services can also more easily fall prey to what the security industry terms the *man-in-the-middle* attack. In this type of attack, a cracker redirects network traffic by tricking a cracked name server on the network to point to his machine instead of the intended server. Once someone opens a remote session to the server, the attacker's machine acts as an invisible conduit, sitting quietly between the remote service and the unsuspecting user capturing information. In this way a cracker can gather administrative passwords and raw data without the server or the user realizing it.

Another category of insecure services include network file systems and information services such as NFS or NIS, which are developed explicitly for LAN usage but are, unfortunately, extended to include WANs (for remote users). NFS does not, by default, have any authentication or security mechanisms configured to prevent a cracker from mounting the NFS share and accessing anything contained therein. NIS, as well, has vital information that must be known by every computer on a network, including passwords and file permissions, within a plain text ASCII or DBM (ASCII-derived) database. A cracker who gains access to this database can then access every user account on a network, including the administrator's account.

By default, Red Hat Enterprise Linux 8 is released with all such services turned off. However, since administrators often find themselves forced to use these services, careful configuration is critical.

29.6.3. Threats to workstation and home PC security

Workstations and home PCs may not be as prone to attack as networks or servers, but since they often contain sensitive data, such as credit card information, they are targeted by system crackers. Workstations can also be co-opted without the user's knowledge and used by attackers as "slave" machines in coordinated attacks. For these reasons, knowing the vulnerabilities of a workstation can save users the headache of reinstalling the operating system, or worse, recovering from data theft.

Bad passwords

Bad passwords are one of the easiest ways for an attacker to gain access to a system.

Vulnerable client applications

Although an administrator may have a fully secure and patched server, that does not mean remote users are secure when accessing it. For instance, if the server offers Telnet or FTP services over a public network, an attacker can capture the plain text user names and passwords as they pass over the network, and then use the account information to access the remote user's workstation.

Even when using secure protocols, such as SSH, a remote user may be vulnerable to certain attacks if they do not keep their client applications updated. For instance, SSH protocol version 1 clients are vulnerable to an X-forwarding attack from malicious SSH servers. Once connected to the server, the attacker can quietly capture any keystrokes and mouse clicks made by the client over the network. This problem was fixed in the SSH version 2 protocol, but it is up to the user to keep track of what applications have such vulnerabilities and update them as necessary.

29.7. COMMON EXPLOITS AND ATTACKS

Table 29.1, “Common exploits” details some of the most common exploits and entry points used by intruders to access organizational network resources. Key to these common exploits are the explanations of how they are performed and how administrators can properly safeguard their network against such attacks.

Table 29.1. Common exploits

Exploit	Description	Notes
Null or default passwords	Leaving administrative passwords blank or using a default password set by the product vendor. This is most common in hardware such as routers and firewalls, but some services that run on Linux can contain default administrator passwords as well (though Red Hat Enterprise Linux 8 does not ship with them).	Commonly associated with networking hardware such as routers, firewalls, VPNs, and network attached storage (NAS) appliances. Common in many legacy operating systems, especially those that bundle services (such as UNIX and Windows.) Administrators sometimes create privileged user accounts in a rush and leave the password null, creating a perfect entry point for malicious users who discover the account.
Default shared keys	Secure services sometimes package default security keys for development or evaluation testing purposes. If these keys are left unchanged and are placed in a production environment on the Internet, all users with the same default keys have access to that shared-key resource, and any sensitive information that it contains.	Most common in wireless access points and preconfigured secure server appliances.

Exploit	Description	Notes
IP spoofing	<p>A remote machine acts as a node on your local network, finds vulnerabilities with your servers, and installs a backdoor program or Trojan horse to gain control over your network resources.</p>	<p>Spoofing is quite difficult as it involves the attacker predicting TCP/IP sequence numbers to coordinate a connection to target systems, but several tools are available to assist crackers in performing such a vulnerability.</p> <p>Depends on target system running services (such as rsh, telnet, FTP and others) that use <i>source-based</i> authentication techniques, which are not recommended when compared to PKI or other forms of encrypted authentication used in ssh or SSL/TLS.</p>
Eavesdropping	<p>Collecting data that passes between two active nodes on a network by eavesdropping on the connection between the two nodes.</p>	<p>This type of attack works mostly with plain text transmission protocols such as Telnet, FTP, and HTTP transfers.</p> <p>Remote attacker must have access to a compromised system on a LAN in order to perform such an attack; usually the cracker has used an active attack (such as IP spoofing or man-in-the-middle) to compromise a system on the LAN.</p> <p>Preventative measures include services with cryptographic key exchange, one-time passwords, or encrypted authentication to prevent password snooping; strong encryption during transmission is also advised.</p>

Exploit	Description	Notes
Service vulnerabilities	<p>An attacker finds a flaw or loophole in a service run over the Internet; through this vulnerability, the attacker compromises the entire system and any data that it may hold, and could possibly compromise other systems on the network.</p>	<p>HTTP-based services such as CGI are vulnerable to remote command execution and even interactive shell access. Even if the HTTP service runs as a non-privileged user such as "nobody", information such as configuration files and network maps can be read, or the attacker can start a denial of service attack which drains system resources or renders it unavailable to other users.</p> <p>Services sometimes can have vulnerabilities that go unnoticed during development and testing; these vulnerabilities (such as <i>buffer overflows</i>, where attackers crash a service using arbitrary values that fill the memory buffer of an application, giving the attacker an interactive command prompt from which they may execute arbitrary commands) can give complete administrative control to an attacker.</p> <p>Administrators should make sure that services do not run as the root user, and should stay vigilant of patches and errata updates for applications from vendors or security organizations such as CERT and CVE.</p>

Exploit	Description	Notes
Application vulnerabilities	<p>Attackers find faults in desktop and workstation applications (such as email clients) and execute arbitrary code, implant Trojan horses for future compromise, or crash systems. Further exploitation can occur if the compromised workstation has administrative privileges on the rest of the network.</p>	<p>Workstations and desktops are more prone to exploitation as workers do not have the expertise or experience to prevent or detect a compromise; it is imperative to inform individuals of the risks they are taking when they install unauthorized software or open unsolicited email attachments.</p> <p>Safeguards can be implemented such that email client software does not automatically open or execute attachments. Additionally, the automatic update of workstation software using Red Hat Network; or other system management services can alleviate the burdens of multi-seat security deployments.</p>
Denial of Service (DoS) attacks	<p>Attacker or group of attackers coordinate against an organization's network or server resources by sending unauthorized packets to the target host (either server, router, or workstation). This forces the resource to become unavailable to legitimate users.</p>	<p>The most reported DoS case in the US occurred in 2000. Several highly-trafficked commercial and government sites were rendered unavailable by a coordinated ping flood attack using several compromised systems with high bandwidth connections acting as zombies, or redirected broadcast nodes.</p> <p>Source packets are usually forged (as well as rebroadcast), making investigation as to the true source of the attack difficult.</p> <p>Advances in ingress filtering (IETF rfc2267) using iptables and Network Intrusion Detection Systems such as snort assist administrators in tracking down and preventing distributed DoS attacks.</p>

CHAPTER 30. SECURING RHEL DURING INSTALLATION

Security begins even before you start the installation of Red Hat Enterprise Linux. Configuring your system securely from the beginning makes it easier to implement additional security settings later.

30.1. BIOS AND UEFI SECURITY

Password protection for the BIOS (or BIOS equivalent) and the boot loader can prevent unauthorized users who have physical access to systems from booting using removable media or obtaining root privileges through single user mode. The security measures you should take to protect against such attacks depends both on the sensitivity of the information on the workstation and the location of the machine.

For example, if a machine is used in a trade show and contains no sensitive information, then it may not be critical to prevent such attacks. However, if an employee’s laptop with private, unencrypted SSH keys for the corporate network is left unattended at that same trade show, it could lead to a major security breach with ramifications for the entire company.

If the workstation is located in a place where only authorized or trusted people have access, however, then securing the BIOS or the boot loader may not be necessary.

30.1.1. BIOS passwords

The two primary reasons for password protecting the BIOS of a computer are^[1]:

1. **Preventing changes to BIOS settings** – If an intruder has access to the BIOS, they can set it to boot from a CD-ROM or a flash drive. This makes it possible for them to enter rescue mode or single user mode, which in turn allows them to start arbitrary processes on the system or copy sensitive data.
2. **Preventing system booting** – Some BIOSes allow password protection of the boot process. When activated, an attacker is forced to enter a password before the BIOS launches the boot loader.

Because the methods for setting a BIOS password vary between computer manufacturers, consult the computer’s manual for specific instructions.

If you forget the BIOS password, it can either be reset with jumpers on the motherboard or by disconnecting the CMOS battery. For this reason, it is good practice to lock the computer case if possible. However, consult the manual for the computer or motherboard before attempting to disconnect the CMOS battery.

30.1.1.1. Non-BIOS-based systems security

Other systems and architectures use different programs to perform low-level tasks roughly equivalent to those of the BIOS on x86 systems. For example, the *Unified Extensible Firmware Interface (UEFI)* shell.

For instructions on password protecting BIOS-like programs, see the manufacturer’s instructions.

30.2. DISK PARTITIONING

Red Hat recommends creating separate partitions for the **/boot**, **/**, **/home**, **/tmp**, and **/var/tmp** directories. The reasons for each are different, and we will address each partition.

/boot

This partition is the first partition that is read by the system during boot up. The boot loader and kernel images that are used to boot your system into Red Hat Enterprise Linux 8 are stored in this partition. This partition should not be encrypted. If this partition is included in / and that partition is encrypted or otherwise becomes unavailable then your system will not be able to boot.

/home

When user data (**/home**) is stored in / instead of in a separate partition, the partition can fill up causing the operating system to become unstable. Also, when upgrading your system to the next version of Red Hat Enterprise Linux 8 it is a lot easier when you can keep your data in the **/home** partition as it will not be overwritten during installation. If the root partition (/) becomes corrupt your data could be lost forever. By using a separate partition there is slightly more protection against data loss. You can also target this partition for frequent backups.

/tmp and /var/tmp

Both the **/tmp** and **/var/tmp** directories are used to store data that does not need to be stored for a long period of time. However, if a lot of data floods one of these directories it can consume all of your storage space. If this happens and these directories are stored within / then your system could become unstable and crash. For this reason, moving these directories into their own partitions is a good idea.



NOTE

During the installation process, you have an option to encrypt partitions. You must supply a passphrase. This passphrase serves as a key to unlock the bulk encryption key, which is used to secure the partition's data.

30.3. RESTRICTING NETWORK CONNECTIVITY DURING THE INSTALLATION PROCESS

When installing Red Hat Enterprise Linux 8, the installation medium represents a snapshot of the system at a particular time. Because of this, it may not be up-to-date with the latest security fixes and may be vulnerable to certain issues that were fixed only after the system provided by the installation medium was released.

When installing a potentially vulnerable operating system, always limit exposure only to the closest necessary network zone. The safest choice is the “no network” zone, which means to leave your machine disconnected during the installation process. In some cases, a LAN or intranet connection is sufficient while the Internet connection is the riskiest. To follow the best security practices, choose the closest zone with your repository while installing Red Hat Enterprise Linux 8 from a network.

30.4. INSTALLING THE MINIMUM AMOUNT OF PACKAGES REQUIRED

It is best practice to install only the packages you will use because each piece of software on your computer could possibly contain a vulnerability. If you are installing from the DVD media, take the opportunity to select exactly what packages you want to install during the installation. If you find you need another package, you can always add it to the system later.

30.5. POST-INSTALLATION PROCEDURES

The following steps are the security-related procedures that should be performed immediately after installation of Red Hat Enterprise Linux.

1. Update your system. Enter the following command as root:

```
# yum update
```

2. Even though the firewall service, **firewalld**, is automatically enabled with the installation of Red Hat Enterprise Linux, there are scenarios where it might be explicitly disabled, for example in the kickstart configuration. In such a case, it is recommended to consider re-enabling the firewall. To start **firewalld** enter the following commands as root:

```
# systemctl start firewalld  
# systemctl enable firewalld
```

3. To enhance security, disable services you do not need. For example, if there are no printers installed on your computer, disable the **cups** service using the following command:

```
# systemctl disable cups
```

To review active services, enter the following command:

```
$ systemctl list-units | grep service
```

[1] Since system BIOSes differ between manufacturers, some may not support password protection of either type, while others may support one type but not the other.

CHAPTER 31. USING SYSTEM-WIDE CRYPTOGRAPHIC POLICIES

Crypto policies is a system component that configures the core cryptographic subsystems, covering the TLS, IPsec, SSH, DNSsec, and Kerberos protocols. It provides a small set of policies, which the administrator can select.

31.1. SYSTEM-WIDE CRYPTOGRAPHIC POLICIES

Once a system-wide policy is set up, applications in RHEL follow it and refuse to use algorithms and protocols that do not meet the policy, unless you explicitly request the application to do so. That is, the policy applies to the default behavior of applications when running with the system-provided configuration but you can override it if required so.

Red Hat Enterprise Linux 8 contains the following policy levels:

DEFAULT	The default system-wide cryptographic policy level offers secure settings for current threat models. It allows the TLS 1.2 and 1.3 protocols, as well as the IKEv2 and SSH2 protocols. The RSA keys and Diffie-Hellman parameters are accepted if they are at least 2048 bits long.
LEGACY	This policy ensures maximum compatibility with Red Hat Enterprise Linux 5 and earlier; it is less secure due to an increased attack surface. In addition to the DEFAULT level algorithms and protocols, it includes support for the TLS 1.0 and 1.1 protocols. The algorithms DSA, 3DES, and RC4 are allowed, while RSA keys and Diffie-Hellman parameters are accepted if they are at least 1023 bits long.
FUTURE	A conservative security level that is believed to withstand any near-term future attacks. This level does not allow the use of SHA-1 in signature algorithms. The RSA keys and Diffie-Hellman parameters are accepted if they are at least 3072 bits long.
FIPS	A policy level that conforms with the FIPS140-2 requirements. This is used internally by the fips-mode-setup tool, which switches the RHEL system into FIPS mode.



IMPORTANT

Because a cryptographic key used by a certificate on the Customer Portal API does not meet the requirements by the **FUTURE** system-wide cryptographic policy, the **redhat-support-tool** utility does not work with this policy level at the moment.

To work around this problem, use the **DEFAULT** crypto policy while connecting to the Customer Portal API.



NOTE

The specific algorithms and ciphers described in the policy levels as allowed are available only if an application supports them.

Tool for managing crypto policies

To view or change the current system-wide cryptographic policy, use the **update-crypto-policies** tool, for example:

```
$ update-crypto-policies --show
DEFAULT
# update-crypto-policies --set FUTURE
Setting system policy to FUTURE
```

To ensure that the change of the cryptographic policy is applied, restart the system.

Strong crypto defaults by removing insecure cipher suites and protocols

The following list contains cipher suites and protocols removed from the core cryptographic libraries in RHEL 8. They are not present in the sources, or their support is disabled during the build, so applications cannot use them.

- DES (since RHEL 7)
- All export grade cipher suites (since RHEL 7)
- MD5 in signatures (since RHEL 7)
- SSLv2 (since RHEL 7)
- SSLv3 (since RHEL 8)
- All ECC curves < 224 bits (since RHEL 6)
- All binary field ECC curves (since RHEL 6)

Cipher suites and protocols disabled in all policy levels

The following cipher suites and protocols are disabled in all crypto policy levels. They can be enabled only by an explicit configuration of individual applications.

- DH with parameters < 1024 bits
- RSA with key size < 1024 bits
- Camellia
- ARIA
- SEED
- IDEA
- Integrity-only cipher suites
- TLS CBC mode cipher suites using SHA-384 HMAC
- AES-CCM8
- All ECC curves incompatible with TLS 1.3, including secp256k1
- IKEv1 (since RHEL 8)

Cipher suites and protocols enabled in the crypto-policies levels

The following table shows the enabled cipher suites and protocols in all four crypto-policies levels.

	LEGACY	DEFAULT	FIPS	FUTURE
IKEv1	no	no	no	no
3DES	yes	no	no	no
RC4	yes	no	no	no
DH	min. 1024-bit	min. 2048-bit	min. 2048-bit	min. 3072-bit
RSA	min. 1024-bit	min. 2048-bit	min. 2048-bit	min. 3072-bit
DSA	yes	no	no	no
TLS v1.0	yes	no	no	no
TLS v1.1	yes	no	no	no
SHA-1 in digital signatures	yes	yes	no	no
CBC mode ciphers	yes	yes	yes	no
Symmetric ciphers with keys < 256 bits	yes	yes	yes	no
SHA-1 and SHA-224 signatures in certificates	yes	yes	yes	no

Additional resources

- For more details, see the **update-crypto-policies(8)** man page.

31.2. SWITCHING THE SYSTEM-WIDE CRYPTOGRAPHIC POLICY TO MODE COMPATIBLE WITH EARLIER RELEASES

The default system-wide cryptographic policy in Red Hat Enterprise Linux 8 does not allow communication using older, insecure protocols. For environments that require to be compatible with Red Hat Enterprise Linux 5 and in some cases also with earlier releases, the less secure **LEGACY** policy level is available.

**WARNING**

Switching to the **LEGACY** policy level results in a less secure system and applications.

Procedure

1. To switch the system-wide cryptographic policy to the **LEGACY** level, enter the following command as **root**:

```
# update-crypto-policies --set LEGACY
Setting system policy to LEGACY
```

Additional resources

- For the list of available cryptographic policy levels, see the **update-crypto-policies(8)** man page.

31.3. SWITCHING THE SYSTEM TO FIPS MODE

The system-wide cryptographic policies contain a policy level that enables cryptographic modules self-checks in accordance with the requirements by Federal Information Processing Standard (FIPS) Publication 140-2. The **fips-mode-setup** tool that enables or disables FIPS mode internally uses the **FIPS** system-wide cryptographic policy level.

Procedure

1. To switch the system to FIPS mode in RHEL 8:

```
# fips-mode-setup --enable
Setting system policy to FIPS
FIPS mode will be enabled.
Please reboot the system for the setting to take effect.
```

2. Restart your system to allow the kernel to switch to FIPS mode:

```
# reboot
```

Verification steps

1. After the restart, you can check the current state of FIPS mode:

```
# fips-mode-setup --check
FIPS mode is enabled.
```

Additional resources

- The **fips-mode-setup(8)** man page.

- List of RHEL 8 applications using cryptography that are not compliant with FIPS 140-2
- For more details on FIPS 140-2, see the [Security Requirements for Cryptographic Modules](#) on the National Institute of Standards and Technology (NIST) web site.

31.4. ENABLING FIPS MODE IN A CONTAINER

To enable cryptographic modules self-checks in accordance with the requirements by Federal Information Processing Standard (FIPS) Publication 140-2 in a container:

Prerequisites

- The host system must be switched in FIPS mode first, see [Switching the system to FIPS mode](#).

Procedure

1. Mount the **/etc/system-fips** file on the container from the host.
2. Set the FIPS cryptographic policy level in the container:

```
$ update-crypto-policies --set FIPS
```

RHEL 8.2 introduced an alternative method for switching a container to FIPS mode. It requires only using the following command in the container:

```
# mount --bind /usr/share/crypto-policies/back-ends/FIPS /etc/crypto-policies/back-ends
```



NOTE

In RHEL 8, the **fips-mode-setup** command does not work properly in a container and it cannot be used to enable or check FIPS mode in this scenario.

31.5. EXCLUDING AN APPLICATION FROM FOLLOWING SYSTEM-WIDE CRYPTO POLICIES

You can customize cryptographic settings used by your application preferably by configuring supported cipher suites and protocols directly in the application.

You can also remove a symlink related to your application from the **/etc/crypto-policies/back-ends** directory and replace it with your customized cryptographic settings. This configuration prevents the use of system-wide cryptographic policies for applications that use the excluded back end. Furthermore, this modification is not supported by Red Hat.

31.5.1. Examples of opting out of system-wide crypto policies

wget

To customize cryptographic settings used by the **wget** network downloader, use **--secure-protocol** and **--ciphers** options. For example:

```
# wget --secure-protocol=TLSv1_1 --ciphers="SECURE128" https://example.com
```

See the HTTPS (SSL/TLS) Options section of the **wget(1)** man page for more information.

curl

To specify ciphers used by the **curl** tool, use the **--ciphers** option and provide a colon-separated list of ciphers as a value. For example:

```
# curl https://example.com --ciphers DES-CBC3-SHA:RSA-DES-CBC3-SHA
```

See the **curl(1)** man page for more information.

Firefox

Even though you cannot opt out of system-wide cryptographic policies in the **Firefox** web browser, you can further restrict supported ciphers and TLS versions in Firefox's Configuration Editor. Type **about:config** in the address bar and change the value of the **security.tls.version.min** option as required. Setting **security.tls.version.min** to **1** allows TLS 1.0 as the minimum required, **security.tls.version.min 2** enables TLS 1.1, and so on.

OpenSSH

To opt out of the system-wide crypto policies for your **OpenSSH** server, uncomment the line with the **CRYPTO_POLICY=** variable in the **/etc/sysconfig/sshd** file. After this change, values that you specify in the **Ciphers**, **MACs**, **KexAlgorithms**, and **GSSAPIKexAlgorithms** sections in the **/etc/ssh/sshd_config** file are not overridden. See the **sshd_config(5)** man page for more information.

Libreswan

To allow the **Libreswan** IPsec suite to use the IKEv1 protocol, comment out the following line in the **/etc/ipsec.conf** file:

```
include /etc/crypto-policies/back-ends/libreswan.config
```

Then add the **ikev2=never** option to your connection configuration. See the **ipsec.conf(5)** man page for more information.

Additional resources

- For more details, see the **update-crypto-policies(8)** man page.

31.6. CUSTOMIZING SYSTEM-WIDE CRYPTOGRAPHIC POLICIES WITH POLICY MODIFIERS

Use this procedure to adjust certain algorithms or protocols of any system-wide cryptographic policy level or a full custom policy.



NOTE

Customization of system-wide cryptographic policies is available from RHEL 8.2.

Procedure

- Create policy modules that contain your adjustments, for example:

```
# cat /etc/crypto-policies/policies/modules/MYCRYPTO1.pmod
```

```
sha1_in_certs = 0
min_rsa_size = 3072
# cat /etc/crypto-policies/policies/modules/NO-CAMELLIA.pmod
cipher = -CAMELLIA-256-GCM -CAMELLIA-256-CBC -CAMELLIA-128-GCM -CAMELLIA-
128-CBC
```



IMPORTANT

Use upper-case letters in file names of policy modules.

2. Apply your policy adjustments to the **DEFAULT** system-wide cryptographic policy level:

```
# update-crypto-policies --set DEFAULT:MYCRYPTO1:NO-CAMELLIA
```

3. To make your cryptographic settings effective for already running services and applications, restart the system:

```
# reboot
```

Additional resources

- For more details, see the **Custom Policies** section in the **update-crypto-policies(8)** man page and the **Crypto Policy Definition Format** section in the **update-crypto-policies(8)** man page.

31.7. CREATING AND SETTING A CUSTOM SYSTEM-WIDE CRYPTOGRAPHIC POLICY

The following steps demonstrate customizing the system-wide cryptographic policies by a complete policy file.



NOTE

Customization of system-wide cryptographic policies is available from RHEL 8.2.

Procedure

1. Create a policy file for your customizations:

```
# cd /etc/crypto-policies/policies/
# touch MYPOLICY.pol
```

Alternatively, start by copying one of the four predefined policy levels:

```
# cp /usr/share/crypto-policies/DEFAULT.pol /etc/crypto-policies/policies/MYPOLICY.pol
```

2. Edit the file with your custom cryptographic policy in a text editor of your choice to fit your requirements, for example:

```
# vi /etc/crypto-policies/policies/MYPOLICY.pol
```

3. Switch the system-wide cryptographic policy to your custom level:

```
# update-crypto-policies --set MYPOLICY
```

4. To make your cryptographic settings effective for already running services and applications, restart the system:

```
# reboot
```

Additional resources

- For more details, see the **Custom Policies** section in the **[update-crypto-policies\(8\)](#)** man page and the **Crypto Policy Definition Format** section in the **[update-crypto-policies\(8\)](#)** man page.

31.8. RELATED INFORMATION

- See the [System-wide crypto policies in RHEL 8](#) and [Strong crypto defaults in RHEL 8 and deprecation of weak crypto algorithms](#) Knowledgebase articles on the Red Hat Customer Portal for more information.

CHAPTER 32. CONFIGURING APPLICATIONS TO USE CRYPTOGRAPHIC HARDWARE THROUGH PKCS #11

Separating parts of your secret information on dedicated cryptographic devices, such as smart cards and cryptographic tokens for end-user authentication and hardware security modules (HSM) for server applications, provides an additional layer of security. In Red Hat Enterprise Linux 8, support for cryptographic hardware through the PKCS #11 API is consistent across different applications, and the isolation of secrets on cryptographic hardware is not a complicated task.

32.1. CRYPTOGRAPHIC HARDWARE SUPPORT THROUGH PKCS #11

PKCS #11 (Public-Key Cryptography Standard) defines an application programming interface (API) to cryptographic devices that hold cryptographic information and perform cryptographic functions. These devices are called tokens, and they can be implemented in a hardware or software form.

A PKCS #11 token can store various object types including a certificate; a data object; and a public, private, or secret key. These objects are uniquely identifiable through the PKCS #11 URI scheme.

A PKCS #11 URI is a standard way to identify a specific object in a PKCS #11 module according to the object attributes. This enables you to configure all libraries and applications with the same configuration string in the form of a URI.

Red Hat Enterprise Linux 8 provides the OpenSC PKCS #11 driver for smart cards by default. However, hardware tokens and HSMs can have their own PKCS #11 modules that do not have their counterpart in Red Hat Enterprise Linux. You can register such PKCS #11 modules with the **p11-kit** tool, which acts as a wrapper over the registered smart card drivers in the system.

To make your own PKCS #11 module work on the system, add a new text file to the **/etc/pkcs11/modules/** directory

You can add your own PKCS #11 module into the system by creating a new text file in the **/etc/pkcs11/modules/** directory. For example, the OpenSC configuration file in **p11-kit** looks as follows:

```
$ cat /usr/share/p11-kit/modules/opensc.module
module: opensc-pkcs11.so
```

Additional resources

- [Consistent PKCS #11 support in Red Hat Enterprise Linux 8](#)
- [The PKCS #11 URI Scheme](#)
- [Controlling access to smart cards](#)

32.2. USING SSH KEYS STORED ON A SMART CARD

Red Hat Enterprise Linux 8 enables you to use RSA and ECDSA keys stored on a smart card on OpenSSH clients. Use this procedure to enable authentication using a smart card instead of using a password.

Prerequisites

- On the client side, the **opensc** package is installed and the **pcscd** service is running.

Procedure

1. List all keys provided by the OpenSC PKCS #11 module including their PKCS #11 URIs and save the output to the `keys.pub` file:

```
$ ssh-keygen -D pkcs11: > keys.pub
$ ssh-keygen -D pkcs11:
ssh-rsa AAAAB3NzaC1yc2E...KKZMzcQZzx
pkcs11:id=%02;object=SIGN%20pubkey;token=SSH%20key;manufacturer=piv_II?module-
path=/usr/lib64/pkcs11/opensc-pkcs11.so
ecdsa-sha2-nistp256 AAA...J0hkYnnsM=
pkcs11:id=%01;object=PIV%20AUTH%20pubkey;token=SSH%20key;manufacturer=piv_II?
module-path=/usr/lib64/pkcs11/opensc-pkcs11.so
```

2. To enable authentication using a smart card on a remote server (`example.com`), transfer the public key to the remote server. Use the **ssh-copy-id** command with `keys.pub` created in the previous step:

```
$ ssh-copy-id -f -i keys.pub username@example.com
```

3. To connect to `example.com` using the ECDSA key from the output of the **ssh-keygen -D** command in step 1, you can use just a subset of the URI, which uniquely references your key, for example:

```
$ ssh -i "pkcs11:id=%01?module-path=/usr/lib64/pkcs11/opensc-pkcs11.so" example.com
Enter PIN for 'SSH key':
[example.com] $
```

4. You can use the same URI string in the `~/.ssh/config` file to make the configuration permanent:

```
$ cat ~/.ssh/config
IdentityFile "pkcs11:id=%01?module-path=/usr/lib64/pkcs11/opensc-pkcs11.so"
$ ssh example.com
Enter PIN for 'SSH key':
[example.com] $
```

Because OpenSSH uses the **p11-kit-proxy** wrapper and the OpenSC PKCS #11 module is registered to PKCS#11 Kit, you can simplify the previous commands:

```
$ ssh -i "pkcs11:id=%01" example.com
Enter PIN for 'SSH key':
[example.com] $
```

If you skip the **id=** part of a PKCS #11 URI, OpenSSH loads all keys that are available in the proxy module. This can reduce the amount of typing required:

```
$ ssh -i pkcs11: example.com
Enter PIN for 'SSH key':
[example.com] $
```

Additional resources

- [Fedora 28: Better smart card support in OpenSSH](#)

- **p11-kit(8)** man page
- **ssh(1)** man page
- **ssh-keygen(1)** man page
- **opensc.conf(5)** man page
- **pcscd(8)** man page

32.3. USING HSMS PROTECTING PRIVATE KEYS IN APACHE AND NGINX

The **Apache** and **Nginx** HTTP servers can work with private keys stored on hardware security modules (HSMs), which helps to prevent the keys' disclosure and man-in-the-middle attacks. Note that this usually requires high-performance HSMs for busy servers.

Apache HTTP server

For secure communication in the form of the HTTPS protocol, the **Apache** HTTP server (**httpd**) uses the OpenSSL library. OpenSSL does not support PKCS #11 natively. To utilize HSMs, you have to install the **openssl-pkcs11** package, which provides access to PKCS #11 modules through the engine interface. You can use a PKCS #11 URI instead of a regular file name to specify a server key and a certificate in the **/etc/httpd/conf.d/ssl.conf** configuration file, for example:

```
SSLCertificateFile "pkcs11:id=%01;token=softhsm;type=cert"
SSLCertificateKeyFile "pkcs11:id=%01;token=softhsm;type=private?pin-value=111111"
```

Install the **httpd-manual** package to obtain complete documentation for the **Apache** HTTP Server, including TLS configuration. The directives available in the **/etc/httpd/conf.d/ssl.conf** configuration file are described in detail in /usr/share/httpd/manual/mod/mod_ssl.html.

Nginx HTTP and proxy server

Because **Nginx** also uses the OpenSSL for cryptographic operations, support for PKCS #11 must go through the **openssl-pkcs11** engine. **Nginx** currently supports only loading private keys from an HSM, and a certificate must be provided separately as a regular file. Modify the **ssl_certificate** and **ssl_certificate_key** options in the **server** section of the **/etc/nginx/nginx.conf** configuration file:

```
ssl_certificate /path/to/cert.pem
ssl_certificate_key "engine:pkcs11:pkcs11:token=softhsm:id=%01:type=private?pin-value=111111";
```

Note that the **engine:pkcs11:** prefix is needed for the PKCS #11 URI in the **Nginx** configuration file. This is because the other **pkcs11** prefix refers to the engine name.

32.4. CONFIGURING APPLICATIONS TO AUTHENTICATE USING CERTIFICATES FROM SMART CARDS

- The **wget** network downloader enables you to specify PKCS #11 URIs instead of paths to locally stored private keys, and thus simplifies creating scripts for tasks that require safely stored private keys and certificates. For example:

```
$ wget --private-key 'pkcs11:token=softhsm:id=%01:type=private?pin-value=111111' --
certificate 'pkcs11:token=softhsm:id=%01:type=cert' https://example.com/
```

See the **wget(1)** man page for more information.

- Specifying PKCS #11 URI for use by the **curl** tool is analogous:

```
$ curl --key 'pkcs11:token=softhsm;id=%01;type=private?pin-value=111111' --cert  
'pkcs11:token=softhsm;id=%01;type=cert' https://example.com/
```

See the **curl(1)** man page for more information.

- The **Firefox** web browser automatically loads the **p11-kit-proxy** module. This means that every supported smart card in the system is automatically detected. For using TLS client authentication, no additional setup is required and keys from a smart card are automatically used when a server requests them.

Using PKCS #11 URIs in custom applications

If your application uses the **GnuTLS** or **NSS** library, support for PKCS #11 URIs is ensured by their built-in support for PKCS #11. Also, applications relying on the **OpenSSL** library can access cryptographic hardware modules thanks to the **openssl-pkcs11** engine.

With applications that require working with private keys on smart cards and that do not use **NSS**, **GnuTLS**, or **OpenSSL**, use **p11-kit** to implement registering PKCS #11 modules.

See the **p11-kit(8)** man page for more information.

32.5. RELATED INFORMATION

- **pkcs11.conf(5)** man page

CHAPTER 33. USING SHARED SYSTEM CERTIFICATES

The shared system certificates storage enables NSS, GnuTLS, OpenSSL, and Java to share a default source for retrieving system certificate anchors and black list information. By default, the trust store contains the Mozilla CA list, including positive and negative trust. The system allows updating the core Mozilla CA list or choosing another certificate list.

33.1. THE SYSTEM-WIDE TRUST STORE

In Red Hat Enterprise Linux, the consolidated system-wide trust store is located in the **/etc/pki/ca-trust/** and **/usr/share/pki/ca-trust-source/** directories. The trust settings in **/usr/share/pki/ca-trust-source/** are processed with lower priority than settings in **/etc/pki/ca-trust/**.

Certificate files are treated depending on the subdirectory they are installed to the following directories:

- for trust anchors
 - **/usr/share/pki/ca-trust-source/anchors/** or
 - **/etc/pki/ca-trust/source/anchors/**
- for distrusted certificates
 - **/usr/share/pki/ca-trust-source/blacklist/** or
 - **/etc/pki/ca-trust/source/blacklist/**
- for certificates in the extended BEGIN TRUSTED file format
 - **/usr/share/pki/ca-trust-source/** or
 - **/etc/pki/ca-trust/source/**



NOTE

In a hierarchical cryptographic system, a trust anchor is an authoritative entity which is assumed to be trustworthy. For example, in X.509 architecture, a root certificate is a trust anchor from which a chain of trust is derived. The trust anchor must be put in the possession of the trusting party beforehand to make path validation possible.

33.2. ADDING NEW CERTIFICATES

1. To add a certificate in the simple PEM or DER file formats to the list of CAs trusted on the system, copy the certificate file to the **/usr/share/pki/ca-trust-source/anchors/** or **/etc/pki/ca-trust/source/anchors/** directory, for example:

```
# cp ~/certificate-trust-examples/Cert-trust-test-ca.pem /usr/share/pki/ca-trust-source/anchors/
```

2. To update the system-wide trust store configuration, use the **update-ca-trust** command:

```
# update-ca-trust
```



NOTE

While the Firefox browser is able to use an added certificate without executing **update-ca-trust**, it is recommended to run **update-ca-trust** after a CA change. Also note that browsers, such as Firefox, Epiphany, or Chromium, cache files, and you might need to clear the browser's cache or restart your browser to load the current system certificates configuration.

33.3. MANAGING TRUSTED SYSTEM CERTIFICATES

- To list, extract, add, remove, or change trust anchors, use the **trust** command. To see the built-in help for this command, enter it without any arguments or with the **--help** directive:

```
$ trust
usage: trust command <args>...
```

Common trust commands are:

list	List trust or certificates
extract	Extract certificates and trust
extract-compat	Extract trust compatibility bundles
anchor	Add, remove, change trust anchors
dump	Dump trust objects in internal format

See 'trust <command> --help' for more information

- To list all system trust anchors and certificates, use the **trust list** command:

```
$ trust list
pkcs11:id=%d2%87%b4%e3%df%37%27%93%55%f6%56%ea%81%e5%36%cc%8c%1e%3
f%bd;type=cert
  type: certificate
  label: ACCVRAIZ1
  trust: anchor
  category: authority

pkcs11:id=%a6%b3%e1%2b%2b%49%b6%d7%73%a1%aa%94%f5%01%e7%73%65%4c%
ac%50;type=cert
  type: certificate
  label: ACEDICOM Root
  trust: anchor
  category: authority
...
[trimmed for clarity]
```

- To store a trust anchor into the system-wide trust store, use the **trust anchor** sub-command and specify a path to a certificate. Replace *path.to/certificate.crt* by a path to your certificate and its file name:

```
# trust anchor path.to/certificate.crt
```

- To remove a certificate, use either a path to a certificate or an ID of a certificate:

```
# trust anchor --remove path.to/certificate.crt
# trust anchor --remove "pkcs11:id=%AA%BB%CC%DD%EE;type=cert"
```

Additional resources

All sub-commands of the **trust** commands offer a detailed built-in help, for example:

```
$ trust list --help
usage: trust list --filter=<what>

--filter=<what>    filter of what to export
                   ca-anchors    certificate anchors
                   blacklist     blacklisted certificates
                   trust-policy   anchors and blacklist (default)
                   certificates   all certificates
                   pkcs11:object=xx a PKCS#11 URI
--purpose=<usage>  limit to certificates usable for the purpose
                   server-auth   for authenticating servers
                   client-auth   for authenticating clients
                   email         for email protection
                   code-signing  for authenticating signed code
                   1.2.3.4.5...  an arbitrary object id
-v, --verbose     show verbose debug output
-q, --quiet       suppress command output
```

33.4. RELATED INFORMATION

For more information, see the following man pages:

- **update-ca-trust(8)**
- **trust(1)**

CHAPTER 34. SCANNING THE SYSTEM FOR SECURITY COMPLIANCE AND VULNERABILITIES

34.1. SECURITY COMPLIANCE TOOLS IN RHEL

Red Hat Enterprise Linux provides tools that allow for a fully automated compliance audit. These tools are based on the Security Content Automation Protocol (SCAP) standard and are designed for automated tailoring of compliance policies.

- **SCAP Workbench** - The **scap-workbench** graphical utility is designed to perform configuration and vulnerability scans on a single local or remote system. It can be also used to generate security reports based on these scans and evaluations.
- **OpenSCAP** - The **oscap** command-line utility is designed to perform configuration and vulnerability scans on a local system, to validate security compliance content, and to generate reports and guides based on these scans and evaluations.
- **SCAP Security Guide (SSG)** - The **scap-security-guide** package provides the latest collection of security policies for Linux systems. The guidance consists of a catalog of practical hardening advice, linked to government requirements where applicable. The project bridges the gap between generalized policy requirements and specific implementation guidelines.
- **Script Check Engine (SCE)** - SCE is an extension to the SCAP protocol that allows administrators to write their security content using a scripting language, such as Bash, Python, or Ruby. The SCE extension is provided in the **openscap-engine-sce** package.

If you require performing automated compliance audits on multiple systems remotely, you can utilize OpenSCAP solution for Red Hat Satellite.

Additional resources

- **oscap(8)** - The manual page for the **oscap** command-line utility provides a complete list of available options and their usage explanation.
- **scap-workbench(8)** - The manual page for the **SCAP Workbench** application provides a basic information about the application as well as some links to potential sources of SCAP content.
- **scap-security-guide(8)** - The manual page for the **scap-security-guide** project provides further documentation about the various available SCAP security profiles. Examples how to utilize the provided benchmarks using the OpenSCAP utility are provided as well.
- For more details about using OpenSCAP with Red Hat Satellite, see [Security Compliance Management in the Administering Red Hat Satellite Guide](#).

34.2. RED HAT SECURITY ADVISORIES OVAL FEED

Red Hat Enterprise Linux security auditing capabilities are based on the Security Content Automation Protocol (SCAP) standard. SCAP is a multi-purpose framework of specifications that supports automated configuration, vulnerability and patch checking, technical control compliance activities, and security measurement.

SCAP specifications create an ecosystem where the format of security content is well known and standardized while the implementation of the scanner or policy editor is not mandated. Such a status enables organizations to build their security policy (SCAP content) once, no matter how many security

vendors do they employ.

The Open Vulnerability Assessment Language (OVAL) is the essential and oldest component of SCAP. Unlike other tools or custom scripts, the OVAL language describes a required state of resources in a declarative manner. The OVAL language code is never executed directly but by means of an OVAL interpreter tool called scanner. The declarative nature of OVAL ensures that the state of the assessed system is not accidentally modified.

Like all other SCAP components, OVAL is based on XML. The SCAP standard defines several document formats. Each of them includes a different kind of information and serves a different purpose.

[Red Hat Product Security](#) helps customers evaluate and manage risk by tracking and investigating all security issues affecting Red Hat customers. It provides timely and concise patches and security advisories on the Red Hat Customer Portal. Red Hat creates and supports OVAL patch definitions, providing machine-readable versions of our security advisories.

The [RHSA OVAL definitions](#) are available individually and as a complete package, and are updated within an hour of a new security advisory being made available on the Red Hat Customer Portal.

Each OVAL patch definition maps one-to-one to a Red Hat Security Advisory (RHSA). Since an RHSA can contain fixes for multiple vulnerabilities, each vulnerability is listed separately by its Common Vulnerabilities and Exposures (CVE) name and has a link to its entry in our public bug database.

The RHSA OVAL definitions are designed to check for vulnerable versions of RPM packages installed on a system. It is possible to extend these definitions to include further checks - for instance, to find out if the packages are being used in a vulnerable configuration. These definitions are designed to cover software and updates shipped by Red Hat. Additional definitions are required to detect the patch status of third-party software.

Additional resources

- [Red Hat and OVAL compatibility](#)
- [Red Hat and CVE compatibility](#)
- [Notifications and Advisories](#) in the [Product Security Overview](#)
- [Security Data Metrics](#)

34.3. SCANNING THE SYSTEM FOR VULNERABILITIES

The **oscap** command-line utility enables users to scan local systems, validate security compliance content, and generate reports and guides based on these scans and evaluations. This utility serves as a front end to the OpenSCAP library and groups its functionalities to modules (sub-commands) based on the type of SCAP content it processes.

Prerequisites

- The **AppStream** repository is enabled.

Procedure

1. Install the **openscap-scanner** package:

```
# yum install openscap-scanner
```

2. Download the latest RHSA OVAL definitions for your system:

```
# wget https://www.redhat.com/security/data/oval/com.redhat.rhsa-RHEL8.xml
```

3. Scan the system for vulnerabilities and save results to the *vulnerability.html* file:

```
# oscap oval eval --report vulnerability.html com.redhat.rhsa-RHEL8.xml
```

Verification steps

1. Check the results in a browser of your choice, for example:

```
$ firefox vulnerability.html &
```

Additional resources

- The **oscap(8)** man page.
- The [Red Hat OVAL definitions](#) list.

34.4. SCANNING REMOTE SYSTEMS FOR VULNERABILITIES

You can check also remote systems for vulnerabilities with the OpenSCAP scanner. This functionality is enabled by the **oscap-ssh** tool over the SSH protocol.

Prerequisites

- The **AppStream** repository is enabled.
- The **openscap-scanner** package is installed on the remote systems.
- The SSH server is running on the remote systems.

Procedure

1. Install the **openscap-utils** package:

```
# yum install openscap-utils
```

2. Download the latest RHSA OVAL definitions for your system:

```
# wget https://www.redhat.com/security/data/oval/com.redhat.rhsa-RHEL8.xml
```

3. Scan a remote system with the *machine1* host name, SSH running on port 22, and the *joesec* user name for vulnerabilities and save results to the *remote-vulnerability.html* file:

```
# oscap-ssh joesec@machine1 22 oval eval --report remote-vulnerability.html com.redhat.rhsa-RHEL8.xml
```

Additional resources

- The **oscap-ssh(8)** man page.

- The [Red Hat OVAL definitions](#) list.

34.5. VIEWING PROFILES FOR SECURITY COMPLIANCE

RHEL 8 provides several profiles for compliance with security policies. Before you decide to use them for scanning or remediation, you can list them and check their detailed descriptions using the **oscap info** sub-command.

Prerequisites

- The **openscap-scanner** and **scap-security-guide** packages are installed.

Procedure

1. List all available files with security compliance profiles provided by the SCAP Security Guide project:

```
$ ls /usr/share/xml/scap/ssg/content/
ssg-firefox-cpe-dictionary.xml  ssg-rhel6-ocil.xml
ssg-firefox-cpe-oval.xml        ssg-rhel6-oval.xml
...
ssg-rhel6-ds-1.2.xml          ssg-rhel8-oval.xml
ssg-rhel8-ds.xml              ssg-rhel8-xccdf.xml
...
```

2. Display detailed information about a selected data stream using the **oscap info** sub-command. XML files containing data streams are indicated by the **-ds** string in their names. In the **Profiles** section, you can find a list of available profiles and their IDs:

```
$ oscap info /usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
...
Profiles:
  Title: PCI-DSS v3 Control Baseline for Red Hat Enterprise Linux 8
  Id: xccdf_org.ssgproject.content_profile_pci-dss
  Title: OSPP - Protection Profile for General Purpose Operating Systems
  Id: xccdf_org.ssgproject.content_profile_ospp
...
```

3. Select a profile from the data-stream file and display additional details about the selected profile. To do so, use **oscap info** with the **--profile** option followed by the last section of the ID displayed in the output of the previous command. For example, the ID of the PCI-DSS profile is: **xccdf_org.ssgproject.content_profile_pci-dss**, and the value for the **--profile** option is **pci-dss**:

```
$ oscap info --profile pci-dss /usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
...
Title: PCI-DSS v3.2.1 Control Baseline for Red Hat Enterprise Linux 8
Id: xccdf_org.ssgproject.content_profile_pci-dss

Description: Ensures PCI-DSS v3.2.1 security configuration settings are applied.
...
```

Additional resources

- The **scap-security-guide(8)** man page.

34.6. ASSESSING SECURITY COMPLIANCE WITH A SPECIFIC BASELINE

The **SCAP Security Guide** suite provides profiles for several platforms in a form of XCCDF, OVAL, and data stream documents. The *profile* is a set of rules based on a security policy, such as Operating System Protection Profile (OSPP) or Payment Card Industry Data Security Standard (PCI-DSS). This enables you to audit the system in an automated way in respect of security standards.

Prerequisites

- The **openscap-scanner** package is installed.

Procedure

1. Install the **scap-security-guide** packages:

```
# yum install scap-security-guide
```

2. Display detailed information about a selected data stream using the **oscap info** sub-command. In the **Profiles** section, you can find a list of available profiles and their IDs:

```
$ oscap info /usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

...

Profiles:

Title: PCI-DSS v3 Control Baseline for Red Hat Enterprise Linux 8
Id: xccdf_org.ssgproject.content_profile_pci-dss
Title: OSPP - Protection Profile for General Purpose Operating Systems
Id: xccdf_org.ssgproject.content_profile_ospp

...

[trimmed for clarity]

Select a profile from the data-stream file and display more details about the selected profile by providing the last part of an ID identified in the output of the previous command to the **--profile** option of **oscap info**. For example, the OSPP profile has Id: **xccdf_org.ssgproject.content_profile_ospp**, and the value for the **--profile** option is **ospp**:

```
$ oscap info --profile ospp /usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

The PCI-DSS v3 Control Baseline profile is identified by **xccdf_org.ssgproject.content_profile_pci-dss**:

```
$ oscap info --profile pci-dss /usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

...

Description: Ensures PCI-DSS v3 related security configuration settings are applied.
[trimmed for clarity]

Alternatively, when using GUI, install the **scap-security-guide-doc** package and open the <file:///usr/share/doc/scap-security-guide/guides/ssg-rhel8-guide-index.html> file in a web browser. After you select a required profile in the top right field of the *Guide to the Secure Configuration of Red Hat Enterprise Linux 8* document, you can see a relevant command for the following evaluation.

- Evaluate the compliance of the system with the selected profile and save scan results in the `report.html` HTML file, for example:

```
$ oscap xccdf eval --report report.html --profile ospp /usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

With the **openscap-utils** package installed on your system and the **openscap-scanner** package installed on a remote system, you can also scan the remote system with the `machine1` host name, SSH running on port 22, and the `joesec` user name for vulnerabilities and save results to the `remote-report.html` file:

```
$ oscap-ssh joesec@machine1 22 xccdf eval --report remote_report.html --profile ospp /usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

Additional resources

- **scap-security-guide(8)** man page
- The **SCAP Security Guide** documentation installed in the [file:///usr/share/doc/scap-security-guide/](#) directory.
- The [Guide to the Secure Configuration of Red Hat Enterprise Linux 8](#) installed with the **scap-security-guide-doc** package.

34.7. REMEDIATING THE SYSTEM TO ALIGN WITH OSPP

Use this procedure to remediate the RHEL 8 system to align with the Protection Profile for General Purpose Operating Systems (OSPP).



IMPORTANT

Red Hat does not provide any automated method to revert changes made by security-hardening remediations. Remediations are supported on RHEL systems in the default configuration. If your system has been altered after the installation, running remediation might not make it compliant with the required security profile.

Prerequisites

- The **scap-security-guide** package is installed on your RHEL 8 system.

Procedure

1. Use the **oscap** command with the **--remediate** option:

```
# oscap xccdf eval --profile ospp --remediate /usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

2. Restart your system.

Verification steps

1. Evaluate the system of how it complies with the OSPP profile, and save results to the `ospp_report.html` file:

```
$ oscap xccdf eval --report ospp_report.html --profile ospp
/usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

Additional resources

- **scap-security-guide(8)** and **oscap(8)** man pages

34.8. SCANNING THE SYSTEM WITH A CUSTOMIZED PROFILE USING SCAP WORKBENCH

SCAP Workbench (scap-workbench) is a graphical utility that enables users to perform configuration and vulnerability scans on a single local or a remote system, perform remediation of the system, and generate reports based on scan evaluations. Note that compared with the **oscap** command-line utility, **SCAP Workbench** has only limited functionality. **SCAP Workbench** can also process only security content in the form of XCCDF and data-stream files.

Prerequisites

- **SCAP Workbench** is installed on your system by the **yum install scap-workbench** command.

34.8.1. Using SCAP Workbench to scan and remediate the system

To evaluate your system against the selected security policy, use the following procedure.

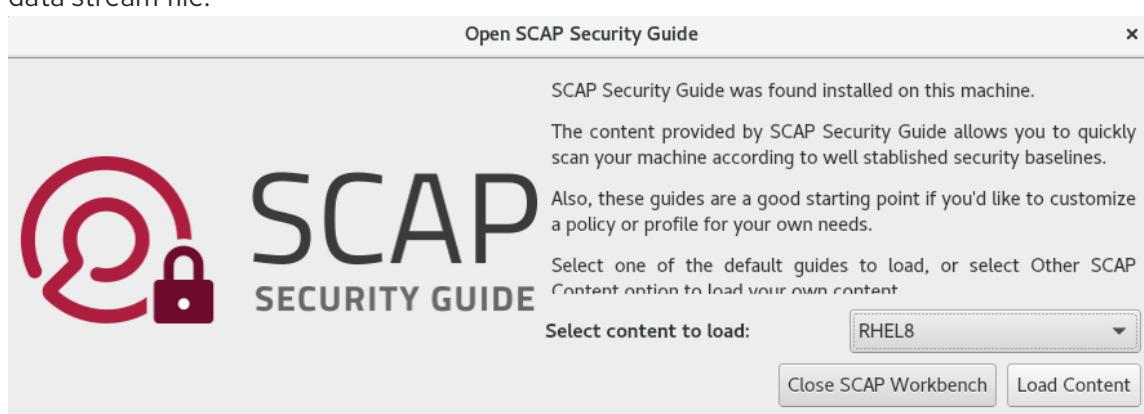
Procedure

1. To run **SCAP Workbench** from the **GNOME Classic** desktop environment, press the **Super** key to enter the **Activities Overview**, type **scap-workbench**, and then press **Enter**. Alternatively, use:

```
$ scap-workbench &
```

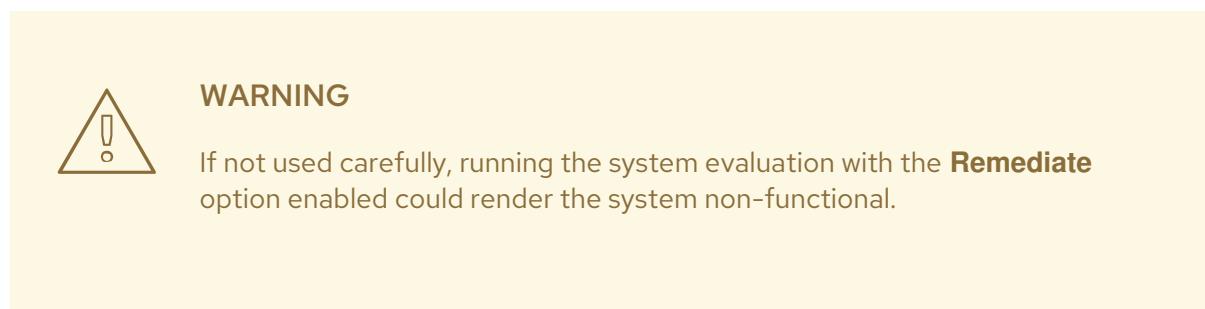
2. Select a security policy by using either the following options:

- **Load Content** button on the starting window
- **Open content from SCAP Security Guide**
- **Open Other Content** in the **File** menu, and search the respective XCCDF, SCAP RPM, or data stream file.



3. You can allow automatic correction of the system configuration by selecting the **Remediate**

check box. With this option enabled, **SCAP Workbench** attempts to change the system configuration in accordance with the security rules applied by the policy. This process should fix the related checks that fail during the system scan.



- Scan your system with the selected profile by clicking the **Scan** button.

The screenshot shows the SCAP Workbench interface with the following configuration:

- Title:** Guide to the Secure Configuration of Red Hat Enterprise Linux 8
- Customization:** /home/joesec/ssg-rhel8-ds-tailoring.xml
- Profile:** Protection Profile for General Purpose Operating Systems (151)
- Target:** Local Machine (radio button selected)

The **Rules** section displays a list of 151 rules, each with a status indicator (fail or pass) and a color-coded bar:

Rule Description	Status
Extend Audit Backlog Limit for the Audit Daemon	fail
Enable Auditing for Processes Which Start Prior to the Audit Daemon	fail
Enable auditd Service	pass
Configure SSSD to Expire Offline Credentials	pass
Configure SSSD's Memory Cache to Expire	pass
Disable SSH Root Login	fail
Disable SSH Access via Empty Passwords	fail
Disable Kerberos Authentication	pass
Disable Host-Based Authentication	pass
Disable SSH Support for Rhoids RSA Authentication	fail
Disable SSH Support for User Known Hosts	fail

At the bottom, there are buttons for **Clear**, **Save Results** (dropdown menu), **Generate remediation role** (dropdown menu), and **Show Report**.

- To store the scan results in form of an XCCDF, ARF, or HTML file, click the **Save Results** combo box. Choose the **HTML Report** option to generate the scan report in human-readable format. The XCCDF and ARF (data stream) formats are suitable for further automatic processing. You can repeatedly choose all three options.
- To export results-based remediations to a file, use the **Generate remediation role** pop-up menu.

34.8.2. Customizing a security profile with SCAP Workbench

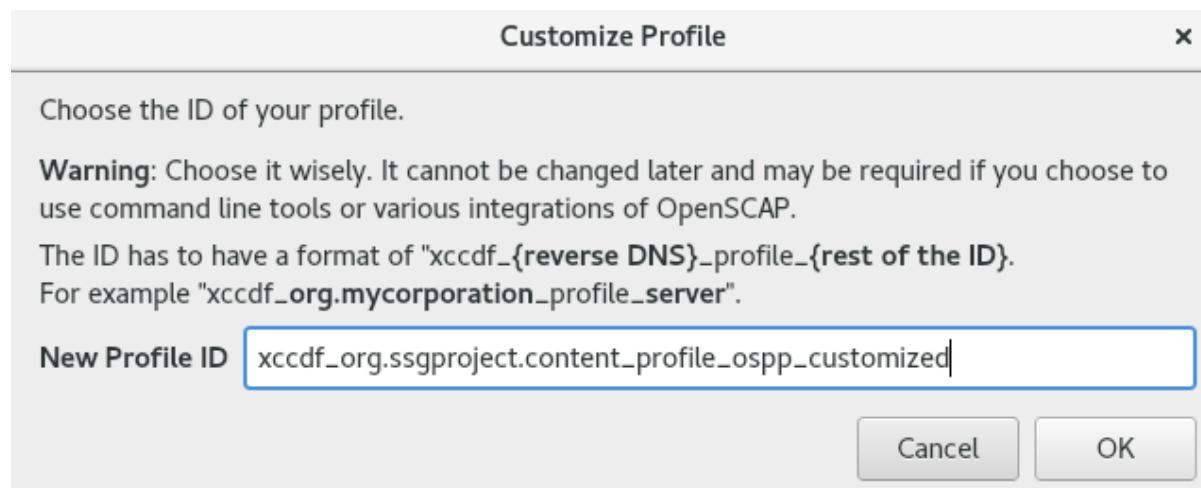
The following procedure demonstrates how to use **SCAP Workbench** to customize a profile. You can also save the customized profile for use with the **oscap** command-line utility.

Procedure

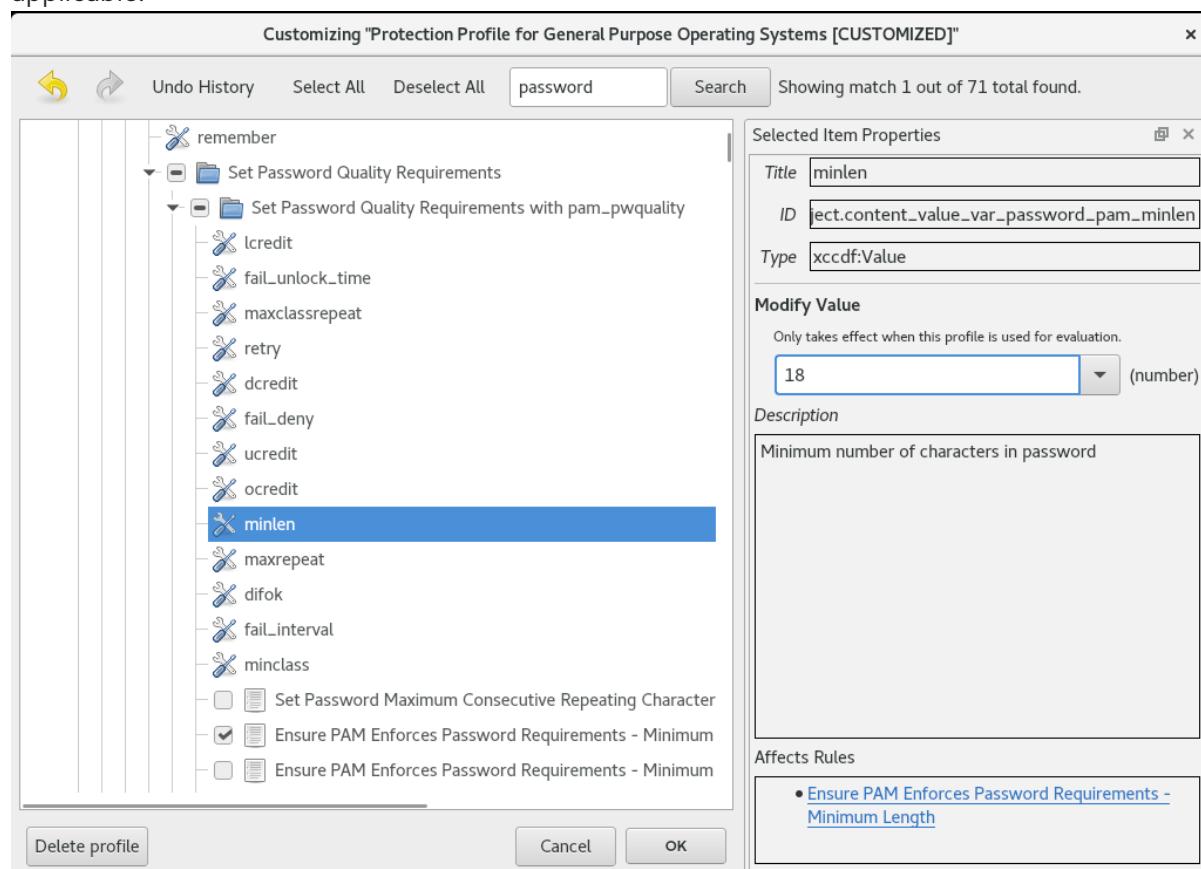
1. Run **SCAP Workbench**, and select the profile to customize by using either **Open content from SCAP Security Guide** or **Open Other Content** in the **File** menu.

2. To further adjust the selected security profile to make it stricter or looser according to your organization needs, click the **Customize** button.

This opens the new Customization window that enables you to modify the currently selected XCCDF profile without changing the respective XCCDF file. Choose the new profile ID.



3. Use either the tree structure with rules organized into logical groups or the Search field to find a rule to modify.
4. Include or exclude rules using check boxes in the tree structure, or modify values in rules where applicable.



5. Confirm the changes by clicking the **OK** button.

6. To store your changes permanently, use one of the following options:

- Save a customization file separately by using **Save Customization Only** in the **File** menu.
- Save all security content at once by **Save All** in the **File** menu.

By selecting the **Into a directory** option, **SCAP Workbench** saves both the XCCDF or data-stream file and the customization file to the specified location. This can be useful as a backup solution.

By selecting the **As RPM** option, you can instruct **SCAP Workbench** to create an RPM package containing the XCCDF or data stream file and customization file. This is useful for distributing the security content to systems that cannot be scanned remotely, or just for delivering the content for further processing.



NOTE

Because **SCAP Workbench** does not support results-based remediations for tailored profiles, use the exported remediations with the **oscap** command-line utility.

34.8.3. Related information

- **scap-workbench(8)** man page
- [SCAP Workbench User Manual](#)

CHAPTER 35. CHECKING INTEGRITY WITH AIDE

Advanced Intrusion Detection Environment (**AIDE**) is a utility that creates a database of files on the system, and then uses that database to ensure file integrity and detect system intrusions.

35.1. INSTALLING AIDE

The following steps are necessary to install **AIDE** and to initiate its database.

Prerequisites

- The **AppStream** repository is enabled.

Procedure

1. To install the *aide* package:

```
# yum install aide
```

2. To generate an initial database:

```
# aide --init
```



NOTE

In the default configuration, the **aide --init** command checks just a set of directories and files defined in the **/etc/aide.conf** file. To include additional directories or files in the **AIDE** database, and to change their watched parameters, edit **/etc/aide.conf** accordingly.

3. To start using the database, remove the **.new** substring from the initial database file name:

```
# mv /var/lib/aide/aide.db.new.gz /var/lib/aide/aide.db.gz
```

4. To change the location of the **AIDE** database, edit the **/etc/aide.conf** file and modify the **DBDIR** value. For additional security, store the database, configuration, and the **/usr/sbin/aide** binary file in a secure location such as a read-only media.

35.2. PERFORMING INTEGRITY CHECKS WITH AIDE

Prerequisites

- **AIDE** is properly installed and its database is initialized. See [Section 35.1, “Installing AIDE”](#)

Procedure

1. To initiate a manual check:

```
# aide --check
Start timestamp: 2018-07-11 12:41:20 +0200 (AIDE 0.16)
AIDE found differences between database and filesystem!!
```

...
[trimmed for clarity]

2. At a minimum, **AIDE** should be configured to run a weekly scan. At most, **AIDE** should be run daily. For example, to schedule a daily execution of **AIDE** at 04:05 a.m. using the **cron** command, add the following line to the **/etc/crontab** file:

```
05 4 * * * root /usr/sbin/aide --check
```

35.3. UPDATING AN AIDE DATABASE

After verifying the changes of your system such as, package updates or configuration files adjustments, updating your baseline **AIDE** database is recommended.

Prerequisites

- **AIDE** is properly installed and its database is initialized. See [Section 35.1, “Installing AIDE”](#)

Procedure

1. Update your baseline **AIDE** database:

```
# aide --update
```

The **aide --update** command creates the **/var/lib/aide/aide.db.new.gz** database file.

2. To start using the updated database for integrity checks, remove the **.new** substring from the file name.

35.4. RELATED INFORMATION

For additional information on **AIDE**, see the **aide(1)** man page.

CHAPTER 36. ENCRYPTING BLOCK DEVICES USING LUKS

Disk encryption protects the data on a block device by encrypting it. To access the device's decrypted contents, a user must provide a passphrase or key as authentication. This is particularly important when it comes to mobile computers and removable media: it helps to protect the device's contents even if it has been physically removed from the system. The LUKS format is a default implementation of block device encryption in RHEL.

36.1. LUKS DISK ENCRYPTION

The Linux Unified Key Setup-on-disk-format (LUKS) enables you to encrypt block devices and it provides a set of tools that simplifies managing the encrypted devices. LUKS allows multiple user keys to decrypt a master key, which is used for the bulk encryption of the partition.

RHEL utilizes LUKS to perform block device encryption. By default, the option to encrypt the block device is unchecked during the installation. If you select the option to encrypt your disk, the system prompts you for a passphrase every time you boot the computer. This passphrase "unlocks" the bulk encryption key that decrypts your partition. If you choose to modify the default partition table, you can choose which partitions you want to encrypt. This is set in the partition table settings.

What LUKS does

- LUKS encrypts entire block devices and is therefore well-suited for protecting contents of mobile devices such as removable storage media or laptop disk drives.
- The underlying contents of the encrypted block device are arbitrary, which makes it useful for encrypting swap devices. This can also be useful with certain databases that use specially formatted block devices for data storage.
- LUKS uses the existing device mapper kernel subsystem.
- LUKS provides passphrase strengthening which protects against dictionary attacks.
- LUKS devices contain multiple key slots, allowing users to add backup keys or passphrases.

What LUKS does not do

- Disk-encryption solutions like LUKS protect the data only when your system is off. Once the system is on and LUKS has decrypted the disk, the files on that disk are available to anyone who would normally have access to them.
- LUKS is not well-suited for scenarios that require many users to have distinct access keys to the same device. The LUKS1 format provides eight key slots, LUKS2 up to 32 key slots.
- LUKS is not well-suited for applications requiring file-level encryption.

Ciphers

The default cipher used for LUKS is **aes-xts-plain64**. The default key size for LUKS is 512 bits. The default key size for LUKS with **Anaconda** (XTS mode) is 512 bits. Ciphers that are available are:

- AES - Advanced Encryption Standard - [FIPS PUB 197](#)
- Twofish (a 128-bit block cipher)
- Serpent

Additional resources

- [LUKS Project Home Page](#)
- [LUKS On-Disk Format Specification](#)

36.2. LUKS VERSIONS IN RHEL 8

In RHEL 8, the default format for LUKS encryption is LUKS2. The legacy LUKS1 format remains fully supported and it is provided as a format compatible with earlier RHEL releases.

The LUKS2 format is designed to enable future updates of various parts without a need to modify binary structures. LUKS2 internally uses JSON text format for metadata, provides redundancy of metadata, detects metadata corruption and allows automatic repairs from a metadata copy.



IMPORTANT

Do not use LUKS2 in systems that need to be compatible with legacy systems that support only LUKS1. Note that RHEL 7 supports the LUKS2 format since version 7.6.



WARNING

LUKS2 and LUKS1 use different commands to encrypt the disk. Using the wrong command for a LUKS version might cause data loss.

LUKS version	Encryption command
LUKS2	cryptsetup reencrypt
LUKS1	cryptsetup-reencrypt

Online re-encryption

The LUKS2 format supports re-encrypting encrypted devices while the devices are in use. For example, you do not have to unmount the file system on the device to perform the following tasks:

- Change the volume key
- Change the encryption algorithm

When encrypting a non-encrypted device, you must still unmount the file system. You can remount the file system after a short initialization of the encryption.

The LUKS1 format does not support online re-encryption.

Conversion

The LUKS2 format is inspired by LUKS1. In certain situations, you can convert LUKS1 to LUKS2. The conversion is not possible specifically in the following scenarios:

- A LUKS1 device is marked as being used by a Policy-Based Decryption (PBD - Clevis) solution. The **cryptsetup** tool refuses to convert the device when some **luksmeta** metadata are detected.
- A device is active. The device must be in the inactive state before any conversion is possible.

36.3. OPTIONS FOR DATA PROTECTION DURING LUKS2 RE-ENCRYPTION

LUKS2 provides several options that prioritize performance or data protection during the re-encryption process:

checksum

This is the default mode. It balances data protection and performance.

This mode stores individual checksums of the sectors in the re-encryption area, so the recovery process can detect which sectors LUKS2 already re-encrypted. The mode requires that the block device sector write is atomic.

journal

That is the safest mode but also the slowest. This mode journals the re-encryption area in the binary area, so LUKS2 writes the data twice.

none

This mode prioritizes performance and provides no data protection. It protects the data only against safe process termination, such as the **SIGTERM** signal or the user pressing **Ctrl+C**. Any unexpected system crash or application crash might result in data corruption.

You can select the mode using the **--resilience** option of **cryptsetup**.

If a LUKS2 re-encryption process terminates unexpectedly by force, LUKS2 can perform the recovery in one of the following ways:

- Automatically, during the next LUKS2 device open action. This action is triggered either by the **cryptsetup open** command or by attaching the device with **systemd-cryptsetup**.
- Manually, by using the **cryptsetup repair** command on the LUKS2 device.

36.4. ENCRYPTING EXISTING DATA ON A BLOCK DEVICE USING LUKS2

This procedure encrypts existing data on a not yet encrypted device using the LUKS2 format. A new LUKS header is stored in the head of the device.

Prerequisites

- The block device contains a file system.
- You have backed up your data.



WARNING

You might lose your data during the encryption process: due to a hardware, kernel, or human failure. Ensure that you have a reliable backup before you start encrypting the data.

Procedure

1. Unmount all file systems on the device that you plan to encrypt. For example:

```
# umount /dev/sdb1
```

2. Make free space for storing a LUKS header. Choose one of the following options that suits your scenario:

- In the case of encrypting a logical volume, you can extend the logical volume without resizing the file system. For example:

```
# lvextend -L+32M vg00/lv00
```

- Extend the partition using partition management tools, such as **parted**.
- Shrink the file system on the device. You can use the **resize2fs** utility for the ext2, ext3, or ext4 file systems. Note that you cannot shrink the XFS file system.

3. Initialize the encryption. For example:

```
# cryptsetup reencrypt \
    --encrypt \
    --init-only \
    --reduce-device-size 32M \
    /dev/sdb1 sdb1_encrypted
```

The command asks you for a passphrase and starts the encryption process.

4. Mount the device:

```
# mount /dev/mapper/sdb1_encrypted /mnt/sdb1_encrypted
```

5. Start the online encryption:

```
# cryptsetup reencrypt --resume-only /dev/sdb1
```

Additional resources

- For more details, see the **cryptsetup(8)**, **lvextend(8)**, **resize2fs(8)**, and **parted(8)** man pages.

36.5. ENCRYPTING EXISTING DATA ON A BLOCK DEVICE USING LUKS2 WITH A DETACHED HEADER

This procedure encrypts existing data on a block device without creating free space for storing a LUKS header. The header is stored in a detached location, which also serves as an additional layer of security. The procedure uses the LUKS2 encryption format.

Prerequisites

- The block device contains a file system.
- You have backed up your data.



WARNING

You might lose your data during the encryption process: due to a hardware, kernel, or human failure. Ensure that you have a reliable backup before you start encrypting the data.

Procedure

1. Unmount all file systems on the device. For example:

```
# umount /dev/sdb1
```

2. Initialize the encryption:

```
# cryptsetup reencrypt \
--encrypt \
--init-only \
--header /path/to/header \
/dev/sdb1 sdb1_encrypted
```

Replace */path/to/header* with a path to the file with a detached LUKS header. The detached LUKS header has to be accessible so that the encrypted device can be unlocked later.

The command asks you for a passphrase and starts the encryption process.

3. Mount the device:

```
# mount /dev/mapper/sdb1_encrypted /mnt/sdb1_encrypted
```

4. Start the online encryption:

```
# cryptsetup reencrypt --resume-only --header /path/to/header /dev/sdb1
```

Additional resources

- For more details, see the **cryptsetup(8)** man page.

CHAPTER 37. CONFIGURING AUTOMATED UNLOCKING OF ENCRYPTED VOLUMES USING POLICY-BASED DECRYPTION

The Policy-Based Decryption (PBD) is a collection of technologies that enable unlocking encrypted root and secondary volumes of hard drives on physical and virtual machines. PBD uses a variety of unlocking methods, such as user passwords, a Trusted Platform Module (TPM) device, a PKCS #11 device connected to a system, for example, a smart card, or a special network server.

PBD allows combining different unlocking methods into a policy, which makes it possible to unlock the same volume in different ways. The current implementation of the PBD in Red Hat Enterprise Linux consists of the Clevis framework and plug-ins called *pins*. Each pin provides a separate unlocking capability. Currently, the following pins are available:

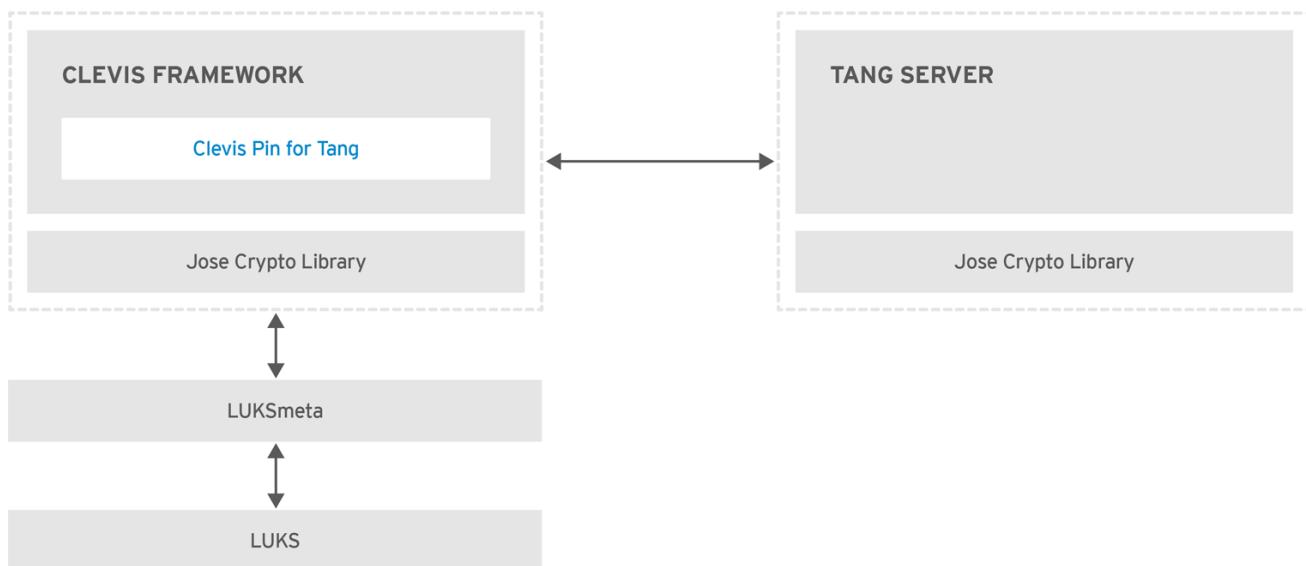
- **tang** – allows volumes to be unlocked using a network server
- **tpm2** – allows volumes to be unlocked using a TPM2 policy

The Network Bound Disc Encryption (NBDE) is a subcategory of PBD that allows binding encrypted volumes to a special network server. The current implementation of the NBDE includes a Clevis pin for Tang server and the Tang server itself.

37.1. NETWORK-BOUND DISK ENCRYPTION

In Red Hat Enterprise Linux, NBDE is implemented through the following components and technologies:

Figure 37.1. NBDE scheme when using a LUKS1-encrypted volume. The `luksmeta` package is not used for LUKS2 volumes.



RHEL_453350_0717

Tang is a server for binding data to network presence. It makes a system containing your data available when the system is bound to a certain secure network. Tang is stateless and does not require TLS or authentication. Unlike escrow-based solutions, where the server stores all encryption keys and has knowledge of every key ever used, Tang never interacts with any client keys, so it never gains any identifying information from the client.

Clevis is a pluggable framework for automated decryption. In NBDE, Clevis provides automated unlocking of LUKS volumes. The `clevis` package provides the client side of the feature.

A *Clevis pin* is a plug-in into the Clevis framework. One of such pins is a plug-in that implements interactions with the NBDE server – Tang.

Clevis and Tang are generic client and server components that provide network-bound encryption. In Red Hat Enterprise Linux, they are used in conjunction with LUKS to encrypt and decrypt root and non-root storage volumes to accomplish Network-Bound Disk Encryption.

Both client- and server-side components use the *José* library to perform encryption and decryption operations.

When you begin provisioning NBDE, the Clevis pin for Tang server gets a list of the Tang server’s advertised asymmetric keys. Alternatively, since the keys are asymmetric, a list of Tang’s public keys can be distributed out of band so that clients can operate without access to the Tang server. This mode is called *offline provisioning*.

The Clevis pin for Tang uses one of the public keys to generate a unique, cryptographically-strong encryption key. Once the data is encrypted using this key, the key is discarded. The Clevis client should store the state produced by this provisioning operation in a convenient location. This process of encrypting data is the *provisioning step*.

The LUKS version 2 (LUKS2) is the default format in Red Hat Enterprise Linux 8, hence, the provisioning state for NBDE is stored as a token in a LUKS2 header. The leveraging of provisioning state for NBDE by the **luksmeta** package is used only for volumes encrypted with LUKS1. The Clevis pin for Tang supports both LUKS1 and LUKS2 without specification need.

When the client is ready to access its data, it loads the metadata produced in the provisioning step and it responds to recover the encryption key. This process is the *recovery step*.

In NBDE, Clevis binds a LUKS volume using a pin so that it can be automatically unlocked. After successful completion of the binding process, the disk can be unlocked using the provided Dracut unlocker.

37.2. INSTALLING AN ENCRYPTION CLIENT - CLEVIS

Use this procedure to deploy and start using the Clevis pluggable framework on your system.

Procedure

1. To install Clevis and its pins on a system with an encrypted volume:

```
# yum install clevis
```

2. To decrypt data, use a **clevis decrypt** command and provide a cipher text in the JSON Web Encryption (JWE) format, for example:

```
$ clevis decrypt < secret.jwe
```

Additional resources

- For a quick reference, see the built-in CLI help:

```
$ clevis
Usage: clevis COMMAND [OPTIONS]
```

clevis decrypt	Decrypts using the policy defined at encryption time
----------------	--

clevis encrypt sss	Encrypts using a Shamir's Secret Sharing policy
clevis encrypt tang	Encrypts using a Tang binding server policy
clevis encrypt tpm2	Encrypts using a TPM2.0 chip binding policy
clevis luks bind	Binds a LUKS device using the specified policy
clevis luks list	Lists pins bound to a LUKSv1 or LUKSv2 device
clevis luks pass	Returns the LUKS passphrase used for binding a particular slot.
clevis luks regen	Regenerate LUKS metadata
clevis luks report	Report any key rotation on the server side
clevis luks unbind	Unbinds a pin bound to a LUKS volume
clevis luks unlock	Unlocks a LUKS volume

- For more information, see the **clevis(1)** man page.

37.3. DEPLOYING A TANG SERVER WITH SELINUX IN ENFORCING MODE

Use this procedure to deploy a Tang server running on a custom port as a confined service in SELinux enforcing mode.

Prerequisites

- The **policycoreutils-python-utils** package and its dependencies are installed.

Procedure

- To install the **tang** package and its dependencies, enter the following command as **root**:

```
# yum install tang
```

- Pick an unoccupied port, for example, `7500/tcp`, and allow the **tangd** service to bind to that port:

```
# semanage port -a -t tangd_port_t -p tcp 7500
```

Note that a port can be used only by one service at a time, and thus an attempt to use an already occupied port implies the **ValueError: Port already defined** error message.

- Open the port in the firewall:

```
# firewall-cmd --add-port=7500/tcp
# firewall-cmd --runtime-to-permanent
```

- Enable the **tangd** service:

```
# systemctl enable tangd.socket
```

- Create an override file:

```
# systemctl edit tangd.socket
```

- In the following editor screen, which opens an empty **override.conf** file located in the `/etc/systemd/system/tangd.socket.d/` directory, change the default port for the Tang server from 80 to the previously picked number by adding the following lines:

```
[Socket]
ListenStream=
ListenStream=7500
```

Save the file and exit the editor.

7. Reload the changed configuration:

```
# systemctl daemon-reload
```

8. Check that your configuration is working:

```
# systemctl show tangd.socket -p Listen
Listen=[::]:7500 (Stream)
```

9. Start the **tangd** service:

```
# systemctl start tangd.socket
```

Because **tangd** uses the **systemd** socket activation mechanism, the server starts as soon as the first connection comes in. A new set of cryptographic keys is automatically generated at the first start. To perform cryptographic operations such as manual key generation, use the **jose** utility.

Additional resources

- **tang(8)** man page
- **semanage(8)** man page
- **firewall-cmd(1)** man page
- **systemd.unit(5)** and **systemd.socket(5)** man pages
- **jose(1)** man page

37.4. ROTATING TANG SERVER KEYS AND UPDATING BINDINGS ON CLIENTS

Use the following steps to rotate your Tang server keys and update existing bindings on clients. The precise interval at which you should rotate them depends on your application, key sizes, and institutional policy.

Prerequisites

- A Tang server is running.
- The **clevis** and **clevis-luks** packages are installed on your clients.
- Note that **clevis luks list**, **clevis luks report**, and **clevis luks regen** have been introduced in RHEL 8.2.

Procedure

1. To rotate keys, generate new keys using the **/usr/libexec/tangd-keygen** command in the **/var/db/tang** key database directory on the Tang server:

```
# ls /var/db/tang
UV6dqXSwe1bRKG3KbJmdiR020hY.jwk y9hxLTQSiSB5jSEGWnjhY8fDTJU.jwk
# /usr/libexec/tangd-keygen /var/db/tang
# ls /var/db/tang
UV6dqXSwe1bRKG3KbJmdiR020hY.jwk y9hxLTQSiSB5jSEGWnjhY8fDTJU.jwk
3ZWS6-cDrCG61UPJS2BMmPU4I54.jwk zyLuX6hijUy_PSeUEFDi7hi38.jwk
```

2. Check that your Tang server advertises the signing key from the new key pair, for example:

```
# tang-show-keys 7500
3ZWS6-cDrCG61UPJS2BMmPU4I54
```

3. Rename the old keys to have a leading **.** to hide them from advertisement. Note that the file names in the following example differs from unique file names in the key database directory of your Tang server:

```
# cd /var/db/tang
# ls -l
-rw-r--r--. 1 root tang 354 Sep 23 16:08 3ZWS6-cDrCG61UPJS2BMmPU4I54.jwk
-rw-r--r--. 1 root tang 349 Sep 23 16:08 .I-zyLuX6hijUy_PSeUEFDi7hi38.jwk
-rw-r--r--. 1 root root 349 Feb 7 14:55 UV6dqXSwe1bRKG3KbJmdiR020hY.jwk
-rw-r--r--. 1 root root 354 Feb 7 14:55 y9hxLTQSiSB5jSEGWnjhY8fDTJU.jwk
# mv UV6dqXSwe1bRKG3KbJmdiR020hY.jwk .UV6dqXSwe1bRKG3KbJmdiR020hY.jwk
# mv y9hxLTQSiSB5jSEGWnjhY8fDTJU.jwk .y9hxLTQSiSB5jSEGWnjhY8fDTJU.jwk
```

Tang immediately picks up all changes. No restart is required. At this point, new client bindings pick up the new keys and old clients can continue to utilize the old keys.

4. On your NBDE clients, use the **clevis luks report** command to check if the keys advertised by the Tang server remains the same. You can identify slots with the relevant binding using the **clevis luks list** command, for example:

```
# clevis luks list -d /dev/sda2
1: tang '{"url":"http://tang.srv"}'
# clevis luks report -d /dev/sda2 -s 1
...
Report detected that some keys were rotated.
Do you want to regenerate luks metadata with "clevis luks regen -d /dev/sda2 -s 1"? [ynYN]
```

5. To regenerate LUKS metadata for the new keys either press **y** to the prompt of the previous command, or use the **clevis luks regen** command:

```
# clevis luks regen -d /dev/sda2 -s 1
```

6. When you are sure that all old clients use the new keys, you can remove the old keys from the Tang server, for example:

```
# cd /var/db/tang
# rm *.jwk
```

**WARNING**

Removing the old keys while clients are still using them can result in data loss. If you accidentally remove such keys, use the **clevis luks regen** command on the clients, and provide your LUKS password manually.

Additional resources

- **tang-show-keys(1)**, **clevis-luks-list(1)**, **clevis-luks-report(1)**, and **clevis-luks-regen(1)** man pages

37.5. DEPLOYING AN ENCRYPTION CLIENT FOR AN NBDE SYSTEM WITH TANG

The following procedure contains steps to configure automated unlocking of an encrypted volume with a Tang network server.

Prerequisites

- The Clevis framework is installed.
- A Tang server is available.

Procedure

1. To bind a Clevis encryption client to a Tang server, use the **clevis encrypt tang** sub-command:

```
$ clevis encrypt tang '{"url":"http://tang.srv:port"}' < input-plain.txt > secret.jwe
The advertisement contains the following signing keys:
_OsIk0T-E2l6qjfdDiwVmidoZjA
Do you wish to trust these keys? [ynYN] y
```

Change the *http://tang.srv:port* URL in the previous example to match the URL of the server where **tang** is installed. The *secret.jwe* output file contains your encrypted cipher text in the JSON Web Encryption format. This cipher text is read from the *input-plain.txt* input file.

Alternatively, if your configuration requires a non-interactive communication with a Tang server without SSH access, you can download an advertisement and save it to a file:

```
$ curl -sfg http://tang.srv:port/adv -o adv.jws
```

Use the advertisement in the *adv.jws* file for any following tasks, such as encryption of files or messages:

```
$ echo 'hello' | clevis encrypt tang '{"url":"http://tang.srv:port", "adv":"adv.jws"}'
```

2. To decrypt data, use the **clevis decrypt** command and provide the cipher text (JWE):

```
$ clevis decrypt < secret.jwe > output-plain.txt
```

Additional resources

- For a quick reference, see the **clevis-encrypt-tang(1)** man page or use the built-in CLI help:

```
$ clevis
$ clevis decrypt
$ clevis encrypt tang
Usage: clevis encrypt tang CONFIG < PLAINTEXT > JWE
...
...
```

- For more information, see the following man pages:
 - clevis(1)**
 - clevis-luks-unlockers(7)**

37.6. REMOVING A CLEVIS PIN FROM A LUKS-ENCRYPTED VOLUME MANUALLY

Use the following procedure for manual removing the metadata created by the **clevis luks bind** command and also for wiping a key slot that contains passphrase added by Clevis.



IMPORTANT

The recommended way to remove a Clevis pin from a LUKS-encrypted volume is through the **clevis luks unbind** command. The removal procedure using **clevis luks unbind** consists of only one step and works for both LUKS1 and LUKS2 volumes. The following example command removes the metadata created by the binding step and wipe the key slot 1 on the `/dev/sda2` device:

```
# clevis luks unbind -d /dev/sda2 -s 1
```

Prerequisites

- A LUKS-encrypted volume with a Clevis binding.

Procedure

- Check which LUKS version the volume, for example `/dev/sda2`, is encrypted by and identify a slot and a token that is bound to Clevis:

```
# cryptsetup luksDump /dev/sda2
LUKS header information
Version:      2
...
Keyslots:
  0: luks2
...
  1: luks2
    Key:      512 bits
    Priority: normal
```

```

Cipher: aes-xts-plain64
...
Tokens:
0: clevis
  Keyslot: 1
...

```

In the previous example, the Clevis token is identified by 0 and the associated key slot is 1.

2. In case of LUKS2 encryption, remove the token:

```
# cryptsetup token remove --token-id 0 /dev/sda2
```

3. If your device is encrypted by LUKS1, which is indicated by the **Version: 1** string in the output of the **cryptsetup luksDump** command, perform this additional step with the **luksmeta wipe** command:

```
# luksmeta wipe -d /dev/sda2 -s 1
```

4. Wipe the key slot containing the Clevis passphrase:

```
# cryptsetup luksKillSlot /dev/sda2 1
```

Additional resources

- For more information, see the **clevis-luks-unbind(1)**, **cryptsetup(8)**, and **luksmeta(8)** man pages.

37.7. DEPLOYING AN ENCRYPTION CLIENT WITH A TPM 2.0 POLICY

The following procedure contains steps to configure automated unlocking of an encrypted volume with a Trusted Platform Module 2.0 (TPM 2.0) policy.

Prerequisites

- The Clevis framework is installed. See [Installing an encryption client - Clevis](#)
- A system with the 64-bit Intel or 64-bit AMD architecture

Procedure

1. To deploy a client that encrypts using a TPM 2.0 chip, use the **clevis encrypt tpm2** sub-command with the only argument in form of the JSON configuration object:

```
$ clevis encrypt tpm2 '{}' < input-plain.txt > secret.jwe
```

To choose a different hierarchy, hash, and key algorithms, specify configuration properties, for example:

```
$ clevis encrypt tpm2 '{"hash":"sha1","key":"rsa"}' < input-plain.txt > secret.jwe
```

2. To decrypt the data, provide the ciphertext in the JSON Web Encryption (JWE) format:

```
$ clevis decrypt < secret.jwe > output-plain.txt
```

The pin also supports sealing data to a Platform Configuration Registers (PCR) state. That way, the data can only be unsealed if the PCRs hashes values match the policy used when sealing.

For example, to seal the data to the PCR with index 0 and 1 for the SHA-1 bank:

```
$ clevis encrypt tpm2 '{"pcr_bank":"sha1","pcr_ids":"0,1"}' < input-plain.txt > secret.jwe
```

Additional resources

- For more information and the list of possible configuration properties, see the **clevis-encrypt-tpm2(1)** man page.

37.8. CONFIGURING MANUAL ENROLLMENT OF LUKS-ENCRYPTED VOLUMES

Use the following steps to configure unlocking of LUKS-encrypted volumes with NBDE.

Prerequisite

- A Tang server is running and available.

Procedure

- To automatically unlock an existing LUKS-encrypted volume, install the **clevis-luks** subpackage:

```
# yum install clevis-luks
```

- Identify the LUKS-encrypted volume for PBD. In the following example, the block device is referred as `/dev/sda2`:

```
# lsblk
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda       8:0    0 12G  0 disk
└─sda1    8:1    0  1G  0 part /boot
└─sda2    8:2    0 11G  0 part
  └─luks-40e20552-2ade-4954-9d56-565aa7994fb6 253:0 0 11G 0 crypt
    ├─rhel-root    253:0 0 9.8G 0 lvm /
    └─rhel-swap    253:1 0 1.2G 0 lvm [SWAP]
```

- Bind the volume to a Tang server using the **clevis luks bind** command:

```
# clevis luks bind -d /dev/sda2 tang '{"url":"http://tang.srv"}'
```

The advertisement contains the following signing keys:

```
_Oslk0T-E2l6qjfdDiwVmidoZjA
```

Do you wish to trust these keys? [ynYN] y
 You are about to initialize a LUKS device for metadata storage.
 Attempting to initialize it may result in data loss if data was
 already written into the LUKS header gap in a different format.

A backup is advised before initialization is performed.

```
Do you wish to initialize /dev/sda2? [yn] y
Enter existing LUKS password:
```

This command performs four steps:

- a. Creates a new key with the same entropy as the LUKS master key.
- b. Encrypts the new key with Clevis.
- c. Stores the Clevis JWE object in the LUKS2 header token or uses LUKSMeta if the non-default LUKS1 header is used.
- d. Enables the new key for use with LUKS.



NOTE

The binding procedure assumes that there is at least one free LUKS password slot. The **clevis luks bind** command takes one of the slots.

4. The volume can now be unlocked with your existing password as well as with the Clevis policy.
5. To enable the early boot system to process the disk binding, enter the following commands on an already installed system:

```
# yum install clevis-dracut
# dracut -fv --regenerate-all
```

Verification steps

1. To verify that the Clevis JWE object is successfully placed in a LUKS header, use the **clevis luks list** command:

```
# clevis luks list -d /dev/sda2
1: tang '{"url":"http://tang.srv:port"}'
```



IMPORTANT

To use NBDE for clients with static IP configuration (without DHCP), pass your network configuration to the dracut tool manually, for example:

```
# dracut -fv --regenerate-all --kernel-cmdline
"ip=192.0.2.10::192.0.2.1:255.255.255.0::ens3:none:192.0.2.45"
```

Alternatively, create a .conf file in the **/etc/dracut.conf.d/** directory with the static network information. For example:

```
# cat /etc/dracut.conf.d/static_ip.conf
kernel_cmdline="ip=192.0.2.10::192.0.2.1:255.255.255.0::ens3:none:192.0.2.45"
```

Regenerate the initial RAM disk image:

```
# dracut -fv --regenerate-all
```

Additional resources

For more information, see the following man pages:

- **clevis-luks-bind(1)**
- **dracut.cmdline(7)**

37.9. CONFIGURING AUTOMATED ENROLLMENT OF LUKS-ENCRYPTED VOLUMES USING KICKSTART

Follow the steps in this procedure to configure an automated installation process that uses Clevis for enrollment of LUKS-encrypted volumes.

Procedure

1. Instruct Kickstart to partition the disk such that the root partition has enabled LUKS encryption with a temporary password. The password is temporary for the enrollment process.

```
part /boot --fstype="xfs" --ondisk=vda --size=256
part / --fstype="xfs" --ondisk=vda --grow --encrypted --passphrase=temppass
```

2. Install the related Clevis packages by listing them in the **%packages** section:

```
%packages
clevis-dracut
%end
```

3. Call **clevis luks bind** to perform binding in the **%post** section. Afterward, remove the temporary password:

```
%post
curl -sfg http://tang.srv/adv -o adv.jws
clevis luks bind -f -k- -d /dev/vda2 \
```

```
tang '{"url":"http://tang.srv","adv":"adv.jws"}' \ <<< "temppass"
cryptsetup luksRemoveKey /dev/vda2 <<< "temppass"
%end
```

In the previous example, note that we download the advertisement from the Tang server as part of our binding configuration, enabling binding to be completely non-interactive.



WARNING

The **cryptsetup luksRemoveKey** command prevents any further administration of a LUKS2 device on which you apply it. You can recover a removed master key using the **dmsetup** command only for LUKS1 devices.

You can use an analogous procedure when using a TPM 2.0 policy instead of a Tang server.

Additional resources

- **clevis(1)**, **clevis-luks-bind(1)**, **cryptsetup(8)**, and **dmsetup(8)** man pages
- [Installing Red Hat Enterprise Linux 8 using Kickstart](#)

37.10. CONFIGURING AUTOMATED UNLOCKING OF A LUKS-ENCRYPTED REMOVABLE STORAGE DEVICE

Use this procedure to set up an automated unlocking process of a LUKS-encrypted USB storage device.

Procedure

1. To automatically unlock a LUKS-encrypted removable storage device, such as a USB drive, install the **clevis-udisks2** package:

```
# yum install clevis-udisks2
```

2. Reboot the system, and then perform the binding step using the **clevis luks bind** command as described in *Configuring manual enrollment of LUKS-encrypted volumes*, for example:

```
# clevis luks bind -d /dev/sdb1 tang '{"url":"http://tang.srv"}'
```

3. The LUKS-encrypted removable device can be now unlocked automatically in your GNOME desktop session. The device bound to a Clevis policy can be also unlocked by the **clevis luks unlock** command:

```
# clevis luks unlock -d /dev/sdb1
```

You can use an analogous procedure when using a TPM 2.0 policy instead of a Tang server.

Additional resources

For more information, see the following man page:

- **clevis-luks-unlockers(7)**

37.11. DEPLOYING HIGH-AVAILABILITY NBDE SYSTEMS

Tang provides two methods for building a high-availability deployment:

Client redundancy (recommended)

Clients should be configured with the ability to bind to multiple Tang servers. In this setup, each Tang server has its own keys and clients can decrypt by contacting a subset of these servers. Clevis already supports this workflow through its **sss** plug-in. Red Hat recommends this method for a high-availability deployment.

Key sharing

For redundancy purposes, more than one instance of Tang can be deployed. To set up a second or any subsequent instance, install the **tang** packages and copy the key directory to the new host using **rsync** over **SSH**. Note that Red Hat does not recommend this method because sharing keys increases the risk of key compromise and requires additional automation infrastructure.

37.11.1. High-available NBDE using Shamir's Secret Sharing

Shamir's Secret Sharing (SSS) is a cryptographic scheme that divides a secret into several unique parts. To reconstruct the secret, a number of parts is required. The number is called threshold and SSS is also referred to as a thresholding scheme.

Clevis provides an implementation of SSS. It creates a key and divides it into a number of pieces. Each piece is encrypted using another pin including even SSS recursively. Additionally, you define the threshold **t**. If an NBDE deployment decrypts at least **t** pieces, then it recovers the encryption key and the decryption process succeeds.

37.11.1.1. Example 1: Redundancy with two Tang servers

The following command decrypts a LUKS-encrypted device when at least one of two Tang servers is available:

```
# clevis luks bind -d /dev/sda1 sss '{"t":1,"pins": {"tang": [{"url": "http://tang1.srv"}, {"url": "http://tang2.srv"}]} }'
```

The previous command used the following configuration scheme:

```
{
  "t":1,
  "pins":{
    "tang":[
      {
        "url": "http://tang1.srv"
      },
      {
        "url": "http://tang2.srv"
      }
    ]
  }
}
```

In this configuration, the SSS threshold **t** is set to **1** and the **clevis luks bind** command successfully reconstructs the secret if at least one from two listed **tang** servers is available.

37.11.1.2. Example 2: Shared secret on a Tang server and a TPM device

The following command successfully decrypts a LUKS-encrypted device when both the **tang** server and the **tpm2** device are available:

```
# clevis luks bind -d /dev/sda1 sss '{"t":2,"pins": {"tang": [{"url": "http://tang1.srv"}], "tpm2": {"pcr_ids": "0,1"}}}
```

The configuration scheme with the SSS threshold 't' set to '2' is now:

```
{
  "t":2,
  "pins":{
    "tang":[
      {
        "url": "http://tang1.srv"
      }
    ],
    "tpm2":{
      "pcr_ids": "0,1"
    }
  }
}
```

Additional resources

- For more information about the recommended high-availability NBDE setup, see the following man pages:
 - tang(8)**, section **High Availability**
 - clevis(1)**, section **Shamir's Secret Sharing**
 - clevis-encrypt-sss(1)**

37.12. DEPLOYMENT OF VIRTUAL MACHINES IN A NBDE NETWORK

The **clevis luks bind** command does not change the LUKS master key. This implies that if you create a LUKS-encrypted image for use in a virtual machine or cloud environment, all the instances that run this image will share a master key. This is extremely insecure and should be avoided at all times.

This is not a limitation of Clevis but a design principle of LUKS. If you wish to have encrypted root volumes in a cloud, you need to make sure that you perform the installation process (usually using Kickstart) for each instance of Red Hat Enterprise Linux in a cloud as well. The images cannot be shared without also sharing a LUKS master key.

If you intend to deploy automated unlocking in a virtualized environment, Red Hat strongly recommends that you use systems such as lorax or virt-install together with a Kickstart file (see *Configuring automated enrollment of LUKS-encrypted volumes using Kickstart*) or another automated provisioning tool to ensure that each encrypted VM has a unique master key.

**NOTE**

Automated unlocking with a TPM 2.0 policy is not supported in a virtual machine.

Additional resources

For more information, see the following man page:

- **clevis-luks-bind(1)**

37.13. BUILDING AUTOMATICALLY-ENROLLABLE VM IMAGES FOR CLOUD ENVIRONMENTS USING NBDE

Deploying automatically-enrollable encrypted images in a cloud environment can provide a unique set of challenges. Like other virtualization environments, it is recommended to reduce the number of instances started from a single image to avoid sharing the LUKS master key.

Therefore, the best practice is to create customized images that are not shared in any public repository and that provide a base for the deployment of a limited amount of instances. The exact number of instances to create should be defined by deployment's security policies and based on the risk tolerance associated with the LUKS master key attack vector.

To build LUKS-enabled automated deployments, systems such as Lorax or virt-install together with a Kickstart file should be used to ensure master key uniqueness during the image building process.

Cloud environments enable two Tang server deployment options which we consider here. First, the Tang server can be deployed within the cloud environment itself. Second, the Tang server can be deployed outside of the cloud on independent infrastructure with a VPN link between the two infrastructures.

Deploying Tang natively in the cloud does allow for easy deployment. However, given that it shares infrastructure with the data persistence layer of ciphertext of other systems, it may be possible for both the Tang server's private key and the Clevis metadata to be stored on the same physical disk. Access to this physical disk permits a full compromise of the ciphertext data.

**IMPORTANT**

For this reason, Red Hat strongly recommends maintaining a physical separation between the location where the data is stored and the system where Tang is running. This separation between the cloud and the Tang server ensures that the Tang server's private key cannot be accidentally combined with the Clevis metadata. It also provides local control of the Tang server if the cloud infrastructure is at risk.

37.14. ADDITIONAL RESOURCES

- The **tang(8)**, **clevis(1)**, **jose(1)**, and **clevis-luks-unlockers(7)** man pages.
- The [How to set up Network-Bound Disk Encryption with multiple LUKS devices \(Clevis + Tang unlocking\)](#) Knowledgebase article.

CHAPTER 38. USING SELINUX

CHAPTER 39. GETTING STARTED WITH SELINUX

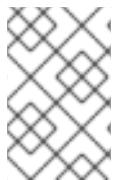
39.1. INTRODUCTION TO SELINUX

Security Enhanced Linux (SELinux) provides an additional layer of system security. SELinux fundamentally answers the question: *May <subject> do <action> to <object>?*, for example: *May a web server access files in users' home directories?*

The standard access policy based on the user, group, and other permissions, known as Discretionary Access Control (DAC), does not enable system administrators to create comprehensive and fine-grained security policies, such as restricting specific applications to only viewing log files, while allowing other applications to append new data to the log files.

SELinux implements Mandatory Access Control (MAC). Every process and system resource has a special security label called an *SELinux context*. A SELinux context, sometimes referred to as an *SELinux label*, is an identifier which abstracts away the system-level details and focuses on the security properties of the entity. Not only does this provide a consistent way of referencing objects in the SELinux policy, but it also removes any ambiguity that can be found in other identification methods. For example, a file can have multiple valid path names on a system that makes use of bind mounts.

The SELinux policy uses these contexts in a series of rules which define how processes can interact with each other and the various system resources. By default, the policy does not allow any interaction unless a rule explicitly grants access.



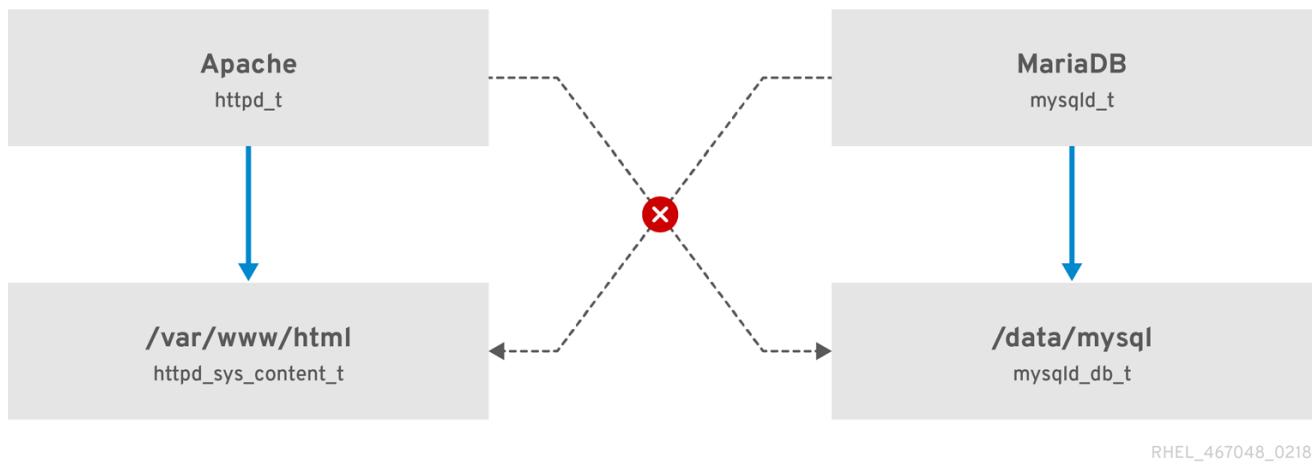
NOTE

Remember that SELinux policy rules are checked after DAC rules. SELinux policy rules are not used if DAC rules deny access first, which means that no SELinux denial is logged if the traditional DAC rules prevent the access.

SELinux contexts have several fields: user, role, type, and security level. The SELinux type information is perhaps the most important when it comes to the SELinux policy, as the most common policy rule which defines the allowed interactions between processes and system resources uses SELinux types and not the full SELinux context. SELinux types end with `_t`. For example, the type name for the web server is `httpd_t`. The type context for files and directories normally found in `/var/www/html/` is `httpd_sys_content_t`. The type contexts for files and directories normally found in `/tmp` and `/var/tmp/` is `tmp_t`. The type context for web server ports is `http_port_t`.

There is a policy rule that permits Apache (the web server process running as `httpd_t`) to access files and directories with a context normally found in `/var/www/html/` and other web server directories (`httpd_sys_content_t`). There is no allow rule in the policy for files normally found in `/tmp` and `/var/tmp/`, so access is not permitted. With SELinux, even if Apache is compromised, and a malicious script gains access, it is still not able to access the `/tmp` directory.

Figure 39.1. An example how can SELinux help to run Apache and MariaDB in a secure way.



As the previous scheme shows, SELinux allows the Apache process running as **httpd_t** to access the **/var/www/html/** directory and it denies the same process to access the **/data/mysql/** directory because there is no allow rule for the **httpd_t** and **mysqld_db_t** type contexts. On the other hand, the MariaDB process running as **mysqld_t** is able to access the **/data/mysql/** directory and SELinux also correctly denies the process with the **mysqld_t** type to access the **/var/www/html/** directory labeled as **httpd_sys_content_t**.

Additional resources

For more information, see the following documentation:

- The **selinux(8)** man page and man pages listed by the **apropos selinux** command.
- Man pages listed by the **man -k _selinux** command when the **selinux-policy-doc** package is installed.
- The [SELinux Coloring Book](#) helps you to better understand SELinux basic concepts.
- [SELinux Wiki FAQ](#)

39.2. BENEFITS OF RUNNING SELINUX

SELinux provides the following benefits:

- All processes and files are labeled. SELinux policy rules define how processes interact with files, as well as how processes interact with each other. Access is only allowed if an SELinux policy rule exists that specifically allows it.
- Fine-grained access control. Stepping beyond traditional UNIX permissions that are controlled at user discretion and based on Linux user and group IDs, SELinux access decisions are based on all available information, such as an SELinux user, role, type, and, optionally, a security level.
- SELinux policy is administratively-defined and enforced system-wide.
- Improved mitigation for privilege escalation attacks. Processes run in domains, and are therefore separated from each other. SELinux policy rules define how processes access files and other processes. If a process is compromised, the attacker only has access to the normal functions of that process, and to files the process has been configured to have access to. For

example, if the Apache HTTP Server is compromised, an attacker cannot use that process to read files in user home directories, unless a specific SELinux policy rule was added or configured to allow such access.

- SELinux can be used to enforce data confidentiality and integrity, as well as protecting processes from untrusted inputs.

However, SELinux is not:

- antivirus software,
- replacement for passwords, firewalls, and other security systems,
- all-in-one security solution.

SELinux is designed to enhance existing security solutions, not replace them. Even when running SELinux, it is important to continue to follow good security practices, such as keeping software up-to-date, using hard-to-guess passwords, and firewalls.

39.3. SELINUX EXAMPLES

The following examples demonstrate how SELinux increases security:

- The default action is deny. If an SELinux policy rule does not exist to allow access, such as for a process opening a file, access is denied.
- SELinux can confine Linux users. A number of confined SELinux users exist in the SELinux policy. Linux users can be mapped to confined SELinux users to take advantage of the security rules and mechanisms applied to them. For example, mapping a Linux user to the SELinux **user_u** user, results in a Linux user that is not able to run unless configured otherwise set user ID (setuid) applications, such as **sudo** and **su**, as well as preventing them from executing potentially malicious files and applications in their home directory.
- Increased process and data separation. The concept of SELinux *domains* allows defining which processes can access certain files and directories. For example, when running SELinux, unless otherwise configured, an attacker cannot compromise a Samba server, and then use that Samba server as an attack vector to read and write to files used by other processes, such as MariaDB databases.
- SELinux helps mitigate the damage made by configuration mistakes. Domain Name System (DNS) servers often replicate information between each other in what is known as a zone transfer. Attackers can use zone transfers to update DNS servers with false information. When running the Berkeley Internet Name Domain (BIND) as a DNS server in Red Hat Enterprise Linux, even if an administrator forgets to limit which servers can perform a zone transfer, the default SELinux policy prevents zone files^[2] from being updated using zone transfers, by the BIND **named** daemon itself, and by other processes.

39.4. SELINUX ARCHITECTURE AND PACKAGES

SELinux is a Linux Security Module (LSM) that is built into the Linux kernel. The SELinux subsystem in the kernel is driven by a security policy which is controlled by the administrator and loaded at boot. All security-relevant, kernel-level access operations on the system are intercepted by SELinux and examined in the context of the loaded security policy. If the loaded policy allows the operation, it continues. Otherwise, the operation is blocked and the process receives an error.

SELinux decisions, such as allowing or disallowing access, are cached. This cache is known as the Access Vector Cache (AVC). When using these cached decisions, SELinux policy rules need to be checked less, which increases performance. Remember that SELinux policy rules have no effect if DAC rules deny access first. Raw audit messages are logged to the `/var/log/audit/audit.log` and they start with the `type=AVC` string.

In Red Hat Enterprise Linux 8, system services are controlled by the **systemd** daemon; **systemd** starts and stops all services, and users and processes communicate with **systemd** using the **systemctl** utility. The **systemd** daemon can consult the SELinux policy and check the label of the calling process and the label of the unit file that the caller tries to manage, and then ask SELinux whether or not the caller is allowed the access. This approach strengthens access control to critical system capabilities, which include starting and stopping system services.

The **systemd** daemon also works as an SELinux Access Manager. It retrieves the label of the process running **systemctl** or the process that sent a **D-Bus** message to **systemd**. The daemon then looks up the label of the unit file that the process wanted to configure. Finally, **systemd** can retrieve information from the kernel if the SELinux policy allows the specific access between the process label and the unit file label. This means a compromised application that needs to interact with **systemd** for a specific service can now be confined by SELinux. Policy writers can also use these fine-grained controls to confine administrators.



IMPORTANT

To avoid incorrect SELinux labeling and subsequent problems, ensure that you start services using a **systemctl start** command.

Red Hat Enterprise Linux 8 provides the following packages for working with SELinux:

- policies: **selinux-policy-targeted**, **selinux-policy-mls**
- tools: **policycoreutils**, **policycoreutils-gui**, **libselinux-utils**, **policycoreutils-python-utils**, **setools-console**, **checkpolicy**

39.5. SELINUX STATES AND MODES

SELinux can run in one of three modes: enforcing, permissive, or disabled.

- Enforcing mode is the default, and recommended, mode of operation; in enforcing mode SELinux operates normally, enforcing the loaded security policy on the entire system.
- In permissive mode, the system acts as if SELinux is enforcing the loaded security policy, including labeling objects and emitting access denial entries in the logs, but it does not actually deny any operations. While not recommended for production systems, permissive mode can be helpful for SELinux policy development and debugging.
- Disabled mode is strongly discouraged; not only does the system avoid enforcing the SELinux policy, it also avoids labeling any persistent objects such as files, making it difficult to enable SELinux in the future.

Use the **setenforce** utility to change between enforcing and permissive mode. Changes made with **setenforce** do not persist across reboots. To change to enforcing mode, enter the **setenforce 1** command as the Linux root user. To change to permissive mode, enter the **setenforce 0** command. Use the **getenforce** utility to view the current SELinux mode:

```
# getenforce  
Enforcing
```

```
# setenforce 0  
# getenforce  
Permissive
```

```
# setenforce 1  
# getenforce  
Enforcing
```

In Red Hat Enterprise Linux, you can set individual domains to permissive mode while the system runs in enforcing mode. For example, to make the *httpd_t* domain permissive:

```
# semanage permissive -a httpd_t
```

Note that permissive domains are a powerful tool that can compromise security of your system. Red Hat recommends to use permissive domains with caution, for example, when debugging a specific scenario.

[2] Text files that include information, such as host name to IP address mappings, that are used by DNS servers.

CHAPTER 40. CHANGING SELINUX STATES AND MODES

When enabled, SELinux can run in one of two modes: enforcing or permissive. The following sections show how to permanently change into these modes.

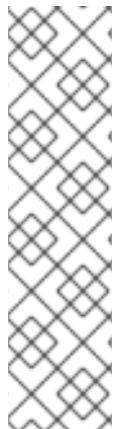
40.1. PERMANENT CHANGES IN SELINUX STATES AND MODES

As discussed in [SELinux states and modes](#), SELinux can be enabled or disabled. When enabled, SELinux has two modes: enforcing and permissive.

Use the **getenforce** or **sestatus** commands to check in which mode SELinux is running. The **getenforce** command returns **Enforcing**, **Permissive**, or **Disabled**.

The **sestatus** command returns the SELinux status and the SELinux policy being used:

```
$ 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
Memory protection checking: actual (secure)
Max kernel policy version: 31
```



NOTE

When systems run SELinux in permissive mode, users and processes can label various file-system objects incorrectly. File-system objects created while SELinux is disabled are not labeled at all. This behavior causes problems when changing to enforcing mode because SELinux relies on correct labels of file-system objects.

To prevent incorrectly labeled and unlabeled files from causing problems, file systems are automatically relabeled when changing from the disabled state to permissive or enforcing mode. In permissive mode, use the **fixfiles -F onboot** command as root to create the **./autorelabel** file containing the **-F** option to ensure that files are relabeled upon next reboot.

40.2. CHANGING TO PERMISSIVE MODE

Use the following procedure to permanently change SELinux mode to permissive. When SELinux is running in permissive mode, SELinux policy is not enforced. The system remains operational and SELinux does not deny any operations but only logs AVC messages, which can be then used for troubleshooting, debugging, and SELinux policy improvements. Each AVC is logged only once in this case.

Prerequisites

- The **selinux-policy-targeted**, **libselinux-utils**, and **policycoreutils** packages are installed on your system.
- The **selinux=0** or **enforcing=0** kernel parameters are not used.

Procedure

1. Open the **/etc/selinux/config** file in a text editor of your choice, for example:

```
# vi /etc/selinux/config
```

2. Configure the **SELINUX=permissive** option:

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.

SELINUX=permissive
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.

SELINUXTYPE=targeted
```

3. Restart the system:

```
# reboot
```

Verification steps

1. After the system restarts, confirm that the **getenforce** command returns **Permissive**:

```
$ getenforce
Permissive
```

40.3. CHANGING TO ENFORCING MODE

Use the following procedure to switch SELinux to enforcing mode. When SELinux is running in enforcing mode, it enforces the SELinux policy and denies access based on SELinux policy rules. In RHEL, enforcing mode is enabled by default when the system was initially installed with SELinux.

Prerequisites

- The **selinux-policy-targeted**, **libselinux-utils**, and **policycoreutils** packages are installed on your system.
- The **selinux=0** or **enforcing=0** kernel parameters are not used.

Procedure

1. Open the **/etc/selinux/config** file in a text editor of your choice, for example:

```
# vi /etc/selinux/config
```

2. Configure the **SELINUX=enforcing** option:

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
```

```

# enforcing - SELinux security policy is enforced.
# permissive - SELinux prints warnings instead of enforcing.
# disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these two values:
# targeted - Targeted processes are protected,
# mls - Multi Level Security protection.
SELINUXTYPE=targeted

```

3. Save the change, and restart the system:

```
# reboot
```

On the next boot, SELinux relabels all the files and directories within the system and adds SELinux context for files and directories that were created when SELinux was disabled.

Verification steps

1. After the system restarts, confirm that the **getenforce** command returns **Enforcing**:

```

$ getenforce
Enforcing

```

NOTE

After changing to enforcing mode, SELinux may deny some actions because of incorrect or missing SELinux policy rules. To view what actions SELinux denies, enter the following command as root:

```
# ausearch -m AVC,USER_AVC,SELINUX_ERR,USER_SELINUX_ERR -ts today
```

Alternatively, with the **setroubleshoot-server** package installed, enter:

```
# grep "SELinux is preventing" /var/log/messages
```

If SELinux is active and the Audit daemon (**auditd**) is not running on your system, then search for certain SELinux messages in the output of the **dmesg** command:

```
# dmesg | grep -i -e type=1300 -e type=1400
```

See [Troubleshooting problems related to SELinux](#) for more information.

40.4. ENABLING SELINUX ON SYSTEMS THAT PREVIOUSLY HAD IT DISABLED

When you enable SELinux on systems that previously had it disabled, to avoid problems, such as systems unable to boot or process failures, follow this procedure:

Procedure

1. Enable SELinux in permissive mode. For more information, see [Changing to permissive mode](#).

2. Restart your system:

```
# reboot
```

3. Check for SELinux denial messages. For more information, see [Identifying SELinux denials](#).
4. If there are no denials, switch to enforcing mode. For more information, see [Changing SELinux modes at boot time](#).

Verification steps

1. After the system restarts, confirm that the **getenforce** command returns **Enforcing**:

```
$ getenforce
Enforcing
```

Additional resources

- To run custom applications with SELinux in enforcing mode, choose one of the following scenarios:
 - Run your application in the **unconfined_service_t** domain.
 - Write a new policy for your application. See the [Writing Custom SELinux Policy](#) Knowledgebase article for more information.
- Temporary changes in modes are covered in [SELinux states and modes](#).

40.5. DISABLING SELINUX

Use the following procedure to permanently disable SELinux.



IMPORTANT

When SELinux is disabled, SELinux policy is not loaded at all; it is not enforced and AVC messages are not logged. Therefore, all [benefits of running SELinux](#) are lost.

Red Hat strongly recommends to use permissive mode instead of permanently disabling SELinux. See [Changing to permissive mode](#) for more information about permissive mode.



WARNING

Disabling SELinux using the **SELINUX=disabled** option in the **/etc/selinux/config** results in a process in which the kernel boots with SELinux enabled and switches to disabled mode later in the boot process. Because memory leaks and race conditions causing kernel panics can occur, prefer disabling SELinux by adding the **selinux=0** parameter to the kernel command line as described in [Changing SELinux modes at boot time](#) if your scenario really requires to completely disable SELinux.

Procedure

1. Open the **/etc/selinux/config** file in a text editor of your choice, for example:

```
# vi /etc/selinux/config
```

2. Configure the **SELINUX=disabled** option:

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

3. Save the change, and restart your system:

```
# reboot
```

Verification steps

1. After reboot, confirm that the **getenforce** command returns **Disabled**:

```
$ getenforce
Disabled
```

40.6. CHANGING SELINUX MODES AT BOOT TIME

On boot, you can set several kernel parameters to change the way SELinux runs:

enforcing=0

Setting this parameter causes the machine to boot in permissive mode, which is useful when troubleshooting issues. Using permissive mode might be the only option to detect a problem if your file system is too corrupted. Moreover, in permissive mode the system continues to create the labels correctly. The AVC messages that are created in this mode can be different than in enforcing mode. In permissive mode, only the first denial is reported. However, in enforcing mode you might get a denial on reading a directory and an application stops. In permissive mode, you get the same AVC message, but the application continues reading files in the directory and you get an AVC for each denial in addition.

selinux=0

This parameter causes the kernel to not load any part of the SELinux infrastructure. The init scripts notice that the system booted with the **selinux=0** parameter and touch the **/.autorelabel** file. This causes the system to automatically relabel the next time you boot with SELinux enabled.



IMPORTANT

Red Hat does not recommend using the **selinux=0** parameter. To debug your system, prefer using permissive mode.

autorelabel=1

This parameter forces the system to relabel similarly to the following commands:

```
# touch /.autorelabel  
# reboot
```

If a file system contains a large amount of mislabeled objects, start the system in permissive mode to make the autorelabel process successful.

Additional resources

- For additional SELinux-related kernel boot parameters, such as **checkreqprot**, see the **/usr/share/doc/kernel-doc-<KERNEL_VER>/Documentation/admin-guide/kernel-parameters.txt** file installed with the **kernel-doc** package. Replace the **<KERNEL_VER>** string with the version number of the installed kernel, for example:

```
# yum install kernel-doc  
$ less /usr/share/doc/kernel-doc-4.18.0/Documentation/admin-guide/kernel-parameters.txt
```

CHAPTER 41. TROUBLESHOOTING PROBLEMS RELATED TO SELINUX

If you plan to enable SELinux on systems where it has been previously disabled or if you run a service in a non-standard configuration, you might need to troubleshoot situations potentially blocked by SELinux. Note that in most cases, SELinux denials are signs of misconfiguration.

41.1. IDENTIFYING SELINUX DENIALS

Follow only the necessary steps from this procedure; in most cases, you need to perform just step 1.

Procedure

- When your scenario is blocked by SELinux, the **/var/log/audit/audit.log** file is the first place to check for more information about a denial. To query Audit logs, use the **ausearch** tool. Because the SELinux decisions, such as allowing or disallowing access, are cached and this cache is known as the Access Vector Cache (AVC), use the **AVC** and **USER_AVC** values for the message type parameter, for example:

```
# ausearch -m AVC,USER_AVC,SELINUX_ERR,USER_SELINUX_ERR -ts recent
```

If there are no matches, check if the Audit daemon is running. If it does not, repeat the denied scenario after you start **auditd** and check the Audit log again.

- In case **auditd** is running, but there are no matches in the output of **ausearch**, check messages provided by the **systemd** Journal:

```
# journalctl -t setroubleshoot
```

- If SELinux is active and the Audit daemon is not running on your system, then search for certain SELinux messages in the output of the **dmesg** command:

```
# dmesg | grep -i -e type=1300 -e type=1400
```

- Even after the previous three checks, it is still possible that you have not found anything. In this case, AVC denials can be silenced because of **dontaudit** rules.

To temporarily disable **dontaudit** rules, allowing all denials to be logged:

```
# semodule -DB
```

After re-running your denied scenario and finding denial messages using the previous steps, the following command enables **dontaudit** rules in the policy again:

```
# semodule -B
```

- If you apply all four previous steps, and the problem still remains unidentified, consider if SELinux really blocks your scenario:

- Switch to permissive mode:

```
# setenforce 0
$ getenforce
Permissive
```

- Repeat your scenario.

If the problem still occurs, something different than SELinux is blocking your scenario.

41.2. ANALYZING SELINUX DENIAL MESSAGES

After [identifying](#) that SELinux is blocking your scenario, you might need to analyze the root cause before you choose a fix.

Prerequisites

- The **policycoreutils-python-utils** and **setroubleshoot-server** packages are installed on your system.

Procedure

1. List more details about a logged denial using the **sealert** command, for example:

```
$ sealert -l ***
SELinux is preventing /usr/bin/passwd from write access on the file
/root/test.

***** Plugin leaks (86.2 confidence) suggests *****

If you want to ignore passwd trying to write access the test file,
because you believe it should not need this access.
Then you should report this as a bug.
You can generate a local policy module to dontaudit this access.
Do
# ausearch -x /usr/bin/passwd --raw | audit2allow -D -M my-passwd
# semodule -X 300 -i my-passwd.pp

***** Plugin catchall (14.7 confidence) suggests *****
```

...

Raw Audit Messages

```
type=AVC msg=audit(1553609555.619:127): avc: denied { write } for
pid=4097 comm="passwd" path="/root/test" dev="dm-0" ino=17142697
scontext=unconfined_u:unconfined_r:passwd_t:s0-s0:c0.c1023
tcontext=unconfined_u:object_r:admin_home_t:s0 tclass=file permissive=0

...
Hash: passwd,passwd_t,admin_home_t,file,write
```

2. If the output obtained in the previous step does not contain clear suggestions:

- Enable full-path auditing to see full paths to accessed objects and to make additional Linux Audit event fields visible:

- ```
auditctl -w /etc/shadow -p w -k shadow-write
```
- Clear the **setroubleshoot** cache:
 

```
rm -f /var/lib/setroubleshoot/setroubleshoot.xml
```
  - Reproduce the problem.
  - Repeat step 1.
3. If **sealert** returns only **catchall** suggestions or suggests adding a new rule using the **audit2allow** tool, match your problem with examples listed and explained in [SELinux denials in the Audit log](#).

#### Additional resources

- The **sealert(8)** man page.

### 41.3. FIXING ANALYZED SELINUX DENIALS

In most cases, suggestions provided by the **sealert** tool give you the right guidance about how to fix problems related to the SELinux policy. See [Analyzing SELinux denial messages](#) for information how to use **sealert** to analyze SELinux denials.

Be careful when the tool suggests using the **audit2allow** tool for configuration changes. You should not use **audit2allow** to generate a local policy module as your first option when you see an SELinux denial. Troubleshooting should start with a check if there is a labeling problem. The second most often case is that you have changed a process configuration, and you forgot to tell SELinux about it.

#### Labeling problems

A common cause of labeling problems is when a non-standard directory is used for a service. For example, instead of using **/var/www/html/** for a website, an administrator might want to use **/srv/myweb/**. On Red Hat Enterprise Linux, the **/srv** directory is labeled with the **var\_t** type. Files and directories created in **/srv** inherit this type. Also, newly-created objects in top-level directories, such as **/myserver**, can be labeled with the **default\_t** type. SELinux prevents the Apache HTTP Server ( **httpd** ) from accessing both of these types. To allow access, SELinux must know that the files in **/srv/myweb/** are to be accessible by **httpd**:

```
semanage fcontext -a -t httpd_sys_content_t "/srv/myweb(/.*)?"
```

This **semanage** command adds the context for the **/srv/myweb/** directory and all files and directories under it to the SELinux file-context configuration. The **semanage** utility does not change the context. As root, use the **restorecon** utility to apply the changes:

```
restorecon -R -v /srv/myweb
```

#### Incorrect context

The **matchpathcon** utility checks the context of a file path and compares it to the default label for that path. The following example demonstrates the use of **matchpathcon** on a directory that contains incorrectly labeled files:

```
$ matchpathcon -V /var/www/html/*
/var/www/html/index.html has context unconfined_u:object_r:user_home_t:s0, should be
```

```
system_u:object_r:httpd_sys_content_t:s0
/var/www/html/page1.html has context unconfined_u:object_r:user_home_t:s0, should be
system_u:object_r:httpd_sys_content_t:s0
```

In this example, the **index.html** and **page1.html** files are labeled with the **user\_home\_t** type. This type is used for files in user home directories. Using the **mv** command to move files from your home directory may result in files being labeled with the **user\_home\_t** type. This type should not exist outside of home directories. Use the **restorecon** utility to restore such files to their correct type:

```
restorecon -v /var/www/html/index.html
restorecon reset /var/www/html/index.html context unconfined_u:object_r:user_home_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
```

To restore the context for all files under a directory, use the **-R** option:

```
restorecon -R -v /var/www/html/
restorecon reset /var/www/html/page1.html context unconfined_u:object_r:samba_share_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
restorecon reset /var/www/html/index.html context unconfined_u:object_r:samba_share_t:s0-
>system_u:object_r:httpd_sys_content_t:s0
```

## Confined applications configured in non-standard ways

Services can be run in a variety of ways. To account for that, you need to specify how you run your services. You can achieve this through SELinux booleans that allow parts of SELinux policy to be changed at runtime. This enables changes, such as allowing services access to NFS volumes, without reloading or recompiling SELinux policy. Also, running services on non-default port numbers requires policy configuration to be updated using the **semanage** command.

For example, to allow the Apache HTTP Server to communicate with MariaDB, enable the **httpd\_can\_network\_connect\_db** boolean:

```
setsebool -P httpd_can_network_connect_db on
```

Note that the **-P** option makes the setting persistent across reboots of the system.

If access is denied for a particular service, use the **getsebool** and **grep** utilities to see if any booleans are available to allow access. For example, use the **getsebool -a | grep ftp** command to search for FTP related booleans:

```
$ getsebool -a | grep ftp
ftpd_anon_write --> off
ftpd_full_access --> off
ftpd_use_cifs --> off
ftpd_use_nfs --> off

ftpd_connect_db --> off
httpd_enable_ftp_server --> off
tftp_anon_write --> off
```

To get a list of booleans and to find out if they are enabled or disabled, use the **getsebool -a** command. To get a list of booleans including their meaning, and to find out if they are enabled or disabled, install the **selinux-policy-devel** package and use the **semanage boolean -l** command as root.

## Port numbers

Depending on policy configuration, services can only be allowed to run on certain port numbers. Attempting to change the port a service runs on without changing policy may result in the service failing to start. For example, run the **semanage port -l | grep http** command as root to list **http** related ports:

```
semanage port -l | grep http
http_cache_port_t tcp 3128, 8080, 8118
http_cache_port_t udp 3130
http_port_t tcp 80, 443, 488, 8008, 8009, 8443
pegasus_http_port_t tcp 5988
pegasus_https_port_t tcp 5989
```

The **http\_port\_t** port type defines the ports Apache HTTP Server can listen on, which in this case, are TCP ports 80, 443, 488, 8008, 8009, and 8443. If an administrator configures **httpd.conf** so that **httpd** listens on port 9876 (**Listen 9876**), but policy is not updated to reflect this, the following command fails:

```
systemctl start httpd.service
Job for httpd.service failed. See 'systemctl status httpd.service' and 'journalctl -xn' for details.

systemctl status httpd.service
httpd.service - The Apache HTTP Server
 Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled)
 Active: failed (Result: exit-code) since Thu 2013-08-15 09:57:05 CEST; 59s ago
 Process: 16874 ExecStop=/usr/sbin/httpd $OPTIONS -k graceful-stop (code=exited,
 Process: 16870 ExecStart=/usr/sbin/httpd $OPTIONS -DFOREGROUND (code=exited,
 status=1/FAILURE)
```

An SELinux denial message similar to the following is logged to **/var/log/audit/audit.log**:

```
type=AVC msg=audit(1225948455.061:294): avc: denied { name_bind } for pid=4997
comm="httpd" src=9876 scontext=unconfined_u:system_r:httpd_t:s0
tcontext=system_u:object_r:port_t:s0 tclass=tcp_socket
```

To allow **httpd** to listen on a port that is not listed for the **http\_port\_t** port type, use the **semanage port** command to assign a different label to the port:

```
semanage port -a -t http_port_t -p tcp 9876
```

The **-a** option adds a new record; the **-t** option defines a type; and the **-p** option defines a protocol. The last argument is the port number to add.

### Corner cases, evolving or broken applications, and compromised systems

Applications may contain bugs, causing SELinux to deny access. Also, SELinux rules are evolving – SELinux may not have seen an application running in a certain way, possibly causing it to deny access, even though the application is working as expected. For example, if a new version of PostgreSQL is released, it may perform actions the current policy does not account for, causing access to be denied, even though access should be allowed.

For these situations, after access is denied, use the **audit2allow** utility to create a custom policy module to allow access. You can report missing rules in the SELinux policy in [Red Hat Bugzilla](#). For Red Hat Enterprise Linux 8, create bugs against the **Red Hat Enterprise Linux 8** product, and select the **selinux-policy** component. Include the output of the **audit2allow -w -a** and **audit2allow -a** commands in such bug reports.

If an application asks for major security privileges, it could be a signal that the application is compromised. Use intrusion detection tools to inspect such suspicious behavior.

The [Solution Engine](#) on the [Red Hat Customer Portal](#) can also provide guidance in the form of an article containing a possible solution for the same or very similar problem you have. Select the relevant product and version and use SELinux-related keywords, such as `selinux` or `avc`, together with the name of your blocked service or application, for example: **selinux samba**.

## 41.4. SELINUX DENIALS IN THE AUDIT LOG

The Linux Audit system stores log entries in the `/var/log/audit/audit.log` file by default. To list only SELinux-related records, use the `ausearch` command with the message type parameter set to **AVC** and **AVC\_USER** at a minimum, for example:

```
ausearch -m AVC,USER_AVC,SELINUX_ERR,USER_SELINUX_ERR
```

An SELinux denial entry in the Audit log file can look as follows:

```
type=AVC msg=audit(1395177286.929:1638): avc: denied { read } for pid=6591 comm="httpd" name="webpages" dev="0:37" ino=2112 scontext=system_u:system_r:httpd_t:s0 tcontext=system_u:object_r:nfs_t:s0 tclass=dir
```

The most important parts of this entry are:

- **avc: denied** - the action performed by SELinux and recorded in Access Vector Cache (AVC)
- **{ read }** - the denied action
- **pid=6591** - the process identifier of the subject that tried to perform the denied action
- **comm="httpd"** - the name of the command that was used to invoke the analyzed process
- **httpd\_t** - the SELinux type of the process
- **nfs\_t** - the SELinux type of the object affected by the process action
- **tclass=dir** - the target object class

The previous log entry can be translated to:

*SELinux denied the **httpd** process with PID 6591 and the **httpd\_t** type to read from a directory with the **nfs\_t** type.*

The following SELinux denial message occurs when the Apache HTTP Server attempts to access a directory labeled with a type for the Samba suite:

```
type=AVC msg=audit(1226874073.147:96): avc: denied { getattr } for pid=2465 comm="httpd" path="/var/www/html/file1" dev=dm-0 ino=284133 scontext=unconfined_u:system_r:httpd_t:s0 tcontext=unconfined_u:object_r:samba_share_t:s0 tclass=file
```

- **{ getattr }** - the **getattr** entry indicates the source process was trying to read the target file's status information. This occurs before reading files. SELinux denies this action because the process accesses the file and it does not have an appropriate label. Commonly seen permissions include **getattr**, **read**, and **write**.

- **path="/var/www/html/file1"** - the path to the object (target) the process attempted to access.
- **scontext="unconfined\_u:system\_r:httpd\_t:s0"** - the SELinux context of the process (source) that attempted the denied action. In this case, it is the SELinux context of the Apache HTTP Server, which is running with the **httpd\_t** type.
- **tcontext="unconfined\_u:object\_r:samba\_share\_t:s0"** - the SELinux context of the object (target) the process attempted to access. In this case, it is the SELinux context of **file1**.

This SELinux denial can be translated to:

*SELinux denied the **httpd** process with PID 2465 to access the **/var/www/html/file1** file with the **samba\_share\_t** type, which is not accessible to processes running in the **httpd\_t** domain unless configured otherwise.*

#### Additional resources

- For more information, see the **auditd(8)** and **ausearch(8)** man pages.

## 41.5. RELATED INFORMATION

- The [Basic SELinux Troubleshooting in CLI](#) article on the Customer Portal.
- The [What is SELinux trying to tell me? The 4 key causes of SELinux errors](#) presentation on Fedora People

## PART III. DESIGN OF NETWORK

# CHAPTER 42. OVERVIEW OF NETWORKING TOPICS



## NOTE

The following sections mention some commands to be performed. The commands that need to be entered by the **root** user have `~]#` in the prompt, while the commands that can be performed by a regular user, have `~]$` in their prompt.

## 42.1. IP VERSUS NON-IP NETWORKS

A network is a system of interconnected devices that can communicate sharing information and resources, such as files, printers, applications, and Internet connection. Each of these devices has a unique Internet Protocol (IP) address to send and receive messages between two or more devices using a set of rules called protocol.

### Categories of network communication

#### IP networks

Networks that communicate through IP addresses. An IP network is implemented in the Internet and most internal networks. Ethernet, cable modems, DSL modems, dial-up modems, wireless networks, and VPN connections are typical examples.

#### non-IP networks

Networks that are used to communicate through a lower layer rather than the transport layer. Note that these networks are rarely used. InfiniBand is a non-IP network.

## 42.2. STATIC VERSUS DYNAMIC IP ADDRESSING

### Static IP addressing

When a device is assigned a static IP address, the address does not change over time unless changed manually. Use static **IP** addressing if you want:

- To ensure network address consistency for servers such as **DNS**, and authentication servers.
  - To use out-of-band management devices that work independently of other network infrastructure.
- All the configuration tools listed in [Section 45.1, "Selecting network configuration methods"](#) allow assigning static **IP** addresses manually.

### Dynamic IP addressing

When a device is assigned a dynamic IP address, the address changes over time. For this reason, it is recommended for devices that connect to the network occasionally because IP address might be changed after rebooting the machine.

Dynamic IP addresses are more flexible, easier to set up and administer. The **Dynamic Host Control Protocol (DHCP)** is a traditional method of dynamically assigning network configurations to hosts.



## NOTE

There is no strict rule defining when to use static or dynamic IP address. It depends on user's needs, preferences and the network environment.

## 42.3. CONFIGURING THE DHCP CLIENT BEHAVIOR

A Dynamic Host Configuration Protocol (DHCP) client requests the dynamic IP address and corresponding configuration information from a DHCP server each time a client connects to the network.

### Configuring the DHCP timeout

When a **DHCP** connection is started, a `dhcp` client requests an IP address from a **DHCP** server. The time that a `dhcp` client waits for this request to be completed is 45 seconds by default. This procedure describes how you can configure the **ipv4.dhcp-timeout** property using the `nmcli` tool or the **IPV4\_DHCP\_TIMEOUT** option in the `/etc/sysconfig/network-scripts/ifcfg-ifname` file. For example, using `nmcli`:

```
~]# nmcli connection modify enp1s0 ipv4.dhcp-timeout 10
```

If an address cannot be obtained during this interval, the IPv4 configuration fails. The whole connection may fail, too, and this depends on the **ipv4.may-fail** property:

- If **ipv4.may-fail** is set to **yes** (default), the state of the connection depends on IPv6 configuration:
  - a. If the IPv6 configuration is enabled and successful, the connection is activated, but the IPv4 configuration can never be retried again.
  - b. If the IPv6 configuration is disabled or does not get configured, the connection fails.
- If **ipv4.may-fail** is set to **no** the connection is deactivated. In this case:
  - a. If the **autoconnect** property of the connection is enabled, **NetworkManager** retries to activate the connection as many times as set in the **autoconnect-retries** property. The default is 4.
  - b. If the connection still cannot acquire the dhcp address, auto-activation fails.

Note that after 5 minutes, the auto-connection process starts again and the `dhcp` client retries to acquire an address from the `dhcp` server.

### Lease renewal and expiration

After a DHCP lease is acquired successfully, **NetworkManager** configures the interface with parameters received from the DHCP server for the given time, and tries to renew the lease periodically. When the lease expires and cannot be renewed, **NetworkManager** continues trying to contact the server up to 8 minutes. If the other IP configuration, either IPv4 or IPv6 is successful, DHCP requests continue as long as the connection is active.

#### 42.3.1. Making DHCPv4 persistent

To make DHCPv4 persistent both at startup and during the lease renewal processes, set the **ipv4.dhcp-timeout** property either to the maximum for a 32-bit integer (MAXINT32), which is **2147483647**, or to the **infinity** value:

```
~]$ nmcli connection modify enp1s0 ipv4.dhcp-timeout infinity
```

As a result, **NetworkManager** never stops trying to get or renew a lease from a DHCP server until it is successful.

To ensure a DHCP persistent behavior only during the lease renewal process, you can manually add a static IP to the **IPADDR** property in the **/etc/sysconfig/network-scripts/ifcfg-device\_name** configuration file or by using **nmcli**:

```
~]$ nmcli connection modify enp1s0 ipv4.address 192.168.122.88/24
```

When an IP address lease expires, the static IP preserves the IP state as configured or partially configured – you can have an IP address, but you are not connected to the Internet.

## 42.4. INFINIBAND AND RDMA NETWORKS

For details about InfiniBand and Remote Direct Memory Access (RDMA) networks, see the [Configuring InfiniBand and RDMA networks](#) documentation.

## 42.5. SETTING THE WIRELESS REGULATORY DOMAIN

In Red Hat Enterprise Linux, the **crda** package contains the Central Regulatory Domain Agent that provides the kernel with the wireless regulatory rules for a given jurisdiction. It is used by certain **udev** scripts and should not be run manually unless debugging **udev** scripts. The kernel runs **crda** by sending a **udev** event upon a new regulatory domain change. Regulatory domain changes are triggered by the Linux wireless subsystem (IEEE-802.11). This subsystem uses the **regulatory.bin** file to keep its regulatory database information.

The **setregdomain** utility sets the regulatory domain for your system. **Setregdomain** takes no arguments and is usually called through system script such as **udev** rather than manually by the administrator. If a country code look-up fails, the system administrator can define the **COUNTRY** environment variable in the **/etc/sysconfig/regdomain** file.

### Additional resources

See the following man pages for more information about the regulatory domain:

- **setregdomain(1)** man page – Sets regulatory domain based on country code.
- **crda(8)** man page – Sends to the kernel a wireless regulatory domain for a given ISO or IEC 3166 alpha2.
- **regulatory.bin(5)** man page – Shows the Linux wireless regulatory database.
- **iw(8)** man page – Shows or manipulates wireless devices and their configuration.

## 42.6. USING NETWORK KERNEL TUNABLES WITH SYSCTL

Using certain kernel tunables through the **sysctl** utility, you can adjust network configuration on a running system and directly affect the networking performance.

To change network settings, use the **sysctl** commands. For permanent changes that persist across system restarts, add lines to the **/etc/sysctl.conf** file.

To display a list of all available **sysctl** parameters, enter as **root**:

```
~]# sysctl -a
```

## 42.7. MANAGING DATA USING THE NCAT UTILITY

The **ncat** networking utility replaces **netcat** in Red Hat Enterprise Linux 7. **ncat** is a reliable back-end tool that provides network connectivity to other applications and users. It reads and writes data across the network from the command line, and uses Transmission Control Protocol (TCP), User Datagram Protocol (UDP), Stream Control Transmission Protocol (SCTP) or Unix sockets for communication. **ncat** can deal with both **IPv4** and **IPv6**, open connections, send packets, perform port scanning, and supports higher-level features such as **SSL**, and connection broker.

The **nc** command can also be entered as **ncat**, using the identical options. For more information about the **ncat** options, see [the New networking utility \(ncat\) section in the Migration Planning Guide](#) and the **ncat(1)** man page.

## Installing ncat

To install the **ncat** package, enter as **root**:

```
~]# yum install nmap-ncat
```

## Brief selection of ncat use cases

### Example 42.1. Enabling communication between a client and a server

1. Set a client machine to listen for connections on TCP port 8080:

```
~]$ ncat -l 8080
```

2. On a server machine, specify the IP address of the client and use the same port number:

```
~]$ ncat 10.0.11.60 8080
```

You can send messages on either side of the connection and they appear on both local and remote machines.

3. Press **Ctrl+D** to close the TCP connection.



## NOTE

To check a UDP port, use the same **nc** commands with the **-u** option. For example:

```
~]$ ncat -u -l 8080
```

### Example 42.2. Sending files

Instead of printing information on the screen, as mentioned in the previous example, you can send all information to a file. For example, to send a file over TCP port 8080 from a client to a server:

1. On a client machine, to listen a specific port transferring a file to the server machine:

```
~]$ ncat -l 8080 > outfile
```

2. On a server machine, specify the IP address of the client, the port and the file which is to be transferred:

```
~]$ ncat -l 10.0.11.60 8080 < inputfile
```

After the file is transferred, the connection closes automatically.



### NOTE

You can transfer a file in the other direction as well:

```
~]$ ncat -l 8080 < inputfile
```

```
~]$ ncat -l 10.0.11.60 8080 > outputfile
```

### Example 42.3. Creating an HTTP proxy server

To create an HTTP proxy server on localhost port 8080:

```
~]$ ncat -l --proxy-type http localhost 8080
```

### Example 42.4. Port scanning

To view which ports are open, use the **-z** option and specify a range of ports to scan:

```
~]$ ncat -z 10.0.11.60 80-90
```

Connection to 192.168.0.1 80 port [tcp/http] succeeded!

### Example 42.5. Setting up secure client-server communication using SSL

Set up **SSL** on a server:

```
~]$ ncat -e /bin/bash -k -l 8080 --ssl
```

On a client machine:

```
~]$ ncat --ssl 10.0.11.60 8080
```



### NOTE

To ensure true confidentiality of the **SSL** connection, the server requires the **--ssl-cert** and **--ssl-key** options, and the client requires the **--ssl-verify** and **--ssl-trustfile** options.

## Additional resources

For more examples, see the *ncat(1)* man page.

# CHAPTER 43. NETCONSOLE

The netconsole kernel module enables logging of kernel messages over the network to another computer. It allows kernel debugging when disk logging fails or when using the serial console is not possible.

## 43.1. CONFIGURING NETCONSOLE

This procedure describes how you can configure netconsole in Red Hat Enterprise Linux (RHEL) 8.

### Prerequisites

The **netconsole-service** package is installed.

```
~]# yum install netconsole-service
```

### Procedure

1. Set the **SYSLOGADDR** to the IP address of the **syslogd** server in the **/etc/sysconfig/netconsole** file to match the IP address of the **syslogd** server. For example:

```
SYSLOGADDR=192.168.0.1
```

2. Restart the **netconsole.service**.

```
~]# systemctl restart netconsole.service
```

3. Enable **netconsole.service** to run after rebooting the system.

```
~]# systemctl enable netconsole.service
```

4. View the **netconsole** messages from the client in the **/var/log/messages** file (default) or in the file specified in **rsyslog.conf**.

```
~]# cat /var/log/messages
```

### Additional resources

[How to configure netconsole under Red Hat Enterprise Linux 8 ?](#)

# CHAPTER 44. GETTING STARTED WITH MANAGING NETWORKING WITH NETWORKMANAGER

## 44.1. OVERVIEW OF NETWORKMANAGER

Red Hat Enterprise Linux 8 uses the default networking service, **NetworkManager**, which is a dynamic network control and configuration daemon to keep network devices and connections up and active when they are available. The traditional **ifcfg** type configuration files are still supported.

Each network device corresponds to a **NetworkManager** device. The configuration of a network device is completely stored in a single **NetworkManager** connection. You can perform a network configuration applying a **NetworkManager** connection to a **NetworkManager** device.

### 44.1.1. Benefits of using NetworkManager

The main benefits of using NetworkManager are:

- Offering an API through D-Bus which allows to query and control network configuration and state. In this way, networking can be checked and configured by multiple applications ensuring a synced and up-to-date networking status. For example, the RHEL web console, which monitors and configures servers through a web browser, uses the **NetworkManager** D-BUS interface to configure networking, as well as the **Gnome GUI**, the **nmcli** and the **nm-connection-editor** tools. Each change made in one of these tools is detected by all the others.
- Making Network management easier: **NetworkManager** ensures that network connectivity works. When it detects that there is no network configuration in a system but there are network devices, **NetworkManager** creates temporary connections to provide connectivity.
- Providing easy setup of connection to the user: **NetworkManager** offers management through different tools – **GUI**, **nmtui**, **nmcli**.
- Supporting configuration flexibility. For example, configuring a WiFi interface, **NetworkManager** scans and shows the available wifi networks. You can select an interface, and **NetworkManager** displays the required credentials providing automatic connection after the reboot process. **NetworkManager** can configure network aliases, IP addresses, static routes, DNS information, and VPN connections, as well as many connection-specific parameters. You can modify the configuration options to reflect your needs.
- Maintaining the state of devices after the reboot process and taking over interfaces which are set into managed mode during restart.
- Handling devices which are not explicitly set unmanaged but controlled manually by the user or another network service.

### Additional resources

- [Section 44.5, “NetworkManager tools”](#)
- For more information on installing and using the RHEL 8 web console, see [Managing systems using the RHEL 8 web console](#).

## 44.2. INSTALLING NETWORKMANAGER

**NetworkManager** is installed by default on Red Hat Enterprise Linux 8 . If it is not, enter as **root**:

```
~]# yum install NetworkManager
```

#### Additional resources

- [Section 44.1, “Overview of NetworkManager”](#)
- [Section 44.1.1, “Benefits of using NetworkManager”](#)

### 44.3. CHECKING THE STATUS OF NETWORKMANAGER

To check whether **NetworkManager** is running:

```
~]$ systemctl status NetworkManager
NetworkManager.service - Network Manager
 Loaded: loaded (/lib/systemd/system/NetworkManager.service; enabled)
 Active: active (running) since Fri, 08 Mar 2013 12:50:04 +0100; 3 days ago
```

Note that the **systemctl status** command displays **Active: inactive (dead)** when **NetworkManager** is not running.

### 44.4. STARTING NETWORKMANAGER

To start **NetworkManager**:

```
~]# systemctl start NetworkManager
```

To enable **NetworkManager** automatically at boot time:

```
~]# systemctl enable NetworkManager
```

### 44.5. NETWORKMANAGER TOOLS

Table 44.1. A summary of NetworkManager tools and applications

| Application or Tool | Description                                                                                                                                                                                                                                                                 |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>nmcli</b>        | A command-line tool which enables users and scripts to interact with <b>NetworkManager</b> . Note that <b>nmcli</b> can be used on systems without a GUI such as servers to control all aspects of <b>NetworkManager</b> . It provides a deeper functionality as GUI tools. |
| <b>nmtui</b>        | A simple curses-based text user interface (TUI) for <b>NetworkManager</b>                                                                                                                                                                                                   |

| Application or Tool               | Description                                                                                                                                                                                                                                                                                                         |
|-----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>nm-connection-editor</code> | A graphical user interface tool for certain tasks not yet handled by the <b>control-center</b> utility such as configuring bonds and teaming connections. You can add, remove, and modify network connections stored by <b>NetworkManager</b> . To start it, enter <code>nm-connection-editor</code> in a terminal. |
| <code>control-center</code>       | A graphical user interface tool provided by the GNOME Shell, available for desktop users. It incorporates a Network settings tool. To start it, press the <b>Super</b> key to enter the Activities Overview, type <b>Network</b> and then press <b>Enter</b> . The Network settings tool appears.                   |
| <b>network connection icon</b>    | A graphical user interface tool provided by the GNOME Shell representing network connection states as reported by <b>NetworkManager</b> . The icon has multiple states that serve as visual indicators for the type of connection you are currently using.                                                          |

#### Additional resources

- [Configuring IP networking with nmtui](#)
- [Getting started with nmcli](#)
- [Getting started with configuring networking using the GNOME GUI](#)

## 44.6. RUNNING DISPATCHER SCRIPTS

**NetworkManager** provides a way to run additional custom scripts to start or stop services based on the connection status. By default, the `/etc/NetworkManager/dispatcher.d/` directory exists and **NetworkManager** runs scripts there, in alphabetical order. Each script must be an executable file owned by **root** and must have **write permission** only for the file owner.

#### Additional resources

- For more information about running NetworkManager dispatcher scripts, see the Red Hat Knowledgebase solution [How to write a NetworkManager dispatcher script to apply ethtool commands](#).

## 44.7. USING NETWORKMANAGER WITH SYSCONFIG FILES

The `/etc/sysconfig/` directory is a location for configuration files and scripts. Most network configuration information is stored there, with the exception of VPN, mobile broadband and PPPoE configuration, which are stored in the `/etc/NetworkManager/` subdirectories. For example, interface-specific information is stored in the `ifcfg` files in the `/etc/sysconfig/network-scripts/` directory.

Information for VPNs, mobile broadband and PPPoE connections is stored in `/etc/NetworkManager/system-connections/`.

In Red Hat Enterprise Linux 8, if you edit an **ifcfg** file, **NetworkManager** is not automatically aware of the change and has to be prompted to notice the change. If you use one of the tools to update **NetworkManager** profile settings, **NetworkManager** does not implement those changes until you reconnect using that profile. For example, if configuration files have been changed using an editor, **NetworkManager** must read the configuration files again.

To ensure this, enter as **root** to reload all connection profiles:

```
~]# nmcli connection reload
```

Alternatively, to reload **only one** changed file, **ifcfg-ifname**:

```
~]# nmcli con load /etc/sysconfig/network-scripts/ifcfg-ifname
```

Note that you can specify multiple file names using the above command.

To restart the connection after changes are made, use:

```
~]# nmcli con up connection-name
```

#### 44.7.1. Legacy network scripts support

Network scripts are deprecated in Red Hat Enterprise Linux 8 and are no longer provided by default. The basic installation provides a new version of the **ifup** and **ifdown** scripts which call **NetworkManager** through the **nmcli** tool. In Red Hat Enterprise Linux 8, to run the **ifup** and the **ifdown** scripts, **NetworkManager** must be running.



#### NOTE

Custom commands in **/sbin/ifup-local**, **ifdown-pre-local** and **ifdown-local** scripts are not executed.

If any of these scripts are required, the installation of the deprecated network scripts in the system is still possible with the following command:

```
~]# yum install network-scripts
```

The **ifup** and the **ifdown** scripts link to the installed legacy network scripts.

Calling the legacy network scripts shows a warning about their deprecation.

#### Additional resources

- **NetworkManager(8)** man page – Describes the network management daemon.
- **NetworkManager.conf(5)** man page – Describes the **NetworkManager** configuration file.
- **/usr/share/doc/initscripts/sysconfig.txt** – Describes **ifcfg** configuration files and their directives as understood by the legacy network service.
- **ifcfg(8)** man page – Describes briefly the **ifcfg** command.

# CHAPTER 45. OVERVIEW OF NETWORK CONFIGURATION METHODS

The following section provides an overview of network configuration methods that are available in Red Hat Enterprise Linux 8.

## 45.1. SELECTING NETWORK CONFIGURATION METHODS

- To configure a network interface using **NetworkManager**, use one of the following tools:
  - the text user interface tool, **nmtui**.
  - the command-line tool, **nmcli**.
  - the graphical user interface tools, **GNOME GUI**.
- To configure a network interface **without** using **NetworkManager**:
  - edit the **ifcfg** files manually.
- To configure the network settings when the root filesystem is **not** local:
  - use the kernel command-line.

### Additional resources

- [Configuring IP networking with nmtui](#)
- [Getting started with nmcli](#)
- [Getting started with configuring networking using the GNOME GUI](#)

# CHAPTER 46. CONFIGURING IP NETWORKING WITH NMUI

The following section provides how you can configure a network interface using the **NetworkManager**'s tool, **nmtui**.

## 46.1. GETTING STARTED WITH NMUI

**nmtui** is a simple curses-based text user interface (TUI) for **NetworkManager**.

This procedure describes how to start the text user interface tool, **nmtui**.

### Prerequisites

- The **nmtui** tool is used in a terminal window. It is contained in the **NetworkManager-tui** package, but it is not installed along with **NetworkManager** by default. To install **NetworkManager-tui**:

```
~]# yum install NetworkManager-tui
```

- To verify that **NetworkManager** is running, see [Section 44.3, “Checking the status of NetworkManager”](#)

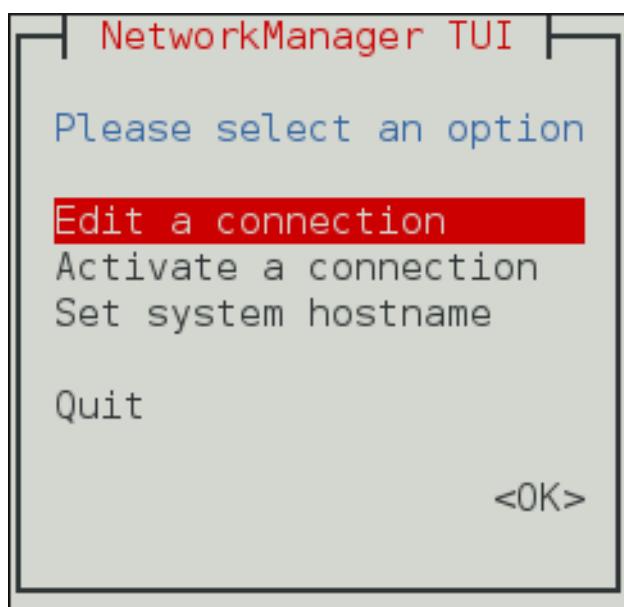
### Procedure

1. Start the **nmtui** tool:

```
~]$ nmtui
```

The text user interface appears.

Figure 46.1. The NetworkManager text user interface starting menu



2. To navigate, use the arrow keys or press **Tab** to step forwards and press **Shift+Tab** to step back through the options. Press **Enter** to select an option. The **Space** bar toggles the status of a check box.

### 46.1.1. Adding a connection profile using nmtui

The **nmtui** application provides a text user interface to NetworkManager. This procedure describes how to add a new connection profile.

#### Prerequisites

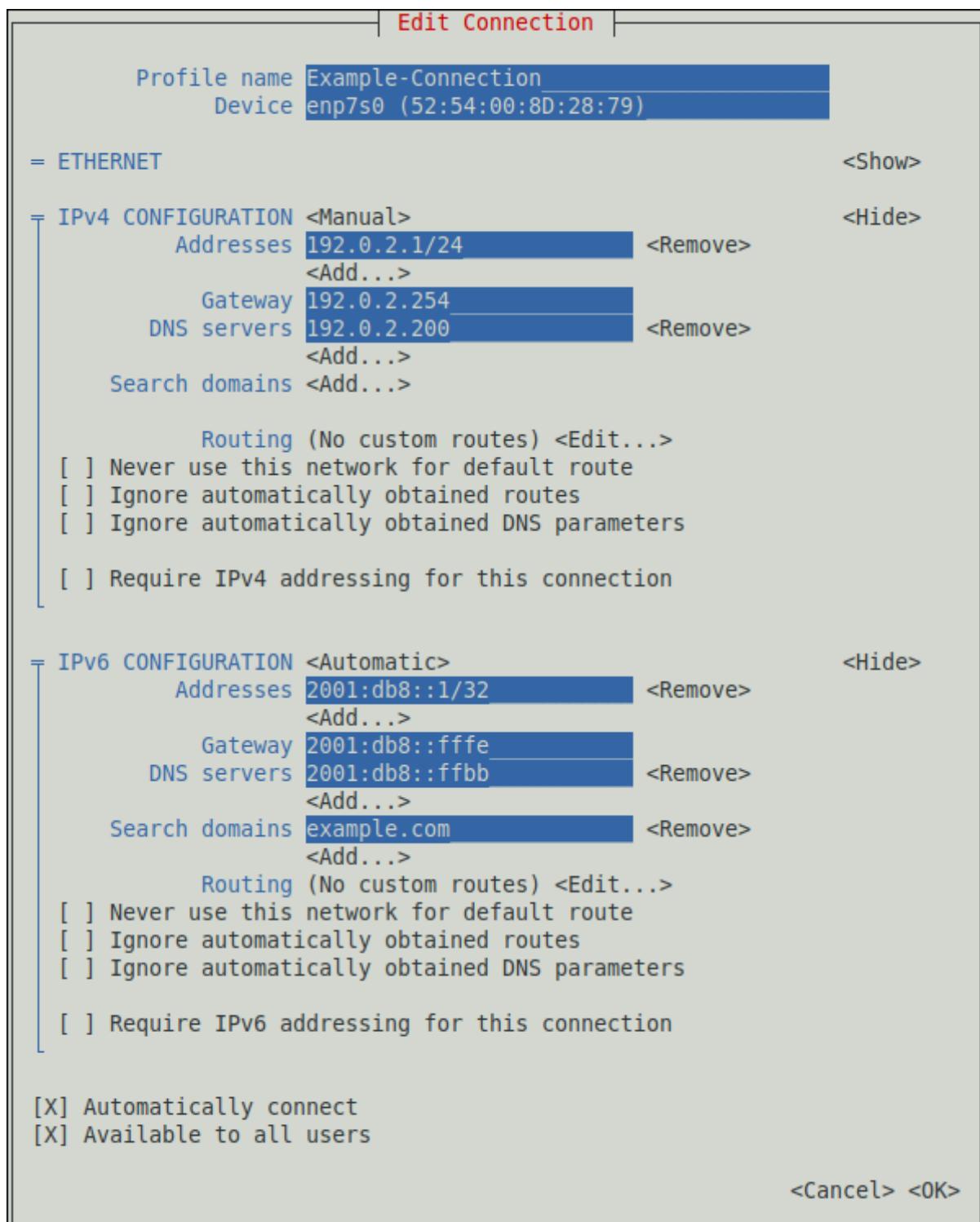
- The **NetworkManager-tui** package is installed.

#### Procedure

1. Start the NetworkManager text user interface utility:

```
█ # nmtui
```

2. Select the **Edit a connection** menu entry, and press **Enter**.
3. Select the **Add** button, and press **Enter**.
4. Select **Ethernet**, and press **Enter**.
5. Fill the fields with the connection details.



6. Select **OK** to save the changes.
7. Select **Back** to return to the main menu.
8. Select **Activate a connection**, and press **Enter**.
9. Select the new connection entry, and press **Enter** to activate the connection.
10. Select **Back** to return to the main menu.
11. Select **Quit**.

#### Verification steps

1. Display the status of the devices and connections:

```
nmcli device status
DEVICE TYPE STATE CONNECTION
enp7s0 ethernet connected Example-Connection
```

2. To display all settings of the connection profile:

```
nmcli connection show Example-Connection
connection.id: Example-Connection
connection.uuid: b6cd1c-e4ad-46e5-af8b-a75f06b79f76
connection.stable-id: --
connection.type: 802-3-ethernet
connection.interface-name: enp7s0
...
...
```

## Additional resources

- For further details about the **nmtui** application, see the **nmtui(1)** man page.

### 46.1.2. Applying changes to a modified connection with nmtui

To apply changes after a modified connection which is already active requires a reactivation of the connection. In this case, follow the procedure below:

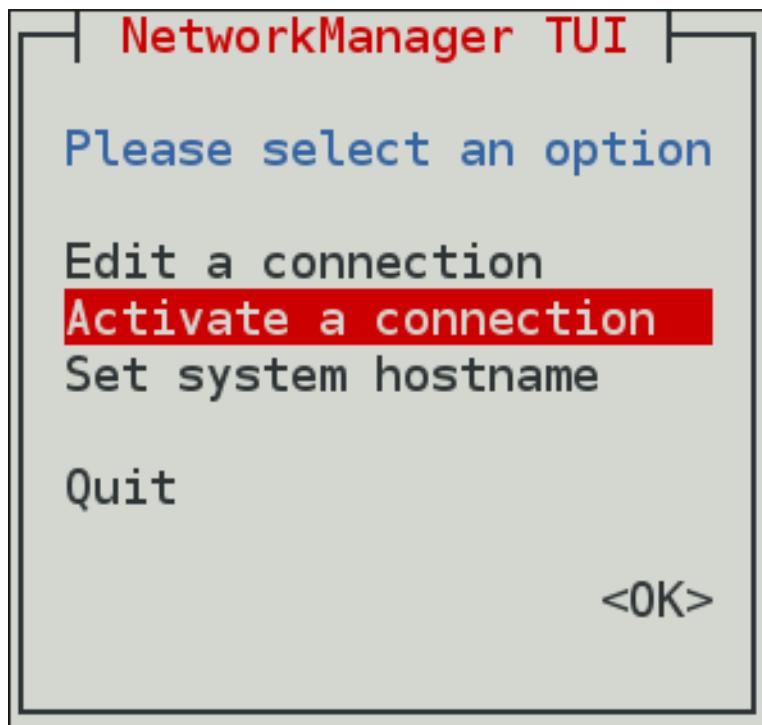
#### Prerequisites

- [Section 46.1, "Getting started with nmtui"](#)

#### Procedure

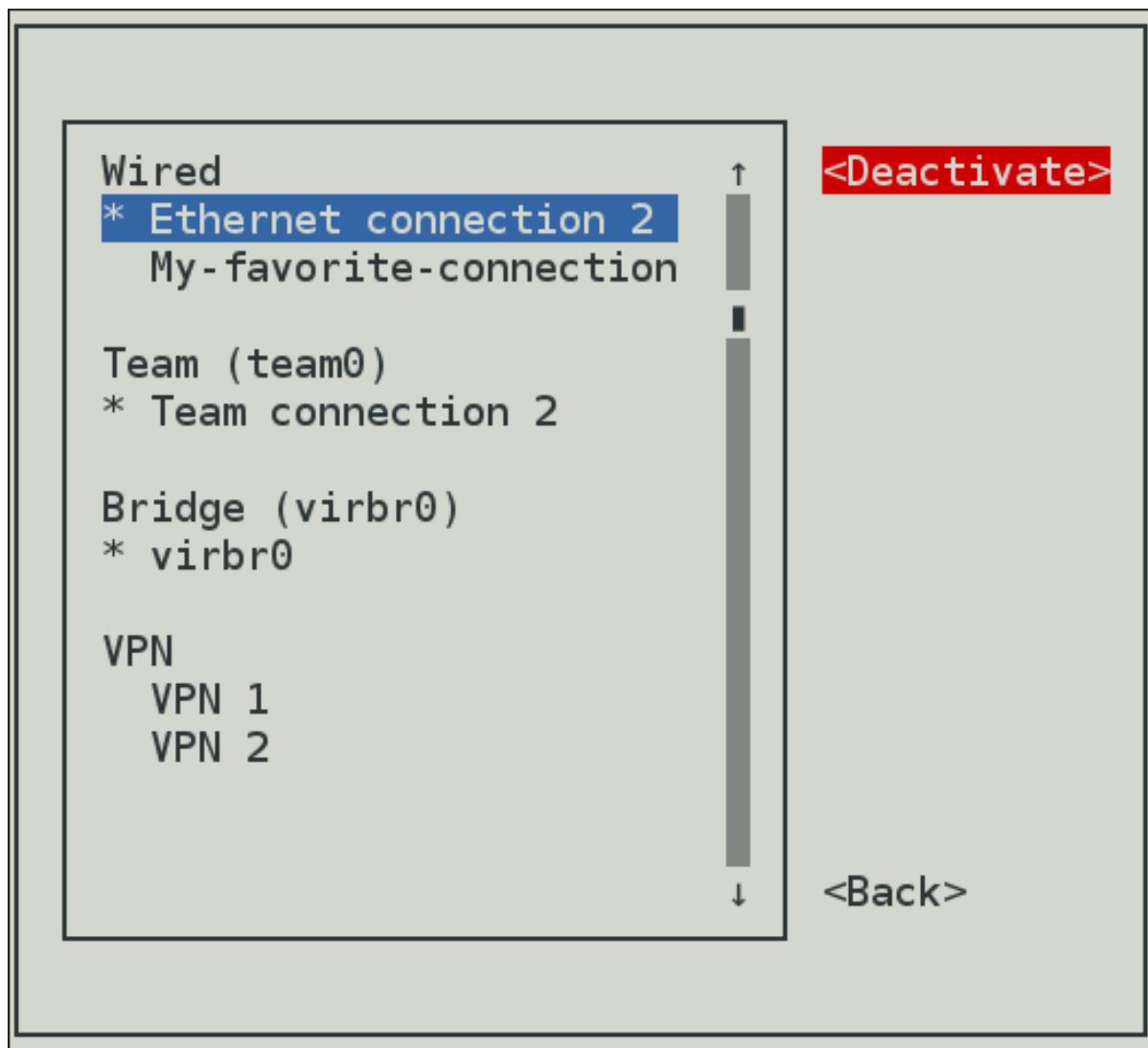
1. Select the **Activate a connection** menu entry.

Figure 46.2. Activating a connection with nmtui



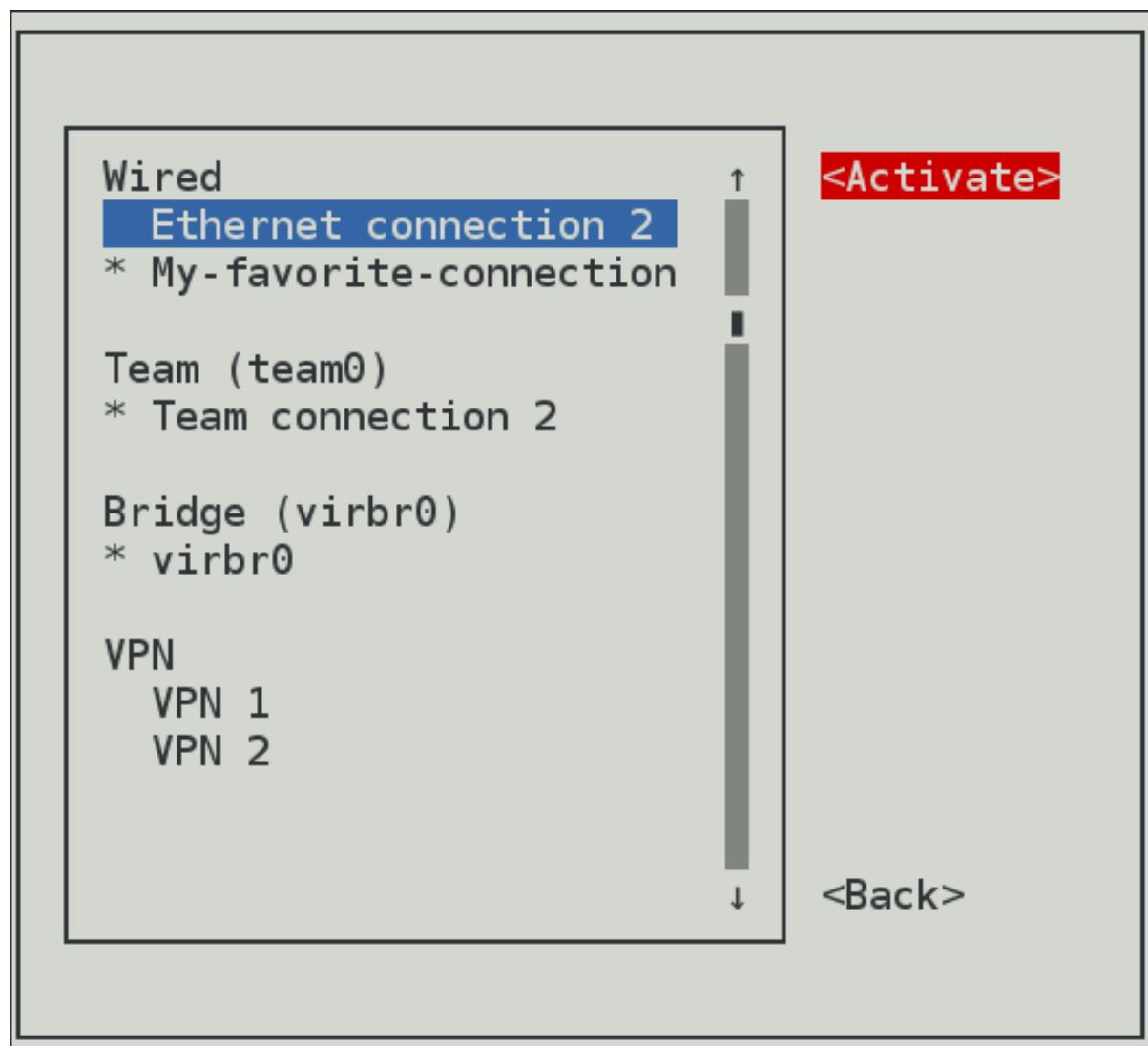
2. Select the modified connection. On the right, click the **Deactivate** button.

Figure 46.3. Deactivating a modified connection with nmtui



3. Choose the connection again and click the **Activate** button.

Figure 46.4. Reactivating a modified connection with nmtui



The following commands are also available:

```
~]$ nmtui edit connection-name
```

If no connection name is supplied, the selection menu appears. If the connection name is supplied and correctly identified, the relevant **Edit connection** screen appears.

```
~]$ nmtui connect connection-name
```

If no connection name is supplied, the selection menu appears. If the connection name is supplied and correctly identified, the relevant connection is activated. Any invalid command prints a usage message.

Note that **nmtui** does not support all types of connections. In particular, you cannot edit VPNs, wireless network connections using WPA Enterprise, or Ethernet connections using **802.1X**.

## Additional resources

- For more information about the **NetworkManager**'s tools, see [Section 44.5, “NetworkManager tools”](#)

# CHAPTER 47. GETTING STARTED WITH NMCLI

This section describes general information about the **nmcli** utility.

## 47.1. UNDERSTANDING NMCLI

**nmcli** (NetworkManager Command Line Interface) is the command-line utility to configure networking through **NetworkManager**. **nmcli** is used to create, display, edit, delete, activate, and deactivate network connections, as well as control and display network device status.

The **nmcli** utility can be used by both users and scripts:

- For servers, headless machines, and terminals, **nmcli** can be used to control **NetworkManager** directly, without GUI.
- For scripts, **nmcli** supports options to change the output to a format better suited for script processing.

Each network device corresponds to a **NetworkManager** device. The configuration of a network device is completely stored in a single **NetworkManager** connection. You can perform a network configuration applying a **NetworkManager** connection to a **NetworkManager** device.

To get started with **nmcli** the most common **nmcli** commands are **nmcli device** and **nmcli connection**:

- The **nmcli device** command lists the available network devices in the system.

A device can be:

1. **managed** - under the **NetworkManager** control. A **managed** device may be **connected**, meaning that it is activated and configured, or **disconnected**, meaning that it is not configured but ready to be activated again.
2. **unmanaged** - **NetworkManager** does not control it.

For more details on setting a **managed** or **unmanaged** device, see [Section 47.4, “Setting a device managed or unmanaged with nmcli”](#).

The **nmcli device** command can take many arguments. Most notable are: **status**, **show**, **set**, **connect**, **disconnect**, **modify**, **delete**, **wifi**. Enter the **nmcli device help** command to see the full list.

- The **nmcli connection** command lists the available connection profiles in **NetworkManager**.

Every connection that is active is displayed as green on top of the list. The inactive connections are displayed as white. The **DEVICE** field identifies the device on which the connection is applied on.

The **nmcli connection** command can take many arguments to manage connection profiles. Most notable are: **show**, **up**, **down**, **add**, **modify**, **delete**. Enter the **nmcli connection help** command to see the full list.



## IMPORTANT

If you use the **nmcli** commands, it is recommended to type a partial **nmcli** command, and then press the **Tab** key to auto-complete the command sequence. If multiple completions are possible, then **Tab** lists them all. This helps users to type commands faster and easier. To enable the **nmcli** auto-complete feature be sure to install the **bash-completion** package:

```
~]# yum install bash-completion
```

After the package installation, **nmcli auto-complete** will be available next time you login into a console. To activate it also in the current console, enter:

```
~]$ source /etc/profile.d/bash_completion.sh
```

The basic format of using **nmcli** is:

```
nmcli [OPTIONS] OBJECT { COMMAND | help }
```

- where [OPTIONS] can be optional options, such as:

### **-t, terse**

This mode can be used for computer script processing as you can see a terse output displaying only the values.

#### Example 47.1. Viewing a terse output

```
~]$ nmcli -t device
ens3:ethernet:connected:Profile 1
lo:loopback:unmanaged:
```

### **-f, field**

This option specifies what fields can be displayed in output. For example, NAME,UUID,TYPE,AUTOCONNECT,ACTIVE,DEVICE,STATE. You can use one or more fields. If you want to use more, do not use space after comma to separate the fields.

#### Example 47.2. Specifying fields in the output

```
~]$ nmcli -f DEVICE,TYPE device
DEVICE TYPE
ens3 ethernet
lo loopback
```

or even better for scripting:

```
~]$ nmcli -t -f DEVICE,TYPE device
ens3:ethernet
lo:loopback
```

### **-p, pretty**

This option causes **nmcli** to produce human-readable output. For example, values are aligned and headers are printed.

#### Example 47.3. Viewing an output in pretty mode

```
~]$ nmcli -p device
=====
Status of devices
=====
DEVICE TYPE STATE CONNECTION

ens3 ethernet connected Profile 1
lo loopback unmanaged --
```

#### **-h, help**

Prints help information.

- where OBJECT can be one of the following options: **general**, **networking**, **radio**, **connection**, **device**, **agent**, and **monitor**.



#### NOTE

You can use any prefix of the above options in your commands. For example, **nmcli con help**, **nmcli c help**, **nmcli connection help** generate the same output.

- where COMMAND, the required **nmcli** command.
- where *help* is to list available actions related to a specified object:

```
~]$ nmcli OBJECT help
```

For example,

```
~]$ nmcli c help
```

#### Additional resources

- [Section 44.5, “NetworkManager tools”](#)
- the **nmcli(1)** man page.
- [Section 47.3, “Brief selection of nmcli commands”](#)
- [Section 47.5, “Creating a connection profile with nmcli”](#)

## 47.2. OVERVIEW OF NMCLI PROPERTY NAMES AND ALIASES

### Prerequisites

**Property** names are specific names that **NetworkManager** uses to identify a common option. Following are some of the important **nmcli** **property** names:

## connection.type

A type of a specific connection. Allowed values are: **adsl**, **bond**, **bond-slave**, **bridge**, **bridge-slave**, **bluetooth**, **cdma**, **ethernet**, **gsm**, **infiniband**, **olpc-mesh**, **team**, **team-slave**, **vlan**, **wifi**, **wimax**. Each connection type has type-specific command options. You can see the **TYPE\_SPECIFIC\_OPTIONS** list in the **nmcli(1)** man page. For example, a **gsm** connection requires the access point name specified in an **apn**. A **wifi** device requires the service set identifier specified in a **ssid**.

## connection.interface-name

A device name relevant for the connection. For example, **enp1s0**.

## connection.id

A name used for the connection profile. If you do not specify a connection name, one will be generated as follows:

**connection.type -connection.interface-name**

The **connection.id** is the name of a *connection profile* and should not be confused with the interface name which denotes a device (**wlan0**, **ens3**, **em1**). However, users can name the connections after interfaces, but they are not the same thing. There can be multiple connection profiles available for a device. This is particularly useful for mobile devices or when switching a network cable back and forth between different devices. Rather than edit the configuration, create different profiles and apply them to the interface as needed. The **id** option also refers to the connection profile name.

The most important options for **nmcli** commands such as **show**, **up**, **down** are:

### id

An identification string assigned by the user to a connection profile. Id can be used in **nmcli** connection commands to identify a connection. The NAME field in the command output always denotes the connection id. It refers to the same connection profile name that the con-name does.

### uuid

A unique identification string assigned by the system to a connection profile. The **uuid** can be used in **nmcli connection** commands to identify a connection.

## Aliases and property names

An **alias** is an alternative name for a **property** name – aliases are translated to properties internally in **nmcli**. **Aliases** are more readable but **property names** are preferable to use. Both can be used interchangeably.

| Alias  | Example              | Property                  | Example                                 | Definition                                                                                                                                                     |
|--------|----------------------|---------------------------|-----------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| type   | <b>type bond</b>     | connection.type           | <b>connection.type bond</b>             | type of a specific connection. Some of the connection types are: <b>bond</b> , <b>bridge</b> , <b>ethernet</b> , <b>wifi</b> , <b>infiniband</b> , <b>vlan</b> |
| ifname | <b>ifname enp1s0</b> | connection.interface-name | <b>connection.interface-name enp1s0</b> | name of the device to which a connection belongs to                                                                                                            |

| Alias    | Example                               | Property                   | Example                                    | Definition           |
|----------|---------------------------------------|----------------------------|--------------------------------------------|----------------------|
| con-name | <code>con-name "My Connection"</code> | <code>connection.id</code> | <code>connection.id "My Connection"</code> | name of a connection |

## 47.3. BRIEF SELECTION OF NMCLI COMMANDS



### IMPORTANT

If you use the `nmcli` commands, it is recommended to type a partial `nmcli` command, and then press the **Tab** key to auto-complete the command sequence. If multiple completions are possible, then **Tab** lists them all. This helps users to type commands faster and easier. To enable the `nmcli` auto-complete feature be sure to install the **bash-completion** package:

```
~]# yum install bash-completion
```

After the package installation, **nmcli auto-complete** will be available next time you login into a console. To activate it also in the current console, enter:

```
~]$ source /etc/profile.d/bash_completion.sh
```

The following examples show how to use `nmcli` in specific use cases:

#### Example 47.4. Viewing all connections

```
~]$ nmcli connection show
 NAME UUID TYPE DEVICE
Profile 1 db1060e9-c164-476f-b2b5-caec62dc1b05 ethernet ens3
bond0 aaf6eb56-73e5-4746-9037-eed42caa8a65 ethernet --
```

#### Example 47.5. Viewing only currently active connections

```
~]$ nmcli connection show --active
 NAME UUID TYPE DEVICE
Profile 1 db1060e9-c164-476f-b2b5-caec62dc1b05 ethernet ens3
```

#### Example 47.6. Activating a connection

Use the **up** argument to activate a connection.

```
~]$ nmcli connection show
 NAME UUID TYPE DEVICE
Profile 1 db1060e9-c164-476f-b2b5-caec62dc1b05 ethernet ens3
bond0 aaf6eb56-73e5-4746-9037-eed42caa8a65 ethernet --
```

```
~]$ nmcli connection up id bond0
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/4)
```

```
~]$ nmcli connection show
NAME UUID TYPE DEVICE
Profile 1 db1060e9-c164-476f-b2b5-caec62dc1b05 ethernet ens3
bond0 aaf6eb56-73e5-4746-9037-eed42caa8a65 ethernet bond0
```

#### Example 47.7. Deactivating a specific active connection

Use the **down** argument to deactivate a specific active connection:

```
~]$ nmcli connection down id bond0
```

```
~]$ nmcli connection show
NAME UUID TYPE DEVICE
Profile 1 db1060e9-c164-476f-b2b5-caec62dc1b05 ethernet ens3
bond0 aaf6eb56-73e5-4746-9037-eed42caa8a65 ethernet --
```

#### Example 47.8. Disconnecting a device preventing it from automatically started again

```
~]$ nmcli device disconnect id bond0
```



#### NOTE

The **nmcli connection down** command, deactivates a connection from a device without preventing the device from further auto-activation. The **nmcli device disconnect** command, disconnects a device and prevent the device from automatically activating further connections without manual intervention. If the connection has the **connection.autoconnect** flag set to **yes**, the connection automatically starts on the disconnected device again. In this case, use the **nmcli device disconnect** command instead of the **nmcli connection down** command.

#### Example 47.9. Viewing only devices recognized by NetworkManager and their state

```
~]$ nmcli device status
DEVICE TYPE STATE CONNECTION
ens3 ethernet connected Profile 1
lo loopback unmanaged --
```

#### Example 47.10. Viewing general information for a device

```
~]$ nmcli device show
GENERAL.DEVICE: ens3
GENERAL.TYPE: ethernet
```

```
GENERAL.HWADDR: 52:54:00:0A:2F:ED
GENERAL.MTU: 1500
GENERAL.STATE: 100 (connected)
GENERAL.CONNECTION: ens3
[...]
```

#### Example 47.11. Checking the overall status of NetworkManager

```
~]$ nmcli general status
STATE CONNECTIVITY WIFI-HW WIFI WWAN-HW WWAN
connected full enabled enabled enabled enabled
```

In terse mode:

```
~]$ nmcli -t -f STATE general
connected
```

#### Example 47.12. Viewing NetworkManager logging status

```
~]$ nmcli general logging
LEVEL DOMAINS
INFO PLATFORM,RFKILL,ETHER,WIFI,BT,MB,DHCP4,DHCP6,PPP,WIFI_SCAN,IP4,IP6,A
UTOIP4,DNS,VPN,SHARING,SUPPLICANT,AGENTS,SETTINGS,SUSPEND,CORE,DEVICE,OL
PC,
WIMAX,INFINIBAND,FIREWALL,ADSL,BOND,VLAN,BRIDGE,DBUS_PROPS,TEAM,CONCHECK
,DC
B,DISPATCH
```

You can also use the following abbreviations of the **nmcli** commands:

Table 47.1. Abbreviations of some nmcli commands

| nmcli command                  | abbreviation                    |
|--------------------------------|---------------------------------|
| nmcli general status           | nmcli g                         |
| nmcli general logging          | nmcli g log                     |
| nmcli connection show          | nmcli con show or nmcli c       |
| nmcli connection show --active | nmcli con show -a or nmcli c -a |
| nmcli device status            | nmcli dev or nmcli d            |
| nmcli device show              | nmcli dev show or nmcli d show  |

#### Additional resources

- For more information on the comprehensive list of **nmcli** options, see the ***nmcli(1)*** man page.
- For more examples, see the ***nmcli-examples(5)*** man page.
- [Section 47.5, “Creating a connection profile with nmcli”](#)

## 47.4. SETTING A DEVICE MANAGED OR UNMANAGED WITH NMCLI

### Prerequisites

- [Section 47.1, “Understanding nmcli”](#)
- [Section 47.2, “Overview of nmcli property names and aliases”](#)

### Procedure

1. To list the currently available network connections:

```
~]$ nmcli con show
NAME UUID TYPE DEVICE
Auto Ethernet 9b7f2511-5432-40ae-b091-af2457dfd988 802-3-ethernet --
ens3 fb157a65-ad32-47ed-858c-102a48e064a2 802-3-ethernet ens3
MyWiFi 91451385-4eb8-4080-8b82-720aab8328dd 802-11-wireless wlan0
```

Note that the **NAME** field in the output always denotes the **connection ID** (name). It is not the interface name even though it might look the same. In the second connection shown above, **ens3** in the **NAME** field is the **connection ID** given by the user to the profile applied to the interface **ens3**. In the last connection shown, the user has assigned the connection ID **MyWiFi** to the interface **wlan0**.

Adding an Ethernet connection means creating a configuration profile which is then assigned to a device. Before creating a new profile, review the available devices as follows:

```
~]$ nmcli device status
DEVICE TYPE STATE CONNECTION
ens3 ethernet disconnected --
ens9 ethernet disconnected --
lo loopback unmanaged --
```

2. To set the device unmanaged by the **NetworkManager**:

```
~]$ nmcli device set ifname managed no
```

For example, to set **enp1s0** unmanaged:

```
~]$ nmcli device status
DEVICE TYPE STATE CONNECTION
bond0 bond connected bond0
virbr0 bridge connected virbr0
enp7s0 ethernet connected bond-slave-enp7s0
enp1s0 ethernet connected bond-slave-enp1s0
enp8s0 ethernet unmanaged --
```

```
~]$ nmcli device set enp1s0 managed no
```

```
~]$ nmcli device status
```

| DEVICE | TYPE     | STATE     | CONNECTION        |
|--------|----------|-----------|-------------------|
| bond0  | bond     | connected | bond0             |
| virbr0 | bridge   | connected | virbr0            |
| enp7s0 | ethernet | connected | bond-slave-enp7s0 |
| enp1s0 | ethernet | unmanaged | --                |
| enp8s0 | ethernet | unmanaged | --                |



### NOTE

When you set the device unmanaged, **NetworkManager** does not control it. If the device you want to configure is listed as unmanaged, no **nmcli** command has any effect on this device. However, the device is still connected.

### Additional resources

- For more information, see the [nmcli\(1\)](#) man page.

## 47.5. CREATING A CONNECTION PROFILE WITH NMCLI

You can create a connection profile to be associated with a device.

### Prerequisites

- [Section 47.1, “Understanding nmcli”](#)
- [Section 47.2, “Overview of nmcli property names and aliases”](#)



### IMPORTANT

If you use the **nmcli** commands, it is recommended to type a partial **nmcli** command, and then press the **Tab** key to auto-complete the command sequence. If multiple completions are possible, then **Tab** lists them all. This helps users to type commands faster and easier. To enable the **nmcli** auto-complete feature be sure to install the **bash-completion** package:

```
~]# yum install bash-completion
```

After the package installation, **nmcli auto-complete** will be available next time you login into a console. To activate it also in the current console, enter:

```
~]$ source /etc/profile.d/bash_completion.sh
```

### Procedure

The basic format to **create** a new profile for **NetworkManager** using **nmcli**:

```
nmcli c add {COMMON_OPTIONS} [IP_OPTIONS]/[NETMASK] [GATEWAY]
```

- a. where **{COMMON\_OPTIONS}** are the aliases or property names, see [Aliases and Property names](#).
- b. where **[IP\_OPTIONS]** are the IP addresses:
  - For IPv4 addresses: **ip4**
  - For IPv6 addresses: **ip6**
- c. where **[NETMASK]** is the network mask width. For example, **255.255.255.0** is the network mask for the prefix **198.51.100.0/24**.
- d. where **[GATEWAY]** is the gateway information:
  - For IPv4 addresses: **gw4**
  - For IPv6 addresses: **gw6**

```
nmcli connection add type ethernet con-name connection-name ifname interface-name ip4
address/network mask gw4 address
```

1. To create a connection profile with an IPv4 address:

```
~]$ nmcli c add type ethernet ifname enp1s0 con-name "My Connection" ip4
192.168.2.100/24 gw4 192.168.2.1
Connection 'My Connection' (f0c23472-1aec-4e84-8f1b-be8a2ecbeade) successfully added.
```

2. To activate the created connection:

```
~]$ nmcli c up _"My Connection"
```

3. To view the created connection:

```
~]$ nmcli c show "My Connection"
```

Note that the **nmcli c showcon-name** command displays all the properties present in the connection, even those that are empty or have a default value. If the output is longer than a terminal page, **nmcli** generates a pager to allow an easy navigation on the output. In the pager, use arrows to move up and down and the **q** key to quit.

For a more compact display of the connection, use the **-o** option:

```
~]$ nmcli -o c show "My Connection"
```

The **nmcli -o c showcon-name** command still displays the connection content, but omits empty properties or those that are set to a default value. This usually results in a shorter output that is more readable.

#### Additional resources

- See the **nm-settings(5)** man page for more information on properties and their settings.

## 47.6. USING THE NMCLI INTERACTIVE CONNECTION EDITOR

The **nmcli** tool has an interactive connection editor. It allows you to change connection parameters according to your needs. To use it:

```
~]$ nmcli con edit
```

You should enter a valid **connection type** from the list displayed. Then, you are able to modify its parameters.

```
~]$ nmcli con edit
```

Valid connection types: generic, 802-3-ethernet (ethernet), pppoe, 802-11-wireless (wifi), wimax, gsm, cdma, infiniband, adsl, bluetooth, vpn, 802-11-olpc-mesh (olpc-mesh), wlan, bond, team, bridge, bond-slave, team-slave, bridge-slave, no-slave, tun, ip-tunnel, macsec, macvlan, vxlan, dummy  
Enter connection type: **ethernet**

====| nmcli interactive connection editor |====

Adding a new '802-11-wireless' connection

Type 'help' or '?' for available commands.

Type 'describe [<setting>.<prop>]' for detailed property description.

You may edit the following settings: connection, 802-11-wireless (wifi), 802-11-wireless-security (wifi-sec), 802-1x, ipv4, ipv6, proxy  
nmcli>

It is possible now to edit the **ethernet** connection settings. To get the list of available commands, type **help** or **?**:

```
nmcli> ?
```

---[ Main menu ]---

goto  [<setting> | <prop>]   :: go to a setting or property  
remove  <setting>[.<prop>] | <prop> :: remove setting or reset property value  
set  [<setting>.<prop> <value>] :: set property value  
describe [<setting>.<prop>]   :: describe property  
print  [all | <setting>[.<prop>]] :: print the connection  
verify  [all | fix]   :: verify the connection  
save  [persistent|temporary] :: save the connection  
activate [<ifname>] [/<ap>|<nsp>] :: activate the connection  
back   :: go one level up (back)  
help/?  [<command>]   :: print this help  
nmcli  <conf-option> <value>   :: nmcli configuration  
quit   :: exit nmcli

```
nmcli>
```

To exit, enter the **quit** command.

#### Example 47.13. Adding a new Ethernet connection using the nmcli interactive connection editor

```
~]$ nmcli con edit
```

Valid connection types: generic, 802-3-ethernet (ethernet), pppoe, 802-11-wireless (wifi), wimax, gsm, cdma, infiniband, adsl, bluetooth, vpn, 802-11-olpc-mesh (olpc-mesh), wlan, bond, team, bridge, bond-slave, team-slave, bridge-slave, no-slave, tun, ip-tunnel, macsec, macvlan, vxlan,

```

dummy
Enter connection type: ethernet

====| nmcli interactive connection editor |====

Adding a new '802-3-ethernet' connection

Type 'help' or '?' for available commands.
Type 'describe [<setting>.<prop>]' for detailed property description.

You may edit the following settings: connection, 802-3-ethernet (ethernet), 802-1x, dcb, ipv4, ipv6, proxy
nmcli> set connection.id new_enp7s0
nmcli> set connection.interface-name enp7s0
nmcli> set connection.autoconnect yes
nmcli> save
Saving the connection with 'autoconnect=yes'. That might result in an immediate activation of the connection.
Do you still want to save? (yes/no) [yes] yes
Connection 'new_enp7s0' (34ac8f9a-e9d8-4e0b-9751-d5dc87cc0467) successfully saved.
nmcli> quit

```

A new network interface configuration file is created in the **/etc/sysconfig/network-scripts** directory:

```

~]# ls -lrt /etc/sysconfig/network-scripts/ifcfg*
-rw-r--r--. 1 root root 254 Aug 15 2017 /etc/sysconfig/network-scripts/ifcfg-lo
-rw-r--r--. 1 root root 304 Apr 26 22:14 /etc/sysconfig/network-scripts/ifcfg-ens3
-rw-r--r--. 1 root root 266 Aug 6 11:03 /etc/sysconfig/network-scripts/ifcfg-new_enp7s0

```

## 47.7. MODIFYING A CONNECTION PROFILE WITH NMCLI

You can modify the existing configuration of a connection profile.

### Prerequisites

- [Section 47.1, “Understanding nmcli”](#)
- [Section 47.2, “Overview of nmcli property names and aliases”](#)
- [Section 47.5, “Creating a connection profile with nmcli”](#)



## IMPORTANT

If you use the **nmcli** commands, it is recommended to type a partial **nmcli** command, and then press the **Tab** key to auto-complete the command sequence. If multiple completions are possible, then **Tab** lists them all. This helps users to type commands faster and easier. To enable the **nmcli** auto-complete feature be sure to install the **bash-completion** package:

```
~]# yum install bash-completion
```

After the package installation, **nmcli auto-complete** will be available next time you login into a console. To activate it also in the current console, enter:

```
~]$ source /etc/profile.d/bash_completion.sh
```

## Procedure

1. To modify one or more properties of a connection profile, use the following command:

```
~]$ nmcli c modify
```

For example, to change the **connection.id** from "My Connection" to "My favorite connection" and the **connection.interface-name** to **enp7s0**:

```
~]$ nmcli c modify "My Connection" connection.id "My favorite connection"
connection.interface-name enp7s0
```

2. To **apply** changes after a modified connection using **nmcli**, activate again the connection by entering:

```
~]$ nmcli con up "My favorite connection"
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/16)
```

3. To view the modified connection, enter the **nmcli con show con-name** command.

# CHAPTER 48. GETTING STARTED WITH CONFIGURING NETWORKING USING THE GNOME GUI

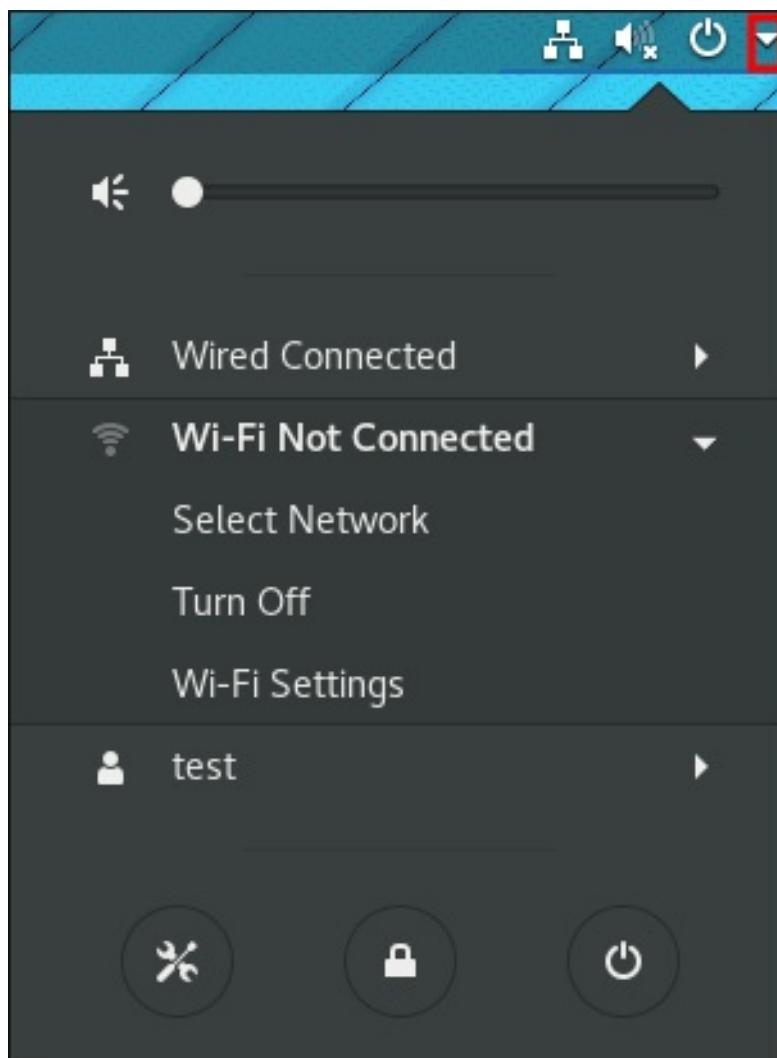
You can configure a network interface using the following **Graphical User Interface (GUI)** ways:

- the GNOME Shell **network connection icon** on the top right of the desktop
- the GNOME **control-center** application
- the GNOME **nm-connection-editor** application

## 48.1. CONNECTING TO A NETWORK USING THE GNOME SHELL NETWORK CONNECTION ICON

To access the **Network** settings, click on the GNOME Shell **network connection icon** in the top right-hand corner of the screen to open its menu:

Figure 48.1. The network connection icon menu



When you click on the GNOME Shell network connection icon, you can see:

- A list of categorized networks you are currently connected to (such as **Wired** and **Wi-Fi**).
- A list of all **Available Networks** that **NetworkManager** has detected. If you are connected to a network, this is indicated on the left of the connection name.

- Options for connecting to any configured Virtual Private Networks (VPNs) and
- An option for selecting the **Network Settings** menu entry.

## 48.2. CREATING A NETWORK CONNECTION USING CONTROL-CENTER

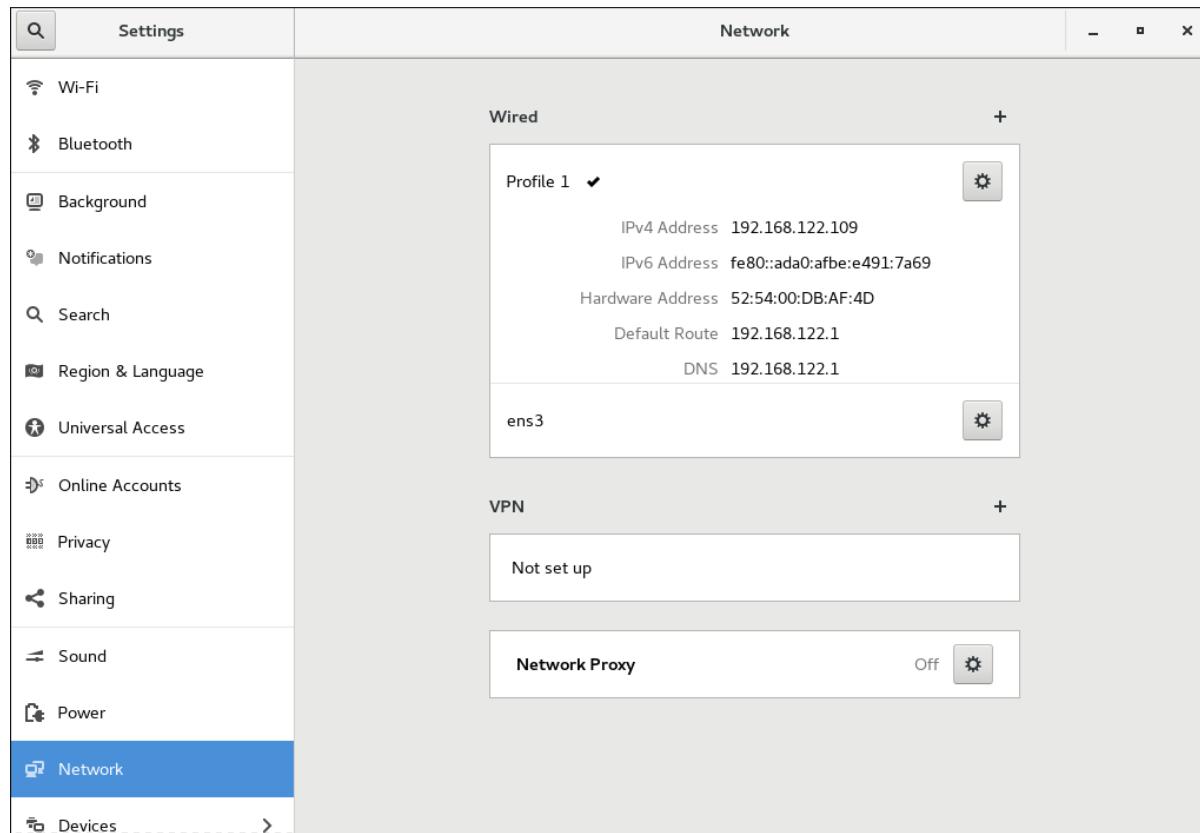
You can create a network connection through the GNOME **control-center** application, which is a graphical user interface that provides a view of available network devices and their current configuration.

This procedures describes how to create a new **wired**, **wireless**, **vpn** connection using **control-center**:

### Procedure

1. Press the **Super** key to enter the Activities Overview, type **Settings**, and press **Enter**. Then, select the **Network** tab on the left-hand side, and the **Network** settings tool appears:

**Figure 48.2. Opening the network settings window**



2. Click the plus button to add a new connection:
  - For **Wired** connections, click the plus button next to **Wired** entry and configure the connection.
  - For **VPN** connections, click the plus button next to **VPN** entry. If you want to add an **IPsec** **VPN**, click on **IPsec based VPN** and configure the connection.
  - For **Wi-Fi** connections, click the **Wi-Fi** entry on the left-hand side in the **Settings** menu and configure the connection.

# CHAPTER 49. CONFIGURING IP NETWORKING WITH IFCFG FILES

This section describes how to configure a network interface manually by editing the **ifcfg** files.

Interface configuration (ifcfg) files control the software interfaces for individual network devices. As the system boots, it uses these files to determine what interfaces to bring up and how to configure them. These files are usually named **ifcfg-name**, where the suffix *name* refers to the name of the device that the configuration file controls. By convention, the **ifcfg** file's suffix is the same as the string given by the **DEVICE** directive in the configuration file itself.

## 49.1. CONFIGURING AN INTERFACE WITH STATIC NETWORK SETTINGS USING IFCFG FILES

This procedure describes how to configure a network interface using **ifcfg** files.

### Procedure

To configure an interface with static network settings using **ifcfg** files, for an interface with the name **enp1s0**, create a file with the name **ifcfg-enp1s0** in the **/etc/sysconfig/network-scripts/** directory that contains:

- For **IPv4** configuration:

```
DEVICE=enp1s0
BOOTPROTO=none
ONBOOT=yes
PREFIX=24
IPADDR=10.0.1.27
GATEWAY=10.0.1.1
```

- For **IPv6** configuration:

```
DEVICE=enp1s0
BOOTPROTO=none
ONBOOT=yes
IPV6INIT=yes
IPV6ADDR=2001:db8::2/48
```

For more **IPv6** ifcfg configuration options, see **nm-settings-ifcfg-rh(5)** man page.

## 49.2. CONFIGURING AN INTERFACE WITH DYNAMIC NETWORK SETTINGS USING IFCFG FILES

This this procedure describes how to configure a network interface with dynamic network settings using **ifcfg** files.

### Procedure

1. To configure an interface named **em1** with dynamic network settings using **ifcfg** files, create a file with the name **ifcfg-em1** in the **/etc/sysconfig/network-scripts/** directory that contains:

```
DEVICE=em1
BOOTPROTO=dhcp
ONBOOT=yes
```

2. To configure an interface to send a different host name to the **DHCP** server, add the following line to the **ifcfg** file:

```
DHCP_HOSTNAME=hostname
```

3. To configure an interface to send a different fully qualified domain name (FQDN) to the **DHCP** server, add the following line to the **ifcfg** file:

```
DHCP_FQDN=fully.qualified.domain.name
```



#### NOTE

Only one directive, either **DHCP\_HOSTNAME** or **DHCP\_FQDN**, should be used in a given **ifcfg** file. In case both **DHCP\_HOSTNAME** and **DHCP\_FQDN** are specified, only the latter is used.

4. To configure an interface to use particular **DNS** servers, add the following lines to the **ifcfg** file:

```
PEERDNS=no
DNS1=ip-address
DNS2=ip-address
```

where *ip-address* is the address of a **DNS** server. This will cause the network service to update **/etc/resolv.conf** with the specified **DNS** servers specified. Only one **DNS** server address is necessary, the other is optional.

### 49.3. MANAGING SYSTEM-WIDE AND PRIVATE CONNECTION PROFILES WITH IFCFG FILES

This procedure describes how to configure **ifcfg** files to manage the system-wide and private connection profiles.

#### Procedure

The permissions correspond to the **USERS** directive in the **ifcfg** files. If the **USERS** directive is not present, the network profile will be available to all users.

1. As an example, modify the **ifcfg** file with the following row, which will make the connection available only to the users listed:

```
USERS="joe bob alice"
```

# CHAPTER 50. GETTING STARTED WITH IPVLAN

This document describes the IPVLAN driver.

## 50.1. IPVLAN OVERVIEW

IPVLAN is a driver for a virtual network device that can be used in container environment to access the host network. IPVLAN exposes a single MAC address to the external network regardless the number of IPVLAN device created inside the host network. This means that a user can have multiple IPVLAN devices in multiple containers and the corresponding switch reads a single MAC address. IPVLAN driver is useful when the local switch imposes constraints on the total number of MAC addresses that it can manage.

## 50.2. IPVLAN MODES

The following modes are available for IPVLAN:

- **L2 mode**

In IPVLAN **L2 mode**, virtual devices receive and respond to Address Resolution Protocol (ARP) requests. The **netfilter** framework runs only inside the container that owns the virtual device. No **netfilter** chains are executed in the default namespace on the containerized traffic. Using **L2 mode** provides good performance, but less control on the network traffic.

- **L3 mode**

In **L3 mode**, virtual devices process only **L3** traffic and above. Virtual devices do not respond to ARP request and users must configure the neighbour entries for the IPVLAN IP addresses on the relevant peers manually. The egress traffic of a relevant container is landed on the **netfilter** POSTROUTING and OUTPUT chains in the default namespace while the ingress traffic is threaded in the same way as **L2 mode**. Using **L3 mode** provides good control but decreases the network traffic performance.

- **L3S mode**

In **L3S mode**, virtual devices process the same way as in **L3 mode**, except that both egress and ingress traffics of a relevant container are landed on **netfilter** chain in the default namespace. **L3S mode** behaves in a similar way to **L3 mode** but provides greater control of the network.



### NOTE

The IPVLAN virtual device does not receive broadcast and multicast traffic in case of **L3** and **L3S modes**.

## 50.3. OVERVIEW OF MACVLAN

The MACVLAN driver allows to create multiple virtual network devices on top of a single NIC, each of them identified by its own unique MAC address. Packets which land on the physical NIC are demultiplexed towards the relevant MACVLAN device via MAC address of the destination. MACVLAN devices do not add any level of encapsulation.

## 50.4. COMPARISON OF IPVLAN AND MACVLAN

The following table shows the major differences between MACVLAN and IPVLAN.

| MACVLAN                                                                                                                               | IPVLAN                                                                                             |
|---------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------|
| Uses MAC address for each MACVLAN device. The overlimit of MAC addresses of MAC table in switch might cause loosing the connectivity. | Uses single MAC address which does not limit the number of IPVLAN devices.                         |
| Netfilter rules for global namespace cannot affect traffic to or from MACVLAN device in a child namespace.                            | It is possible to control traffic to or from IPVLAN device in <b>L3 mode</b> and <b>L3S mode</b> . |

Note that both IPVLAN and MACVLAN do not require any level of encapsulation.

## 50.5. CONFIGURING IPVLAN NETWORK

### 50.5.1. Creating and configuring the IPVLAN device using iproute2

This procedure shows how to set up the IPVLAN device using iproute2.

#### Procedure

1. To create an IPVLAN device, enter the following command:

```
~]# ip link add link real_NIC_device name IPVLAN_device type ipvlan mode l2
```

Note that network interface controller (NIC) is a hardware component which connects a computer to a network.

#### Example 50.1. Creating an IPVLAN device

```
~]# ip link add link enp0s31f6 name my_ipvlan type ipvlan mode l2
~]# ip link
47: my_ipvlan@enp0s31f6: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT group default qlen 1000 link/ether e8:6a:6e:8a:a2:44 brd ff:ff:ff:ff:ff:ff
```

2. To assign an **IPv4** or **IPv6** address to the interface, enter the following command:

```
~]# ip addr add dev IPVLAN_device IP_address/subnet_mask_prefix
```

3. In case of configuring an IPVLAN device in **L3 mode** or **L3S mode**, make the following setups:

- a. Configure the neighbor setup for the remote peer on the remote host:

```
~]# ip neigh add dev peer_device IPVLAN_device_IP_address lladdr MAC_address
```

where *MAC\_address* is the MAC address of the real NIC on which an IPVLAN device is based on.

- b. Configure an IPVLAN device for **L3 mode** with the following command:

```
~]# ip neigh add dev real_NIC_device peer_IP_address lladdr peer_MAC_address
```

For **L3S mode**:

```
~]# ip route dev add real_NIC_device peer_IP_address/32
```

where *IP-address* represents the address of the remote peer.

4. To set an IPVLAN device active, enter the following command:

```
~]# ip link set dev IPVLAN_device up
```

5. To check if the IPVLAN device is active, execute the following command on the remote host:

```
~]# ping IP_address
```

where the *IP\_address* uses the IP address of the IPVLAN device.

# CHAPTER 51. CONFIGURING VIRTUAL ROUTING AND FORWARDING (VRF)

With Virtual routing and forwarding (VRF), Administrators can use multiple routing tables simultaneously on the same host. For that, VRF partitions a network at layer 3. This enables the administrator to isolate traffic using separate and independent route tables per VRF domain. This technique is similar to virtual LANs (VLAN), which partitions a network at layer 2, where the operating system uses different VLAN tags to isolate traffic sharing the same physical medium.

One benefit of VRF over partitioning on layer 2 is that routing scales better considering the number of peers involved.

Red Hat Enterprise Linux uses a virtual **virt** device for each VRF domain and adds routes to a VRF domain by enslaving existing network devices to a VRF device. Addresses and routes previously attached to the enslaved device will be moved inside the VRF domain.

Note that each VRF domain is isolated from each other.

## 51.1. TEMPORARILY REUSING THE SAME IP ADDRESS ON DIFFERENT INTERFACES

The procedure in this section describes how to temporarily use the same IP address on different interfaces in one server by using the virtual routing and forwarding (VRF) feature. Use this procedure only for testing purposes, because the configuration is temporary and lost after you reboot the system.



### IMPORTANT

To enable remote peers to contact both VRF interfaces while reusing the same IP address, the network interfaces must belong to different broadcasting domains. A broadcast domain in a network is a set of nodes which receive broadcast traffic sent by any of them. In most configurations, all nodes connected to the same switch belong to the same broadcasting domain.

#### Prerequisites

- You are logged in as the **root** user.
- The network interfaces are not configured.

#### Procedure

1. Create and configure the first VRF device:
  - a. Create the VRF device and assign it to a routing table. For example, to create a VRF device named **blue** that is assigned to the **1001** routing table:

```
ip link add dev blue type vrf table 1001
```

- b. Enable the **blue** device:

```
ip link set dev blue up
```

- c. Assign a network device to the VRF device. For example, to add the **enp1s0** Ethernet device to the **blue** VRF device:

```
ip link set dev enp1s0 master blue
```

- d. Enable the **enp1s0** device:

```
ip link set dev enp1s0 up
```

- e. Assign an IP address and subnet mask to the **enp1s0** device. For example, to set it to **192.0.2.1/24**:

```
ip addr add dev enp1s0 192.0.2.1/24
```

2. Create and configure the next VRF device:

- a. Create the VRF device and assign it to a routing table. For example, to create a VRF device named **red** that is assigned to the **1002** routing table:

```
ip link add dev red type vrf table 1002
```

- b. Enable the **red** device:

```
ip link set dev red up
```

- c. Assign a network device to the VRF device. For example, to add the **enp7s0** Ethernet device to the **red** VRF device:

```
ip link set dev enp7s0 master red
```

- d. Enable the **enp7s0** device:

```
ip link set dev enp7s0 up
```

- e. Assign the same IP address and subnet mask to the **enp7s0** device as you used for **enp1s0** in the **blue** VRF domain:

```
ip addr add dev enp7s0 192.0.2.1/24
```

3. Optionally, create further VRF devices as described above.

## 51.2. RELATED INFORMATION

- <https://www.kernel.org/doc/Documentation/networking/vrf.txt>

## CHAPTER 52. SECURING NETWORKS

# CHAPTER 53. USING SECURE COMMUNICATIONS BETWEEN TWO SYSTEMS WITH OPENSSH

SSH (Secure Shell) is a protocol which provides secure communications between two systems using a client-server architecture and allows users to log in to server host systems remotely. Unlike other remote communication protocols, such as FTP or Telnet, SSH encrypts the login session, which prevents intruders to collect unencrypted passwords from the connection.

Red Hat Enterprise Linux includes the basic **OpenSSH** packages: the general **openssh** package, the **openssh-server** package and the **openssh-clients** package. Note that the **OpenSSH** packages require the **OpenSSL** package **openssl-libs**, which installs several important cryptographic libraries that enable **OpenSSH** to provide encrypted communications.

## 53.1. SSH AND OPENSSH

SSH (Secure Shell) is a program for logging into a remote machine and executing commands on that machine. The SSH protocol provides secure encrypted communications between two untrusted hosts over an insecure network. You can also forward X11 connections and arbitrary TCP/IP ports over the secure channel.

The SSH protocol mitigates security threats, such as interception of communication between two systems and impersonation of a particular host, when you use it for remote shell login or file copying. This is because the SSH client and server use digital signatures to verify their identities. Additionally, all communication between the client and server systems is encrypted.

**OpenSSH** is an implementation of the SSH protocol supported by a number of Linux, UNIX, and similar operating systems. It includes the core files necessary for both the OpenSSH client and server. The OpenSSH suite consists of the following user-space tools:

- **ssh** is a remote login program (SSH client)
- **sshd** is an **OpenSSH** SSH daemon
- **scp** is a secure remote file copy program
- **sftp** is a secure file transfer program
- **ssh-agent** is an authentication agent for caching private keys
- **ssh-add** adds private key identities to **ssh-agent**
- **ssh-keygen** generates, manages, and converts authentication keys for **ssh**
- **ssh-copy-id** is a script that adds local public keys to the **authorized\_keys** file on a remote SSH server
- **ssh-keyscan** - gathers SSH public host keys

Two versions of SSH currently exist: version 1, and the newer version 2. The **OpenSSH** suite in Red Hat Enterprise Linux 8 supports only SSH version 2, which has an enhanced key-exchange algorithm not vulnerable to known exploits in version 1.

**OpenSSH**, as one of the RHEL core cryptographic subsystems uses system-wide crypto policies. This ensures that weak cipher suites and cryptographic algorithms are disabled in the default configuration. To adjust the policy, the administrator must either use the **update-crypto-policies** command to make settings stricter or looser or manually opt-out of the system-wide crypto policies.

The **OpenSSH** suite uses two different sets of configuration files: those for client programs (that is, **ssh**, **scp**, and **sftp**), and those for the server (the **sshd** daemon). System-wide SSH configuration information is stored in the **/etc/ssh/** directory. User-specific SSH configuration information is stored in **~/.ssh/** in the user's home directory. For a detailed list of OpenSSH configuration files, see the **FILES** section in the **sshd(8)** man page.

## Additional resources

- Man pages for the **ssh** topic listed by the **man -k ssh** command.
- [Using system-wide cryptographic policies](#).

## 53.2. CONFIGURING AND STARTING AN OPENSSH SERVER

Use the following procedure for a basic configuration that might be required for your environment and for starting an **OpenSSH** server. Note that after the default RHEL installation, the **sshd** daemon is already started and server host keys are automatically created.

### Prerequisites

- The **openssh-server** package is installed.

### Procedure

1. Start the **sshd** daemon in the current session and set it to start automatically at boot time:

```
systemctl start sshd
systemctl enable sshd
```

2. To specify different addresses than the default **0.0.0.0** (IPv4) or **::** (IPv6) for the **ListenAddress** directive in the **/etc/ssh/sshd\_config** configuration file and to use a slower dynamic network configuration, add the dependency on the **network-online.target** target unit to the **sshd.service** unit file. To achieve this, create the **/etc/systemd/system/sshd.service.d/local.conf** file with the following content:

```
[Unit]
Wants=network-online.target
After=network-online.target
```

3. Review if **OpenSSH** server settings in the **/etc/ssh/sshd\_config** configuration file meet the requirements of your scenario.
4. Optionally, change the welcome message that your **OpenSSH** server displays before a client authenticates by editing the **/etc/issue** file, for example:

```
Welcome to ssh-server.example.com
Warning: By accessing this server, you agree to the referenced terms and conditions.
```

Note that to change the message displayed after a successful login you have to edit the **/etc/motd** file on the server. See the **pam\_motd** man page for more information.

5. Reload the **systemd** configuration to apply the changes:

```
systemctl daemon-reload
```

## Verification steps

1. Check that the **sshd** daemon is running:

```
systemctl status sshd
● sshd.service - OpenSSH server daemon
 Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enabled)
 Active: active (running) since Mon 2019-11-18 14:59:58 CET; 6min ago
 Docs: man:sshd(8)
 man:sshd_config(5)
 Main PID: 1149 (sshd)
 Tasks: 1 (limit: 11491)
 Memory: 1.9M
 CGroup: /system.slice/sshd.service
 └─1149 /usr/sbin/sshd -D -oCiphers=aes128-ctr,aes256-ctr,aes128-cbc,aes256-cbc -oMACs=hmac-sha2-256,>

Nov 18 14:59:58 ssh-server-example.com systemd[1]: Starting OpenSSH server daemon...
Nov 18 14:59:58 ssh-server-example.com sshd[1149]: Server listening on 0.0.0.0 port 22.
Nov 18 14:59:58 ssh-server-example.com sshd[1149]: Server listening on :: port 22.
Nov 18 14:59:58 ssh-server-example.com systemd[1]: Started OpenSSH server daemon.
```

2. Connect to the SSH server with an SSH client.

```
ssh user@ssh-server-example.com
ECDSA key fingerprint is SHA256:dXbaS0RG/UzITTku8GtXSz0S1++lPegSy31v3L/FAEc.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ssh-server-example.com' (ECDSA) to the list of known hosts.

user@ssh-server-example.com's password:
```

## Additional resources

- **sshd(8)** and **sshd\_config(5)** man pages

## 53.3. USING KEY PAIRS INSTEAD OF PASSWORDS FOR SSH AUTHENTICATION

To improve system security even further, generate SSH key pairs and then enforce key-based authentication by disabling password authentication.

### 53.3.1. Setting an OpenSSH server for key-based authentication

Follow these steps to configure your OpenSSH server for enforcing key-based authentication.

#### Prerequisites

- The **openssh-server** package is installed.
- The **sshd** daemon is running on the server.

#### Procedure

1. Open the **/etc/ssh/sshd\_config** configuration in a text editor, for example:

```
vi /etc/ssh/sshd_config
```

2. Change the **PasswordAuthentication** option to **no**:

```
 PasswordAuthentication no
```

On a system other than a new default installation, check that **PubkeyAuthentication no** has not been set and the **ChallengeResponseAuthentication** directive is set to **no**. If you are connected remotely, not using console or out-of-band access, test the key-based login process before disabling password authentication.

3. To use key-based authentication with NFS-mounted home directories, enable the **use\_nfs\_home\_dirs** SELinux boolean:

```
setsebool -P use_nfs_home_dirs 1
```

4. Reload the **sshd** daemon to apply the changes:

```
systemctl reload sshd
```

#### Additional resources

- **sshd(8)**, **sshd\_config(5)**, and **setsebool(8)** man pages

### 53.3.2. Generating SSH key pairs

Use this procedure to generate an SSH key pair on a local system and to copy the generated public key to an **OpenSSH** server. If the server is configured accordingly, you can log in to the **OpenSSH** server without providing any password.



#### IMPORTANT

If you complete the following steps as **root**, only **root** is able to use the keys.

#### Procedure

1. To generate an ECDSA key pair for version 2 of the SSH protocol:

```
$ ssh-keygen -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/home/joesec/.ssh/id_ecdsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/joesec/.ssh/id_ecdsa.
Your public key has been saved in /home/joesec/.ssh/id_ecdsa.pub.
The key fingerprint is:
SHA256:Q/x+qms4j7PCQ0qFd09iZEFHA+SqwBKRNaU72oZfaCI
joesec@localhost.example.com
The key's randomart image is:
+---[ECDSA 256]---+
|..00..0=++ |
|.. 0 .00 . |
|.. 0. 0 |
```

```

|....0+... |
|0.oo.o +S . |
|.=+. .0 |
|E.*+. . . |
|.=..+ +.. 0 |
| . oo*+o. |
+---[SHA256]----+

```

You can also generate an RSA key pair by using the **-t rsa** option with the **ssh-keygen** command or an Ed25519 key pair by entering the **ssh-keygen -t ed25519** command.

2. To copy the public key to a remote machine:

```

$ ssh-copy-id joesec@ssh-server-example.com
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are
already installed
...
Number of key(s) added: 1

```

Now try logging into the machine, with: "ssh 'joesec@ssh-server-example.com'" and check to make sure that only the key(s) you wanted were added.

If you do not use the **ssh-agent** program in your session, the previous command copies the most recently modified **~/.ssh/id\*.pub** public key if it is not yet installed. To specify another public-key file or to prioritize keys in files over keys cached in memory by **ssh-agent**, use the **ssh-copy-id** command with the **-i** option.



#### NOTE

If you reinstall your system and want to keep previously generated key pairs, back up the **~/.ssh/** directory. After reinstalling, copy it back to your home directory. You can do this for all users on your system, including **root**.

#### Verification steps

1. Log in to the OpenSSH server without providing any password:

```

$ ssh joesec@ssh-server-example.com
Welcome message.
...
Last login: Mon Nov 18 18:28:42 2019 from ::1

```

#### Additional resources

- **ssh-keygen(1)** and **ssh-copy-id(1)** man pages

## 53.4. USING SSH KEYS STORED ON A SMART CARD

Red Hat Enterprise Linux 8 enables you to use RSA and ECDSA keys stored on a smart card on OpenSSH clients. Use this procedure to enable authentication using a smart card instead of using a password.

#### Prerequisites

- On the client side, the **opensc** package is installed and the **pcscd** service is running.

## Procedure

- List all keys provided by the OpenSC PKCS #11 module including their PKCS #11 URIs and save the output to the *keys.pub* file:

```
$ ssh-keygen -D pkcs11: > keys.pub
$ ssh-keygen -D pkcs11:
ssh-rsa AAAAB3NzaC1yc2E...KKZMzcQZzx
pkcs11:id=%02;object=SIGN%20pubkey;token=SSH%20key;manufacturer=piv_II?module-
path=/usr/lib64/pkcs11/opensc-pkcs11.so
ecdsa-sha2-nistp256 AAA...J0hkYnnsM=
pkcs11:id=%01;object=PIV%20AUTH%20pubkey;token=SSH%20key;manufacturer=piv_II?
module-path=/usr/lib64/pkcs11/opensc-pkcs11.so
```

- To enable authentication using a smart card on a remote server (*example.com*), transfer the public key to the remote server. Use the **ssh-copy-id** command with *keys.pub* created in the previous step:

```
$ ssh-copy-id -f -i keys.pub username@example.com
```

- To connect to *example.com* using the ECDSA key from the output of the **ssh-keygen -D** command in step 1, you can use just a subset of the URI, which uniquely references your key, for example:

```
$ ssh -i "pkcs11:id=%01?module-path=/usr/lib64/pkcs11/opensc-pkcs11.so" example.com
Enter PIN for 'SSH key':
[example.com] $
```

- You can use the same URI string in the *~/.ssh/config* file to make the configuration permanent:

```
$ cat ~/.ssh/config
IdentityFile "pkcs11:id=%01?module-path=/usr/lib64/pkcs11/opensc-pkcs11.so"
$ ssh example.com
Enter PIN for 'SSH key':
[example.com] $
```

Because OpenSSH uses the **p11-kit-proxy** wrapper and the OpenSC PKCS #11 module is registered to PKCS#11 Kit, you can simplify the previous commands:

```
$ ssh -i "pkcs11:id=%01" example.com
Enter PIN for 'SSH key':
[example.com] $
```

If you skip the **id=** part of a PKCS #11 URI, OpenSSH loads all keys that are available in the proxy module. This can reduce the amount of typing required:

```
$ ssh -i pkcs11: example.com
Enter PIN for 'SSH key':
[example.com] $
```

## Additional resources

- [Fedora 28: Better smart card support in OpenSSH](#)
- **p11-kit(8)** man page
- **ssh(1)** man page
- **ssh-keygen(1)** man page
- **opensc.conf(5)** man page
- **pcscd(8)** man page

## 53.5. MAKING OPENSSH MORE SECURE

The following tips help you to increase security when using OpenSSH. Note that changes in the **/etc/ssh/sshd\_config** OpenSSH configuration file require reloading the **sshd** daemon to take effect:

```
systemctl reload sshd
```



### IMPORTANT

The majority of security hardening configuration changes reduce compatibility with clients that do not support up-to-date algorithms or cipher suites.

#### Disabling insecure connection protocols

- To make SSH truly effective, prevent the use of insecure connection protocols that are replaced by the **OpenSSH** suite. Otherwise, a user's password might be protected using SSH for one session only to be captured later when logging in using Telnet. For this reason, consider disabling insecure protocols, such as telnet, rsh, rlogin, and ftp.

#### Enabling key-based authentication and disabling password-based authentication

- Disabling passwords for authentication and allowing only key pairs reduces the attack surface and it also might save users' time. On clients, generate key pairs using the **ssh-keygen** tool and use the **ssh-copy-id** utility to copy public keys from clients on the **OpenSSH** server. To disable password-based authentication on your OpenSSH server, edit **/etc/ssh/sshd\_config** and change the **PasswordAuthentication** option to **no**:

```
PasswordAuthentication no
```

#### Key types

- Although the **ssh-keygen** command generates a pair of RSA keys by default, you can instruct it to generate ECDSA or Ed25519 keys by using the **-t** option. The ECDSA (Elliptic Curve Digital Signature Algorithm) offers better performance than RSA at the equivalent symmetric key strength. It also generates shorter keys. The Ed25519 public-key algorithm is an implementation of twisted Edwards curves that is more secure and also faster than RSA, DSA, and ECDSA. OpenSSH creates RSA, ECDSA, and Ed25519 server host keys automatically if they are missing. To configure the host key creation in RHEL 8, use the **sshd-keygen@.service** instantiated service. For example, to disable the automatic creation of the RSA key type:

```
systemctl mask sshd-keygen@rsa.service
```

- To exclude particular key types for SSH connections, comment out the relevant lines in **/etc/ssh/sshd\_config**, and reload the **sshd** service. For example, to allow only Ed25519 host keys:

```
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
```

### Non-default port

- By default, the **sshd** daemon listens on TCP port 22. Changing the port reduces the exposure of the system to attacks based on automated network scanning and thus increase security through obscurity. You can specify the port using the **Port** directive in the **/etc/ssh/sshd\_config** configuration file. You also have to update the default SELinux policy to allow the use of a non-default port. To do so, use the **semanage** tool from the **policycoreutils-python-utils** package:

```
semanage port -a -t ssh_port_t -p tcp port_number
```

Furthermore, update **firewalld** configuration:

```
firewall-cmd --add-port port_number/tcp
firewall-cmd --runtime-to-permanent
```

In the previous commands, replace *port\_number* with the new port number specified using the **Port** directive.

### No root login

- If your particular use case does not require the possibility of logging in as the root user, you should consider setting the **PermitRootLogin** configuration directive to **no** in the **/etc/ssh/sshd\_config** file. By disabling the possibility of logging in as the root user, the administrator can audit which users run what privileged commands after they log in as regular users and then gain root rights.

Alternatively, set **PermitRootLogin** to **prohibit-password**:

```
PermitRootLogin prohibit-password
```

This enforces the use of key-based authentication instead of the use of passwords for logging in as root and reduces risks by preventing brute-force attacks.

### Using the X Security extension

- The X server in Red Hat Enterprise Linux clients does not provide the X Security extension. Therefore, clients cannot request another security layer when connecting to untrusted SSH servers with X11 forwarding. Most applications are not able to run with this extension enabled anyway.

By default, the **ForwardX11Trusted** option in the **/etc/ssh/ssh\_config.d/05-redhat.conf** file is set to **yes**, and there is no difference between the **ssh -X remote\_machine** (untrusted host) and **ssh -Y remote\_machine** (trusted host) command.

If your scenario does not require the X11 forwarding feature at all, set the **X11Forwarding** directive in the **/etc/ssh/sshd\_config** configuration file to **no**.

## Restricting access to specific users, groups, or domains

- The **AllowUsers** and **AllowGroups** directives in the **/etc/ssh/sshd\_config** configuration file server enable you to permit only certain users, domains, or groups to connect to your OpenSSH server. You can combine **AllowUsers** and **AllowGroups** to restrict access more precisely, for example:

```
AllowUsers *@192.168.1.*,@10.0.0.*,!@192.168.1.2
AllowGroups example-group
```

The previous configuration lines accept connections from all users from systems in 192.168.1.\* and 10.0.0.\* subnets except from the system with the 192.168.1.2 address. All users must be in the **example-group** group. The OpenSSH server denies all other connections.

Note that using whitelists (directives starting with **Allow**) is more secure than using blacklists (options starting with **Deny**) because whitelists block also new unauthorized users or groups.

## Changing system-wide cryptographic policies

- OpenSSH** uses RHEL system-wide cryptographic policies, and the default system-wide cryptographic policy level offers secure settings for current threat models. To make your cryptographic settings more strict, change the current policy level:

```
update-crypto-policies --set FUTURE
Setting system policy to FUTURE
```

- To opt-out of the system-wide crypto policies for your **OpenSSH** server, uncomment the line with the **CRYPTO\_POLICY=** variable in the **/etc/sysconfig/sshd** file. After this change, values that you specify in the **Ciphers**, **MACs**, **KexAlgorithms**, and **GSSAPIKexAlgorithms** sections in the **/etc/ssh/sshd\_config** file are not overridden. Note that this task requires deep expertise in configuring cryptographic options.
- See [Using system-wide cryptographic policies](#) in the [RHEL 8 Security hardening](#) title for more information.

## Additional resources

- sshd\_config(5)**, **ssh-keygen(1)**, **crypto-policies(7)**, and **update-crypto-policies(8)** man pages

## 53.6. CONNECTING TO A REMOTE SERVER USING AN SSH JUMP HOST

Use this procedure for connecting to a remote server through an intermediary server, also called jump host.

### Prerequisites

- A jump host accepts SSH connections from your system.
- A remote server accepts SSH connections only from the jump host.

### Procedure

- Define the jump host by editing the **~/.ssh/config** file, for example:

```
Host jump-server1
HostName jump1.example.com
```

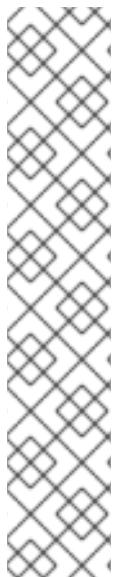
2. Add the remote server jump configuration with the **ProxyJump** directive to `~/.ssh/config`, for example:

```
Host remote-server
HostName remote1.example.com
ProxyJump jump-server1
```

3. Connect to the remote server through the jump server:

```
$ ssh remote-server
```

The previous command is equivalent to the `ssh -J jump-server1 remote-server` command if you omit the configuration steps 1 and 2.



#### NOTE

You can specify more jump servers and you can also skip adding host definitions to the configurations file when you provide their complete host names, for example:

```
$ ssh -J jump1.example.com,jump2.example.com,jump3.example.com
remote1.example.com
```

Change the host name-only notation in the previous command if the user names or SSH ports on the jump servers differ from the names and ports on the remote server, for example:

```
$ ssh -J
johndoe@jump1.example.com:75,johndoe@jump2.example.com:75,johndoe@jump3.e
xample.com:75 joesec@remote1.example.com:220
```

#### Additional resources

- **ssh\_config(5)** and **ssh(1)** man pages

### 53.7. ADDITIONAL RESOURCES

For more information on configuring and connecting to **OpenSSH** servers and clients on Red Hat Enterprise Linux, see the resources listed below.

#### Installed documentation

- **sshd(8)** man page documents available command-line options and provides a complete list of supported configuration files and directories.
- **ssh(1)** man page provides a complete list of available command-line options and supported configuration files and directories.
- **scp(1)** man page provides a more detailed description of the **scp** utility and its usage.
- **sftp(1)** man page provides a more detailed description of the **sftp** utility and its usage.

- **ssh-keygen(1)** man page documents in detail the use of the **ssh-keygen** utility to generate, manage, and convert authentication keys used by ssh.
- **ssh-copy-id(1)** man page describes the use of the **ssh-copy-id** script.
- **ssh\_config(5)** man page documents available SSH client configuration options.
- **sshd\_config(5)** man page provides a full description of available SSH daemon configuration options.
- **update-crypto-policies(8)** man page provides guidance on managing system-wide cryptographic policies
- **crypto-policies(7)** man page provides an overview of system-wide cryptographic policy levels

### Online documentation

- [OpenSSH Home Page](#) – contains further documentation, frequently asked questions, links to the mailing lists, bug reports, and other useful resources.
- [Configuring SELinux for applications and services with non-standard configurations](#) – you can apply analogous procedures for OpenSSH in a non-standard configuration with SELinux in enforcing mode.
- [Controlling network traffic using firewalld](#) – provides guidance on updating **firewalld** settings after changing an SSH port

# CHAPTER 54. PLANNING AND IMPLEMENTING TLS

TLS (Transport Layer Security) is a cryptographic protocol used to secure network communications. When hardening system security settings by configuring preferred key-exchange protocols, authentication methods, and encryption algorithms, it is necessary to bear in mind that the broader the range of supported clients, the lower the resulting security. Conversely, strict security settings lead to limited compatibility with clients, which can result in some users being locked out of the system. Be sure to target the strictest available configuration and only relax it when it is required for compatibility reasons.

## 54.1. SSL AND TLS PROTOCOLS

The Secure Sockets Layer (SSL) protocol was originally developed by Netscape Corporation to provide a mechanism for secure communication over the Internet. Subsequently, the protocol was adopted by the Internet Engineering Task Force (IETF) and renamed to Transport Layer Security (TLS).

The TLS protocol sits between an application protocol layer and a reliable transport layer, such as TCP/IP. It is independent of the application protocol and can thus be layered underneath many different protocols, for example: HTTP, FTP, SMTP, and so on.

| Protocol version | Usage recommendation                                                                                                                                                                                                                                                                      |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SSL v2           | Do not use. Has serious security vulnerabilities. Removed from the core crypto libraries since RHEL 7.                                                                                                                                                                                    |
| SSL v3           | Do not use. Has serious security vulnerabilities. Removed from the core crypto libraries since RHEL 8.                                                                                                                                                                                    |
| TLS 1.0          | Not recommended to use. Has known issues that cannot be mitigated in a way that guarantees interoperability, and does not support modern cipher suites. Enabled only in the <b>LEGACY</b> system-wide cryptographic policy profile.                                                       |
| TLS 1.1          | Use for interoperability purposes where needed. Does not support modern cipher suites. Enabled only in the <b>LEGACY</b> policy.                                                                                                                                                          |
| TLS 1.2          | Supports the modern AEAD cipher suites. This version is enabled in all system-wide crypto policies, but optional parts of this protocol contain vulnerabilities and TLS 1.2 also allows outdated algorithms.                                                                              |
| TLS 1.3          | Recommended version. TLS 1.3 removes known problematic options, provides additional privacy by encrypting more of the negotiation handshake and can be faster thanks usage of more efficient modern cryptographic algorithms. TLS 1.3 is also enabled in all system-wide crypto policies. |

### Additional resources

- [IETF: The Transport Layer Security \(TLS\) Protocol Version 1.3](#)

## 54.2. SECURITY CONSIDERATIONS FOR TLS IN RHEL 8

In RHEL 8, cryptography-related considerations are significantly simplified thanks to the system-wide

crypto policies. The **DEFAULT** crypto policy allows only TLS 1.2 and 1.3. To allow your system to negotiate connections using the earlier versions of TLS, you need to either opt out from following crypto policies in an application or switch to the **LEGACY** policy with the **update-crypto-policies** command. See [Using system-wide cryptographic policies](#) for more information.

The default settings provided by libraries included in RHEL 8 are secure enough for most deployments. The TLS implementations use secure algorithms where possible while not preventing connections from or to legacy clients or servers. Apply hardened settings in environments with strict security requirements where legacy clients or servers that do not support secure algorithms or protocols are not expected or allowed to connect.

The most straightforward way to harden your TLS configuration is switching the system-wide cryptographic policy level to **FUTURE** using the **update-crypto-policies --set FUTURE** command.

If you decide to not follow RHEL system-wide crypto policies, use the following recommendations for preferred protocols, cipher suites, and key lengths on your custom configuration:

### 54.2.1. Protocols

The latest version of TLS provides the best security mechanism. Unless you have a compelling reason to include support for older versions of TLS, allow your systems to negotiate connections using at least TLS version 1.2. Note that despite that RHEL 8 supports TLS version 1.3, not all features of this protocol are fully supported by RHEL 8 components. For example, the 0-RTT (Zero Round Trip Time) feature, which reduces connection latency, is not yet fully supported by Apache or Nginx web servers.

### 54.2.2. Cipher suites

Modern, more secure cipher suites should be preferred to old, insecure ones. Always disable the use of eNULL and aNULL cipher suites, which do not offer any encryption or authentication at all. If at all possible, ciphers suites based on RC4 or HMAC-MD5, which have serious shortcomings, should also be disabled. The same applies to the so-called export cipher suites, which have been intentionally made weaker, and thus are easy to break.

While not immediately insecure, cipher suites that offer less than 128 bits of security should not be considered for their short useful life. Algorithms that use 128 bits of security or more can be expected to be unbreakable for at least several years, and are thus strongly recommended. Note that while 3DES ciphers advertise the use of 168 bits, they actually offer 112 bits of security.

Always give preference to cipher suites that support (perfect) forward secrecy (PFS), which ensures the confidentiality of encrypted data even in case the server key is compromised. This rules out the fast RSA key exchange, but allows for the use of ECDHE and DHE. Of the two, ECDHE is the faster and therefore the preferred choice.

You should also give preference to AEAD ciphers, such as AES-GCM, before CBC-mode ciphers as they are not vulnerable to padding oracle attacks. Additionally, in many cases, AES-GCM is faster than AES in CBC mode, especially when the hardware has cryptographic accelerators for AES.

Note also that when using the ECDHE key exchange with ECDSA certificates, the transaction is even faster than pure RSA key exchange. To provide support for legacy clients, you can install two pairs of certificates and keys on a server: one with ECDSA keys (for new clients) and one with RSA keys (for legacy ones).

### 54.2.3. Public key length

When using RSA keys, always prefer key lengths of at least 3072 bits signed by at least SHA-256, which is sufficiently large for true 128 bits of security.



### WARNING

The security of your system is only as strong as the weakest link in the chain. For example, a strong cipher alone does not guarantee good security. The keys and the certificates are just as important, as well as the hash functions and keys used by the Certification Authority (CA) to sign your keys.

### Additional resources

- [System-wide crypto policies in RHEL 8](#) .
- **update-crypto-policies(8)** man page

## 54.3. HARDENING TLS CONFIGURATION IN APPLICATIONS

In Red Hat Enterprise Linux 8, [system-wide crypto policies](#) provide a convenient way to ensure that your applications using cryptographic libraries do not allow known insecure protocols, ciphers, or algorithms.

If you want to harden your TLS-related configuration with your customized cryptographic settings, you can use the cryptographic configuration options described in this section, and override the system-wide crypto policies just in the minimum required amount.

Regardless of the configuration you choose to use, always make sure to mandate that your server application enforces *server-side cipher order*, so that the cipher suite to be used is determined by the order you configure.

### 54.3.1. Configuring the Apache HTTP server

The **Apache HTTP Server** can use both **OpenSSL** and **NSS** libraries for its TLS needs. Red Hat Enterprise Linux 8 provides the **mod\_ssl** functionality through eponymous packages:

```
yum install mod_ssl
```

The **mod\_ssl** package installs the **/etc/httpd/conf.d/ssl.conf** configuration file, which can be used to modify the TLS-related settings of the **Apache HTTP Server**.

Install the **httpd-manual** package to obtain complete documentation for the **Apache HTTP Server**, including TLS configuration. The directives available in the **/etc/httpd/conf.d/ssl.conf** configuration file are described in detail in [/usr/share/httpd/manual/mod/mod\\_ssl.html](#). Examples of various settings are in [/usr/share/httpd/manual/ssl/ssl\\_howto.html](#).

When modifying the settings in the **/etc/httpd/conf.d/ssl.conf** configuration file, be sure to consider the following three directives at the minimum:

#### **SSLProtocol**

Use this directive to specify the version of TLS or SSL you want to allow.

#### **SSLCipherSuite**

Use this directive to specify your preferred cipher suite or disable the ones you want to disallow.

### SSLHonorCipherOrder

Uncomment and set this directive to **on** to ensure that the connecting clients adhere to the order of ciphers you specified.

For example, to use only the TLS 1.2 and 1.3 protocol:

```
SSLProtocol all -SSLv3 -TLSv1 -TLSv1.1
```

### 54.3.2. Configuring the Nginx HTTP and proxy server

To enable TLS 1.3 support in **Nginx**, add the **TLSv1.3** value to the **ssl\_protocols** option in the **server** section of the **/etc/nginx/nginx.conf** configuration file:

```
server {
 listen 443 ssl http2;
 listen [::]:443 ssl http2;
 ...
 ssl_protocols TLSv1.2 TLSv1.3;
 ssl_ciphers
 ...
}
```

### 54.3.3. Configuring the Dovecot mail server

To configure your installation of the **Dovecot** mail server to use TLS, modify the **/etc/dovecot/conf.d/10-ssl.conf** configuration file. You can find an explanation of some of the basic configuration directives available in that file in the [/usr/share/doc/dovecot/wiki/SSL.DovecotConfiguration.txt](#) file, which is installed along with the standard installation of **Dovecot**.

When modifying the settings in the **/etc/dovecot/conf.d/10-ssl.conf** configuration file, be sure to consider the following three directives at the minimum:

#### ssl\_protocols

Use this directive to specify the version of TLS or SSL you want to allow or disable.

#### ssl\_cipher\_list

Use this directive to specify your preferred cipher suites or disable the ones you want to disallow.

#### ssl\_prefer\_server\_ciphers

Uncomment and set this directive to **yes** to ensure that the connecting clients adhere to the order of ciphers you specified.

For example, the following line in **/etc/dovecot/conf.d/10-ssl.conf** allows only TLS 1.1 and later:

```
ssl_protocols = !SSLv2 !SSLv3 !TLSv1
```

### Additional resources

For more information about TLS configuration and related topics, see the resources listed below.

- **config(5)** man page describes the format of the **/etc/ssl/openssl.conf** configuration file.

- **ciphers(1)** man page includes a list of available **OpenSSL** keywords and cipher strings.
- [Recommendations for Secure Use of Transport Layer Security \(TLS\) and Datagram Transport Layer Security \(DTLS\)](#)
- [Mozilla SSL Configuration Generator](#) can help to create configuration files for **Apache** or **Nginx** with secure settings that disable known vulnerable protocols, ciphers, and hashing algorithms.
- [SSL Server Test](#) verifies that your configuration meets modern security requirements.

# CHAPTER 55. CONFIGURING A VPN WITH IPSEC

In Red Hat Enterprise Linux 8, a virtual private network (VPN) can be configured using the **IPsec** protocol, which is supported by the **Libreswan** application.

## 55.1. LIBRESWAN AS AN IPSEC VPN IMPLEMENTATION

In Red Hat Enterprise Linux 8, a Virtual Private Network (VPN) can be configured using the **IPsec** protocol, which is supported by the **Libreswan** application. **Libreswan** is a continuation of the **Openswan** application, and many examples from the **Openswan** documentation are interchangeable with **Libreswan**.

The **IPsec** protocol for a VPN is configured using the Internet Key Exchange ( **IKE** ) protocol. The terms IPsec and IKE are used interchangeably. An IPsec VPN is also called an IKE VPN, IKEv2 VPN, XAUTH VPN, Cisco VPN or IKE/IPsec VPN. A variant of an IPsec VPN that also uses the Level 2 Tunneling Protocol ( **L2TP** ) is usually called an L2TP/IPsec VPN, which requires the Optional channel **xl2tpd** application.

**Libreswan** is an open-source, user-space **IKE** implementation. **IKE** v1 and v2 are implemented as a user-level daemon. The IKE protocol is also encrypted. The **IPsec** protocol is implemented by the Linux kernel, and **Libreswan** configures the kernel to add and remove VPN tunnel configurations.

The **IKE** protocol uses UDP port 500 and 4500. The **IPsec** protocol consists of two protocols:

- Encapsulated Security Payload ( **ESP** ), which has protocol number 50.
- Authenticated Header ( **AH** ), which has protocol number 51.

The **AH** protocol is not recommended for use. Users of **AH** are recommended to migrate to **ESP** with null encryption.

The **IPsec** protocol provides two modes of operation:

- **Tunnel Mode** (the default)
- **Transport Mode**.

You can configure the kernel with IPsec without IKE. This is called **Manual Keying**. You can also configure manual keying using the **ip xfrm** commands, however, this is strongly discouraged for security reasons. **Libreswan** interfaces with the Linux kernel using netlink. Packet encryption and decryption happen in the Linux kernel.

**Libreswan** uses the Network Security Services ( **NSS** ) cryptographic library. Both **Libreswan** and **NSS** are certified for use with the *Federal Information Processing Standard (FIPS)* Publication 140-2.



### IMPORTANT

**IKE/IPsec** VPNs, implemented by **Libreswan** and the Linux kernel, is the only VPN technology recommended for use in Red Hat Enterprise Linux 8. Do not use any other VPN technology without understanding the risks of doing so.

In Red Hat Enterprise Linux 8, **Libreswan** follows **system-wide cryptographic policies** by default. This ensures that **Libreswan** uses secure settings for current threat models including **IKEv2** as a default protocol. See [Using system-wide crypto policies](#) for more information.

**Libreswan** does not use the terms "source" and "destination" or "server" and "client" because IKE/IPsec are peer to peer protocols. Instead, it uses the terms "left" and "right" to refer to end points (the hosts). This also allows you to use the same configuration on both end points in most cases. However, administrators usually choose to always use "left" for the local host and "right" for the remote host.

## 55.2. INSTALLING LIBRESWAN

This procedure describes the steps for installing and starting the **Libreswan** IPsec/IKE VPN implementation.

### Prerequisites

- The **AppStream** repository is enabled.

### Procedure

1. Install the **libreswan** packages:

```
yum install libreswan
```

2. If you are re-installing **Libreswan**, remove its old database files:

```
systemctl stop ipsec
rm /etc/ipsec.d/*db
```

3. Start the **ipsec** service, and enable the service to be started automatically on boot:

```
systemctl enable ipsec --now
```

4. Configure the firewall to allow 500 and 4500/UDP ports for the IKE, ESP, and AH protocols by adding the **ipsec** service:

```
firewall-cmd --add-service="ipsec"
firewall-cmd --runtime-to-permanent
```

## 55.3. CREATING A HOST-TO-HOST VPN

To configure **Libreswan** to create a host-to-host **IPsec** VPN between two hosts referred to as *left* and *right*, enter the following commands on both of the hosts:

### Procedure

1. Generate an RSA key pair on each host:

```
ipsec newhostkey --output /etc/ipsec.d/hostkey.secrets
```

2. The previous step returned the generated key's **ckaid**. Use that **ckaid** with the following command on *left*, for example:

```
ipsec showhostkey --left --ckaid 2d3ea57b61c9419dfd6cf43a1eb6cb306c0e857d
```

The output of the previous command generated the **leftrsasigkey=** line required for the configuration. Do the same on the second host (*right*):

```
ipsec showhostkey --right --ckaid a9e1f6ce9ecd3608c24e8f701318383f41798f03
```

3. In the **/etc/ipsec.d/** directory, create a new **my\_host-to-host.conf** file. Write the RSA host keys from the output of the **ipsec showhostkey** commands in the previous step to the new file. For example:

```
conn mytunnel
 leftid=@west
 left=192.1.2.23
 leftrsasigkey=0sAQOrlo+hOafUZDICQmXFrje/oZm [...] W2n417C/4urYHQkCvulQ==
 rightid=@east
 right=192.1.2.45
 rightrsasigkey=0sAQO3fwC6nSSGgt64DWiYZzuHbc4 [...] D/v8t5YTQ==
 authby=rsasig
```

4. Start **ipsec**:

```
ipsec setup start
```

5. Load the connection:

```
ipsec auto --add mytunnel
```

6. Establish the tunnel:

```
ipsec auto --up mytunnel
```

7. To automatically start the tunnel when the **ipsec** service is started, add the following line to the connection definition:

```
auto=start
```

## 55.4. CONFIGURING A SITE-TO-SITE VPN

To create a site-to-site **IPsec** VPN, by joining two networks, an **IPsec** tunnel between the two hosts, is created. The hosts thus act as the end points, which are configured to permit traffic from one or more subnets to pass through. Therefore you can think of the host as gateways to the remote portion of the network.

The configuration of the site-to-site VPN only differs from the host-to-host VPN in that one or more networks or subnets must be specified in the configuration file.

### Prerequisites

- A [host-to-host VPN](#) is already configured.

### Procedure

1. Copy the file with the configuration of your host-to-host VPN to a new file, for example:

■

```
cp /etc/ipsec.d/my_host-to-host.conf /etc/ipsec.d/my_site-to-site.conf
```

2. Add the subnet configuration to the file created in the previous step, for example:

```
conn mysubnet
 also=mytunnel
 leftsubnet=192.0.1.0/24
 rightsubnet=192.0.2.0/24
 auto=start

conn mysubnet6
 also=mytunnel
 leftsubnet=2001:db8:0:1::/64
 rightsubnet=2001:db8:0:2::/64
 auto=start

the following part of the configuration file is the same for both host-to-host and site-to-site
connections:

conn mytunnel
 leftid=@west
 left=192.1.2.23
 lefrsasigkey=0sAQOrlo+hOafUZDICQmXFrje/oZm [...] W2n417C/4urYHQkCvulQ==
 rightid=@east
 right=192.1.2.45
 rightrsasigkey=0sAQO3fwC6nSSGgt64DWiYZzuHbc4 [...] D/v8t5YTQ==
 authby=rsasig
```

## 55.5. CONFIGURING A REMOTE ACCESS VPN

Road warriors are traveling users with mobile clients with a dynamically assigned IP address, such as laptops. The mobile clients authenticate using certificates.

The following example shows configuration for **IKEv2**, and it avoids using the **IKEv1 XAUTH** protocol.

On the server:

```
conn roadwarriors
 ikev2=insist
 # Support (roaming) MOBIKE clients (RFC 4555)
 mobike=yes
 fragmentation=yes
 left=1.2.3.4
 # if access to the LAN is given, enable this, otherwise use 0.0.0.0/0
 # leftsubnet=10.10.0.0/16
 leftsubnet=0.0.0.0/0
 leftcert=gw.example.com
 leftid=%fromcert
 leftxauthserver=yes
 leftmodecfgserver=yes
 right=%any
 # trust our own Certificate Agency
 rightca=%same
 # pick an IP address pool to assign to remote users
 # 100.64.0.0/16 prevents RFC1918 clashes when remote users are behind NAT
```

```

rightaddresspool=100.64.13.100-100.64.13.254
if you want remote clients to use some local DNS zones and servers
modecfgdns="1.2.3.4, 5.6.7.8"
modecfgdomains="internal.company.com, corp"
rightxauthclient=yes
rightmodecfgclient=yes
authby=rsasig
optionally, run the client X.509 ID through pam to allow/deny client
pam-authorize=yes
load connection, don't initiate
auto=add
kill vanished roadwarriors
dpddelay=1m
dpdtimeout=5m
dpdaction=clear

```

On the mobile client, the road warrior's device, use a slight variation of the previous configuration:

```

conn to-vpn-server
ikev2=insist
pick up our dynamic IP
left=%defaultroute
leftsubnet=0.0.0.0/0
leftcert=myname.example.com
leftid=%fromcert
leftmodecfgclient=yes
right can also be a DNS hostname
right=1.2.3.4
if access to the remote LAN is required, enable this, otherwise use 0.0.0.0/0
rightsubnet=10.10.0.0/16
rightsubnet=0.0.0.0/0
fragmentation=yes
trust our own Certificate Agency
rightca=%same
authby=rsasig
allow narrowing to the server's suggested assigned IP and remote subnet
narrowing=yes
Support (roaming) MOBIKE clients (RFC 4555)
mobike=yes
Initiate connection
auto=start

```

## 55.6. CONFIGURING A MESH VPN

A mesh VPN network, which is also known as an *any-to-any* VPN, is a network where all nodes communicate using **IPsec**. The configuration allows for exceptions for nodes that cannot use **IPsec**. The mesh VPN network can be configured in two ways:

- To require **IPsec**.
- To prefer **IPsec** but allow a fallback to clear-text communication.

Authentication between the nodes can be based on X.509 certificates or on DNS Security Extensions (DNSSEC).

The following procedure uses X.509 certificates. These certificates can be generated using any kind of Certificate Authority (CA) management system, such as the Dogtag Certificate System. Dogtag assumes that the certificates for each node are available in the PKCS #12 format (.p12 files), which contain the private key, the node certificate, and the Root CA certificate used to validate other nodes' X.509 certificates.

Each node has an identical configuration with the exception of its X.509 certificate. This allows for adding new nodes without reconfiguring any of the existing nodes in the network. The PKCS #12 files require a "friendly name", for which we use the name "node" so that the configuration files referencing the friendly name can be identical for all nodes.

## Prerequisites

- **Libreswan** is installed, and the **ipsec** service is started on each node.

## Procedure

1. On each node, import PKCS #12 files. This step requires the password used to generate the PKCS #12 files:

```
ipsec import nodeXXX.p12
```

2. Create the following three connection definitions for the **IPsec required** (private), **IPsec optional** (private-or-clear), and **No IPsec** (clear) profiles:

```
cat /etc/ipsec.d/mesh.conf
conn clear
auto=ondemand
type=passthrough
authby=never
left=%defaultroute
right=%group

conn private
auto=ondemand
type=transport
authby=rsasig
failureshunt=drop
negotiationshunt=drop
left
left=%defaultroute
leftcert=nodeXXXX
leftid=%fromcert
leftrsasigkey=%cert
right
rightrsasigkey=%cert
rightid=%fromcert
right=%opportunisticgroup

conn private-or-clear
auto=ondemand
type=transport
authby=rsasig
failureshunt=passthrough
negotiationshunt=passthrough
```

```

left
left=%defaultroute
leftcert=nodeXXXX
leftid=%fromcert
leftrsasigkey=%cert
right
rightrsasigkey=%cert
rightid=%fromcert
right=%opportunisticgroup

```

3. Add the IP address of the network in the proper category. For example, if all nodes reside in the 10.15.0.0/16 network, and all nodes should mandate **IPsec** encryption:

```
echo "10.15.0.0/16" >> /etc/ipsec.d/policies/private
```

4. To allow certain nodes, for example, 10.15.34.0/24, to work with and without **IPsec**, add those nodes to the private-or-clear group using:

```
echo "10.15.34.0/24" >> /etc/ipsec.d/policies/private-or-clear
```

5. To define a host, for example, 10.15.1.2, that is not capable of **IPsec** into the clear group, use:

```
echo "10.15.1.2/32" >> /etc/ipsec.d/policies/clear
```

The files in the **/etc/ipsec.d/policies** directory can be created from a template for each new node, or can be provisioned using Puppet or Ansible.

Note that every node has the same list of exceptions or different traffic flow expectations. Two nodes, therefore, might not be able to communicate because one requires **IPsec** and the other cannot use **IPsec**.

6. Restart the node to add it to the configured mesh:

```
systemctl restart ipsec
```

7. Once you finish with the addition of nodes, a **ping** command is sufficient to open an **IPsec** tunnel. To see which tunnels a node has opened:

```
ipsec trafficstatus
```

## 55.7. METHODS OF AUTHENTICATION USED IN LIBRESWAN

You can use the following methods for authentication of end points:

- *Pre-Shared Keys* (PSK) is the simplest authentication method. PSKs should consist of random characters and have a length of at least 20 characters. In FIPS mode, PSKs need to comply to a minimum strength requirement depending on the integrity algorithm used. It is recommended not to use PSKs shorter than 64 random characters.
- *Raw RSA keys* are commonly used for static host-to-host or subnet-to-subnet **IPsec** configurations. The hosts are manually configured with each other's public RSA key. This method does not scale well when dozens or more hosts all need to setup **IPsec** tunnels to each other.

- *X.509 certificates* are commonly used for large-scale deployments where there are many hosts that need to connect to a common **IPsec** gateway. A central *certificate authority* (CA) is used to sign RSA certificates for hosts or users. This central CA is responsible for relaying trust, including the revocations of individual hosts or users.
- *NULL authentication* is used to gain mesh encryption without authentication. It protects against passive attacks but does not protect against active attacks. However, since **IKEv2** allows asymmetrical authentication methods, NULL authentication can also be used for internet scale opportunistic IPsec, where clients authenticate the server, but servers do not authenticate the client. This model is similar to secure websites using **TLS**.

## Protection against quantum computers

In addition to these authentication methods, you can use the *Postquantum Preshared Keys* (PPK) method to protect against possible attacks by quantum computers. Individual clients or groups of clients can use their own PPK by specifying a (PPKID) that corresponds to an out-of-band configured PreShared Key.

Using **IKEv1** with PreShared Keys provided protection against quantum attackers. The redesign of **IKEv2** does not offer this protection natively. **Libreswan** offers the use of *Postquantum Preshared Keys* (PPK) to protect **IKEv2** connections against quantum attacks.

To enable optional PPK support, add **ppk=yes** to the connection definition. To require PPK, add **ppk=insist**. Then, each client can be given a PPK ID with a secret value that is communicated out-of-band (and preferably quantum safe). The PPK's should be very strong in randomness and not be based on dictionary words. The PPK ID and PPK data itself are stored in **ipsec.secrets**, for example:

```
@west @east : PPKS "user1" "thestringsmeanttobearandomstr"
```

The **PPKS** option refers to static PPKs. An experimental function uses one-time-pad based Dynamic PPKs. Upon each connection, a new part of a one-time pad is used as the PPK. When used, that part of the dynamic PPK inside the file is overwritten with zeroes to prevent re-use. If there is no more one-time-pad material left, the connection fails. See the **ipsec.secrets(5)** man page for more information.



### WARNING

The implementation of dynamic PPKs is provided as a Technology Preview, and this functionality should be used with caution.

## 55.8. DEPLOYING A FIPS-COMPLIANT IPSEC VPN

Use this procedure to deploy a FIPS-compliant IPsec VPN solution based on Libreswan. The following steps also enable you to identify which cryptographic algorithms are available and which are disabled for Libreswan in FIPS mode.

### Prerequisites

- The **AppStream** repository is enabled.

### Procedure

1. Install the **libreswan** packages:

```
yum install libreswan
```

2. If you are re-installing **Libreswan**, remove its old NSS database:

```
systemctl stop ipsec
rm /etc/ipsec.d/*db
```

3. Start the **ipsec** service, and enable the service to be started automatically on boot:

```
systemctl enable ipsec --now
```

4. Configure the firewall to allow 500 and 4500/UDP ports for the IKE, ESP, and AH protocols by adding the **ipsec** service:

```
firewall-cmd --add-service="ipsec"
firewall-cmd --runtime-to-permanent
```

5. Switch the system to FIPS mode in RHEL 8:

```
fips-mode-setup --enable
```

6. Restart your system to allow the kernel to switch to FIPS mode:

```
reboot
```

## Verification steps

1. To confirm Libreswan is running in FIPS mode:

```
ipsec whack --fipsstatus
000 FIPS mode enabled
```

2. Alternatively, check entries for the **ipsec** unit in the **systemd** journal:

```
$ journalctl -u ipsec
...
Jan 22 11:26:50 localhost.localdomain pluto[3076]: FIPS Product: YES
Jan 22 11:26:50 localhost.localdomain pluto[3076]: FIPS Kernel: YES
Jan 22 11:26:50 localhost.localdomain pluto[3076]: FIPS Mode: YES
```

3. To see the available algorithms in FIPS mode:

```
ipsec pluto --selftest 2>&1 | head -11
FIPS Product: YES
FIPS Kernel: YES
FIPS Mode: YES
NSS DB directory: sql:/etc/ipsec.d
Initializing NSS
Opening NSS database "sql:/etc/ipsec.d" read-only
NSS initialized
```

NSS crypto library initialized  
 FIPS HMAC integrity support [enabled]  
 FIPS mode enabled for pluto daemon  
 NSS library is running in FIPS mode  
 FIPS HMAC integrity verification self-test passed

4. To query disabled algorithms in FIPS mode:

```

ipsec pluto --selftest 2>&1 | grep disabled
Encryption algorithm CAMELLIA_CTR disabled; not FIPS compliant
Encryption algorithm CAMELLIA_CBC disabled; not FIPS compliant
Encryption algorithm SERPENT_CBC disabled; not FIPS compliant
Encryption algorithm TWOFISH_CBC disabled; not FIPS compliant
Encryption algorithm TWOFISH_SSH disabled; not FIPS compliant
Encryption algorithm NULL disabled; not FIPS compliant
Encryption algorithm CHACHA20_POLY1305 disabled; not FIPS compliant
Hash algorithm MD5 disabled; not FIPS compliant
PRF algorithm HMAC_MD5 disabled; not FIPS compliant
PRF algorithm AES_XCBC disabled; not FIPS compliant
Integrity algorithm HMAC_MD5_96 disabled; not FIPS compliant
Integrity algorithm HMAC_SHA2_256_TRUNCBUG disabled; not FIPS compliant
Integrity algorithm AES_XCBC_96 disabled; not FIPS compliant
DH algorithm MODP1024 disabled; not FIPS compliant
DH algorithm MODP1536 disabled; not FIPS compliant
DH algorithm DH31 disabled; not FIPS compliant

```

5. To list all allowed algorithms and ciphers in FIPS mode:

```

ipsec pluto --selftest 2>&1 | grep ESP | grep FIPS | sed "s/^.*FIPS//"
{256,192,*128} aes_ccm, aes_ccm_c
{256,192,*128} aes_ccm_b
{256,192,*128} aes_ccm_a
[*192] 3des
{256,192,*128} aes_gcm, aes_gcm_c
{256,192,*128} aes_gcm_b
{256,192,*128} aes_gcm_a
{256,192,*128} aesctr
{256,192,*128} aes
{256,192,*128} aes_gmac
sha, sha1, sha1_96, hmac_sha1
sha512, sha2_512, sha2_512_256, hmac_sha2_512
sha384, sha2_384, sha2_384_192, hmac_sha2_384
sha2, sha256, sha2_256, sha2_256_128, hmac_sha2_256
aes_cmac
null
null, dh0
dh14
dh15
dh16
dh17
dh18
ecp_256, ecp256
ecp_384, ecp384
ecp_521, ecp521

```

## Additional resources

- [Using system-wide cryptographic policies](#)

## 55.9. RELATED INFORMATION

The following resources provide additional information regarding **Libreswan** and the **ipsec** daemon.

### Installed documentation

- **ipsec(8)** man page – Describes command options for **ipsec**.
- **ipsec.conf(5)** man page – Contains information on configuring **ipsec**.
- **ipsec.secrets(5)** man page – Describes the format of the **ipsec.secrets** file.
- **ipsec\_auto(8)** man page – Describes the use of the **auto** command-line client for manipulating Libreswan IPsec connections established using automatic exchanges of keys.
- **ipsec\_rsasigkey(8)** man page – Describes the tool used to generate RSA signature keys.
- **/usr/share/doc/libreswan-version/**

### Online documentation

<https://libreswan.org>

The website of the upstream project.

<https://libreswan.org/wiki>

The Libreswan Project Wiki.

<https://libreswan.org/man/>

All Libreswan man pages.

# CHAPTER 56. CONFIGURING MACSEC

The following section provides information on how to configure **Media Control Access Security (MACsec)**, which is an 802.1AE IEEE standard security technology for secure communication in all traffic on Ethernet links.

## 56.1. INTRODUCTION TO MACSEC

**Media Access Control Security (MACsec**, IEEE 802.1AE) encrypts and authenticates all traffic in LANs with the GCM-AES-128 algorithm. **MACsec** can protect not only **IP** but also Address Resolution Protocol (ARP), Neighbor Discovery (ND), or **DHCP**. While **IPsec** operates on the network layer (layer 3) and **SSL** or **TLS** on the application layer (layer 7), **MACsec** operates in the data link layer (layer 2). Combine **MACsec** with security protocols for other networking layers to take advantage of different security features that these standards provide.

## 56.2. USING MACSEC WITH NMCLI TOOL

This procedure shows how to configure **MACsec** with **nmcli** tool.

### Prerequisites

- The **NetworkManager** must be running.
- You already have a 16-byte hexadecimal CAK (**\$MKA\_CAK**) and a 32-byte hexadecimal CKN (**\$MKA\_CKN**).

### Procedure

```
~]# nmcli connection add type macsec \
 con-name test-macsec+ ifname macsec0 \
 connection.autoconnect no \
 macsec.parent enp1s0 macsec.mode psk \
 macsec.mka-cak $MKA_CAK \
 macsec.mka-ckn $MKA_CKN

~]# nmcli connection up test-macsec+
```

After this step, the *macsec0* device is configured and can be used for networking.

## 56.3. USING MACSEC WITH WPA\_SUPPLICANT

This procedure shows how to enable **MACsec** with a switch that performs authentication using a pre-shared Connectivity Association Key/CAK Name (CAK/CKN) pair.

### Procedure

1. Create a CAK/CKN pair. For example, the following command generates a 16-byte key in hexadecimal notation:

```
~]$ dd if=/dev/urandom count=16 bs=1 2> /dev/null | hexdump -e '1/2 "%02x"'
```

2. Create the **wpa\_supplicant.conf** configuration file and add the following lines to it:

```
ctrl_interface=/var/run/wpa_supplicant
eapol_version=3
ap_scan=0
fast_reauth=1

network={
 key_mgmt=NONE
 eapol_flags=0
 macsec_policy=1

 mka_cak=0011... # 16 bytes hexadecimal
 mka_ckn=2233... # 32 bytes hexadecimal
}
```

Use the values from the previous step to complete the **mka\_cak** and **mka\_ckn** lines in the **wpa\_supplicant.conf** configuration file.

For more information, see the **wpa\_supplicant.conf(5)** man page.

3. Assuming you are using *wlp61s0* to connect to your network, start **wpa\_supplicant** using the following command:

```
~]# wpa_supplicant -i wlp61s0 -Dmacsec_linux -c wpa_supplicant.conf
```

## 56.4. RELATED INFORMATION

For more details, see the [What's new in MACsec: setting up MACsec using wpa\\_supplicant and \(optionally\) NetworkManager](#) article. In addition, see the [MACsec: a different solution to encrypt network traffic](#) article for more information about the architecture of a **MACsec** network, use case scenarios, and configuration examples.

# CHAPTER 57. USING AND CONFIGURING FIREWALLD

A *firewall* is a way to protect machines from any unwanted traffic from outside. It enables users to control incoming network traffic on host machines by defining a set of *firewall rules*. These rules are used to sort the incoming traffic and either block it or allow through.

Note that **firewalld** with **nftables** backend does not support passing custom **nftables** rules to **firewalld**, using the **--direct** option.

## 57.1. WHEN TO USE FIREWALLD, NFTABLES, OR IPTABLES

The following is a brief overview in which scenario you should use one of the following utilities:

- **firewalld**: Use the **firewalld** utility to configure a firewall on workstations. The utility is easy to use and covers the typical use cases for this scenario.
- **nftables**: Use the **nftables** utility to set up complex firewalls, such as for a whole network.
- **iptables**: The **iptables** utility is deprecated in Red Hat Enterprise Linux 8. Use instead **nftables**.



### IMPORTANT

To avoid that the different firewall services influence each other, run only one of them on a RHEL host, and disable the other services.

## 57.2. GETTING STARTED WITH FIREWALLD

### 57.2.1. **firewalld**

**firewalld** is a firewall service daemon that provides a dynamic customizable host-based firewall with a **D-Bus** interface. Being dynamic, it enables creating, changing, and deleting the rules without the necessity to restart the firewall daemon each time the rules are changed.

**firewalld** uses the concepts of *zones* and *services*, that simplify the traffic management. Zones are predefined sets of rules. Network interfaces and sources can be assigned to a zone. The traffic allowed depends on the network your computer is connected to and the security level this network is assigned. Firewall services are predefined rules that cover all necessary settings to allow incoming traffic for a specific service and they apply within a zone.

Services use one or more *ports* or *addresses* for network communication. Firewalls filter communication based on ports. To allow network traffic for a service, its ports must be *open*. **firewalld** blocks all traffic on ports that are not explicitly set as open. Some zones, such as *trusted*, allow all traffic by default.

#### Additional resources

- **firewalld(1)** man page

### 57.2.2. **Zones**

**firewalld** can be used to separate networks into different zones according to the level of trust that the user has decided to place on the interfaces and traffic within that network. A connection can only be part of one zone, but a zone can be used for many network connections.

**NetworkManager** notifies **firewalld** of the zone of an interface. You can assign zones to interfaces with:

- **NetworkManager**
- **firewall-config** tool
- **firewall-cmd** command-line tool
- The RHEL web console

The latter three can only edit the appropriate **NetworkManager** configuration files. If you change the zone of the interface using the web console, **firewall-cmd** or **firewall-config**, the request is forwarded to **NetworkManager** and is not handled by **firewalld**.

The predefined zones are stored in the **/usr/lib/firewalld/zones** directory and can be instantly applied to any available network interface. These files are copied to the **/etc/firewalld/zones** directory only after they are modified. The default settings of the predefined zones are as follows:

### **block**

Any incoming network connections are rejected with an icmp-host-prohibited message for **IPv4** and icmp6-adm-prohibited for **IPv6**. Only network connections initiated from within the system are possible.

### **dmz**

For computers in your demilitarized zone that are publicly-accessible with limited access to your internal network. Only selected incoming connections are accepted.

### **drop**

Any incoming network packets are dropped without any notification. Only outgoing network connections are possible.

### **external**

For use on external networks with masquerading enabled, especially for routers. You do not trust the other computers on the network to not harm your computer. Only selected incoming connections are accepted.

### **home**

For use at home when you mostly trust the other computers on the network. Only selected incoming connections are accepted.

### **internal**

For use on internal networks when you mostly trust the other computers on the network. Only selected incoming connections are accepted.

### **public**

For use in public areas where you do not trust other computers on the network. Only selected incoming connections are accepted.

### **trusted**

All network connections are accepted.

### **work**

For use at work where you mostly trust the other computers on the network. Only selected incoming connections are accepted.

One of these zones is set as the *default* zone. When interface connections are added to **NetworkManager**, they are assigned to the default zone. On installation, the default zone in **firewalld** is set to be the **public** zone. The default zone can be changed.



## NOTE

The network zone names should be self-explanatory and to allow users to quickly make a reasonable decision. To avoid any security problems, review the default zone configuration and disable any unnecessary services according to your needs and risk assessments.

### Additional resources

- **firewalld.zone(5)** man page

### 57.2.3. Predefined services

A service can be a list of local ports, protocols, source ports, and destinations, as well as a list of firewall helper modules automatically loaded if a service is enabled. Using services saves users time because they can achieve several tasks, such as opening ports, defining protocols, enabling packet forwarding and more, in a single step, rather than setting up everything one after another.

Service configuration options and generic file information are described in the **firewalld.service(5)** man page. The services are specified by means of individual XML configuration files, which are named in the following format: **service-name.xml**. Protocol names are preferred over service or application names in **firewalld**.

Services can be added and removed using the graphical **firewall-config** tool, **firewall-cmd**, and **firewall-offline-cmd**.

Alternatively, you can edit the XML files in the **/etc/firewalld/services/** directory. If a service is not added or changed by the user, then no corresponding XML file is found in **/etc/firewalld/services/**. The files in the **/usr/lib/firewalld/services/** directory can be used as templates if you want to add or change a service.

### Additional resources

- **firewalld.service(5)** man page

## 57.3. INSTALLING THE FIREWALL-CONFIG GUI CONFIGURATION TOOL

To use the **firewall-config** GUI configuration tool, install the **firewall-config** package.

### Procedure

1. Enter the following command as **root**:

```
yum install firewall-config
```

Alternatively, in **GNOME**, use the **Super** key and type **Software** to launch the **Software Sources** application. Type **firewall** to the search box, which appears after selecting the search button in the top-right corner. Select the **Firewall** item from the search results, and click on the **Install** button.

2. To run **firewall-config**, use either the **firewall-config** command or press the **Super** key to enter the **Activities Overview**, type **firewall**, and press **Enter**.

## 57.4. VIEWING THE CURRENT STATUS AND SETTINGS OF FIREWALLD

### 57.4.1. Viewing the current status of firewalld

The firewall service, **firewalld**, is installed on the system by default. Use the **firewalld** CLI interface to check that the service is running.

#### Procedure

1. To see the status of the service:

```
firewall-cmd --state
```

2. For more information about the service status, use the **systemctl status** sub-command:

```
systemctl status firewalld
firewalld.service - firewalld - dynamic firewall daemon
 Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor pr
 Active: active (running) since Mon 2017-12-18 16:05:15 CET; 50min ago
 Docs: man:firewalld(1)
 Main PID: 705 (firewalld)
 Tasks: 2 (limit: 4915)
 CGroup: /system.slice/firewalld.service
 └─705 /usr/bin/python3 -Es /usr/sbin/firewalld --nofork --nopid
```

#### Additional resources

It is important to know how **firewalld** is set up and which rules are in force before you try to edit the settings. To display the firewall settings, see [Section 57.4.2, “Viewing current firewalld settings”](#)

### 57.4.2. Viewing current firewalld settings

#### 57.4.2.1. Viewing allowed services using GUI

To view the list of services using the graphical **firewall-config** tool, press the **Super** key to enter the Activities Overview, type **firewall**, and press **Enter**. The **firewall-config** tool appears. You can now view the list of services under the **Services** tab.

Alternatively, to start the graphical firewall configuration tool using the command-line, enter the following command:

```
$ firewall-config
```

The **Firewall Configuration** window opens. Note that this command can be run as a normal user, but you are prompted for an administrator password occasionally.

#### 57.4.2.2. Viewing firewalld settings using CLI

With the CLI client, it is possible to get different views of the current firewall settings. The **--list-all** option shows a complete overview of the **firewalld** settings.

**firewalld** uses zones to manage the traffic. If a zone is not specified by the **--zone** option, the command is effective in the default zone assigned to the active network interface and connection.

To list all the relevant information for the default zone:

```
firewall-cmd --list-all
public
 target: default
 icmp-block-inversion: no
 interfaces:
 sources:
 services: ssh dhcpcv6-client
 ports:
 protocols:
 masquerade: no
 forward-ports:
 source-ports:
 icmp-blocks:
 rich rules:
```

To specify the zone for which to display the settings, add the **--zone=zone-name** argument to the **firewall-cmd --list-all** command, for example:

```
firewall-cmd --list-all --zone=home
home
 target: default
 icmp-block-inversion: no
 interfaces:
 sources:
 services: ssh mdns samba-client dhcpcv6-client
 ... [trimmed for clarity]
```

To see the settings for particular information, such as services or ports, use a specific option. See the **firewalld** manual pages or get a list of the options using the command help:

```
firewall-cmd --help

Usage: firewall-cmd [OPTIONS...]

General Options
-h, --help Prints a short help text and exists
-V, --version Print the version string of firewalld
-q, --quiet Do not print status messages

Status Options
--state Return and print firewalld state
--reload Reload firewall and keep state information
... [trimmed for clarity]
```

For example, to see which services are allowed in the current zone:

```
firewall-cmd --list-services
ssh dhcpcv6-client
```



## NOTE

Listing the settings for a certain subpart using the CLI tool can sometimes be difficult to interpret. For example, you allow the **SSH** service and **firewalld** opens the necessary port (22) for the service. Later, if you list the allowed services, the list shows the **SSH** service, but if you list open ports, it does not show any. Therefore, it is recommended to use the **--list-all** option to make sure you receive a complete information.

## 57.5. STARTING FIREWALLD

### Procedure

1. To start **firewalld**, enter the following command as **root**:

```
systemctl unmask firewalld
systemctl start firewalld
```

2. To ensure **firewalld** starts automatically at system start, enter the following command as **root**:

```
systemctl enable firewalld
```

## 57.6. STOPPING FIREWALLD

### Procedure

1. To stop **firewalld**, enter the following command as **root**:

```
systemctl stop firewalld
```

2. To prevent **firewalld** from starting automatically at system start:

```
systemctl disable firewalld
```

3. To make sure **firewalld** is not started by accessing the **firewalld D-Bus** interface and also if other services require **firewalld**:

```
systemctl mask firewalld
```

## 57.7. RUNTIME AND PERMANENT SETTINGS

Any changes committed in *runtime* mode only apply while **firewalld** is running. When **firewalld** is restarted, the settings revert to their *permanent* values.

To make the changes persistent across reboots, apply them again using the **--permanent** option. Alternatively, to make changes persistent while **firewalld** is running, use the **--runtime-to-permanent** **firewall-cmd** option.

If you set the rules while **firewalld** is running using only the **--permanent** option, they do not become effective before **firewalld** is restarted. However, restarting **firewalld** closes all open ports and stops the networking traffic.

### Modifying settings in runtime and permanent configuration using CLI

Using the CLI, you do not modify the firewall settings in both modes at the same time. You only modify either runtime or permanent mode. To modify the firewall settings in the permanent mode, use the **--permanent** option with the **firewall-cmd** command.

```
firewall-cmd --permanent <other options>
```

Without this option, the command modifies runtime mode.

To change settings in both modes, you can use two methods:

1. Change runtime settings and then make them permanent as follows:

```
firewall-cmd <other options>
firewall-cmd --runtime-to-permanent
```

2. Set permanent settings and reload the settings into runtime mode:

```
firewall-cmd --permanent <other options>
firewall-cmd --reload
```

The first method allows you to test the settings before you apply them to the permanent mode.



#### NOTE

It is possible, especially on remote systems, that an incorrect setting results in a user locking themselves out of a machine. To prevent such situations, use the **--timeout** option. After a specified amount of time, any change reverts to its previous state. Using this option excludes the **--permanent** option.

For example, to add the **SSH** service for 15 minutes:

```
firewall-cmd --add-service=ssh --timeout 15m
```

## 57.8. VERIFYING THE PERMANENT FIREWALLD CONFIGURATION

In certain situations, for example after manually editing **firewalld** configuration files, administrators want to verify that the changes are correct. This section describes how to verify the permanent configuration of the **firewalld** service.

### Prerequisites

- The **firewalld** service is running.

### Procedure

1. Verify the permanent configuration of the **firewalld** service:

```
firewall-cmd --check-config
success
```

If the permanent configuration is valid, the command returns **success**. In other cases, the command returns an error with further details, such as the following:



```
firewall-cmd --check-config
Error: INVALID_PROTOCOL: 'public.xml': 'tcpx' not from {'tcp'|'udp'|'sctp'|'dccp'}
```

## 57.9. CONTROLLING NETWORK TRAFFIC USING FIREWALLD

### 57.9.1. Disabling all traffic in case of emergency using CLI

In an emergency situation, such as a system attack, it is possible to disable all network traffic and cut off the attacker.

#### Procedure

1. To immediately disable networking traffic, switch panic mode on:

```
firewall-cmd --panic-on
```



#### IMPORTANT

Enabling panic mode stops all networking traffic. From this reason, it should be used only when you have the physical access to the machine or if you are logged in using a serial console.

Switching off panic mode reverts the firewall to its permanent settings. To switch panic mode off:

```
firewall-cmd --panic-off
```

To see whether panic mode is switched on or off, use:

```
firewall-cmd --query-panic
```

### 57.9.2. Controlling traffic with predefined services using CLI

The most straightforward method to control traffic is to add a predefined service to **firewalld**. This opens all necessary ports and modifies other settings according to the *service definition file*.

#### Procedure

1. Check that the service is not already allowed:

```
firewall-cmd --list-services
ssh dhcpcv6-client
```

2. List all predefined services:

```
firewall-cmd --get-services
RH-Satellite-6 amanda-client amanda-k5-client bacula bacula-client bitcoin bitcoin-rpc
bitcoin-testnet bitcoin-testnet-rpc ceph ceph-mon cfengine condor-collector ctdb dhcp dhcpcv6
dhcpcv6-client dns docker-registry ...
[trimmed for clarity]
```

3. Add the service to the allowed services:

```
firewall-cmd --add-service=<service-name>
```

4. Make the new settings persistent:

```
firewall-cmd --runtime-to-permanent
```

### 57.9.3. Controlling traffic with predefined services using GUI

To enable or disable a predefined or custom service:

1. Start the **firewall-config** tool and select the network zone whose services are to be configured.
2. Select the **Services** tab.
3. Select the check box for each type of service you want to trust or clear the check box to block a service.

To edit a service:

1. Start the **firewall-config** tool.
2. Select **Permanent** from the menu labeled **Configuration**. Additional icons and menu buttons appear at the bottom of the **Services** window.
3. Select the service you want to configure.

The **Ports**, **Protocols**, and **Source Port** tabs enable adding, changing, and removing of ports, protocols, and source port for the selected service. The modules tab is for configuring **Netfilter** helper modules. The **Destination** tab enables limiting traffic to a particular destination address and Internet Protocol (**IPv4** or **IPv6**).



#### NOTE

It is not possible to alter service settings in **Runtime** mode.

### 57.9.4. Adding new services

Services can be added and removed using the graphical **firewall-config** tool, **firewall-cmd**, and **firewall-offline-cmd**. Alternatively, you can edit the XML files in **/etc/firewalld/services/**. If a service is not added or changed by the user, then no corresponding XML file are found in **/etc/firewalld/services/**. The files **/usr/lib/firewalld/services/** can be used as templates if you want to add or change a service.



#### NOTE

Service names must be alphanumeric and can, additionally, include only `_` (underscore) and `-` (dash) characters.

#### Procedure

To add a new service in a terminal, use **firewall-cmd**, or **firewall-offline-cmd** in case of not active **firewalld**.

1. Enter the following command to add a new and empty service:

```
$ firewall-cmd --new-service=service-name --permanent
```

2. To add a new service using a local file, use the following command:

```
$ firewall-cmd --new-service-from-file=service-name.xml --permanent
```

You can change the service name with the additional **--name=service-name** option.

3. As soon as service settings are changed, an updated copy of the service is placed into **/etc/firewalld/services/**.

As **root**, you can enter the following command to copy a service manually:

```
cp /usr/lib/firewalld/services/service-name.xml /etc/firewalld/services/service-name.xml
```

**firewalld** loads files from **/usr/lib/firewalld/services** in the first place. If files are placed in **/etc/firewalld/services** and they are valid, then these will override the matching files from **/usr/lib/firewalld/services**. The overridden files in **/usr/lib/firewalld/services** are used as soon as the matching files in **/etc/firewalld/services** have been removed or if **firewalld** has been asked to load the defaults of the services. This applies to the permanent environment only. A reload is needed to get these fallbacks also in the runtime environment.

### 57.9.5. Controlling ports using CLI

Ports are logical devices that enable an operating system to receive and distinguish network traffic and forward it accordingly to system services. These are usually represented by a daemon that listens on the port, that is it waits for any traffic coming to this port.

Normally, system services listen on standard ports that are reserved for them. The **httpd** daemon, for example, listens on port 80. However, system administrators by default configure daemons to listen on different ports to enhance security or for other reasons.

#### 57.9.5.1. Opening a port

Through open ports, the system is accessible from the outside, which represents a security risk. Generally, keep ports closed and only open them if they are required for certain services.

#### Procedure

To get a list of open ports in the current zone:

1. List all allowed ports:

```
firewall-cmd --list-ports
```

2. Add a port to the allowed ports to open it for incoming traffic:

```
firewall-cmd --add-port=port-number/port-type
```

3. Make the new settings persistent:

```
firewall-cmd --runtime-to-permanent
```

The port types are either **tcp**, **udp**, **sctp**, or **dccp**. The type must match the type of network communication.

### 57.9.5.2. Closing a port

When an open port is no longer needed, close that port in **firewalld**. It is highly recommended to close all unnecessary ports as soon as they are not used because leaving a port open represents a security risk.

#### Procedure

To close a port, remove it from the list of allowed ports:

1. List all allowed ports:

```
firewall-cmd --list-ports
```

```
[WARNING]
```

```
====
```

This command will only give you a list of ports that have been opened as ports. You will not be able to see any open ports that have been opened as a service. Therefore, you should consider using the `--list-all` option instead of `--list-ports`.

```
====
```

2. Remove the port from the allowed ports to close it for the incoming traffic:

```
firewall-cmd --remove-port=port-number/port-type
```

3. Make the new settings persistent:

```
firewall-cmd --runtime-to-permanent
```

### 57.9.6. Opening ports using GUI

To permit traffic through the firewall to a certain port:

1. Start the **firewall-config** tool and select the network zone whose settings you want to change.
2. Select the **Ports** tab and click the **Add** button on the right-hand side. The **Port and Protocol** window opens.
3. Enter the port number or range of ports to permit.
4. Select **tcp** or **udp** from the list.

### 57.9.7. Controlling traffic with protocols using GUI

To permit traffic through the firewall using a certain protocol:

1. Start the **firewall-config** tool and select the network zone whose settings you want to change.
2. Select the **Protocols** tab and click the **Add** button on the right-hand side. The **Protocol** window opens.
3. Either select a protocol from the list or select the **Other Protocol** check box and enter the protocol in the field.

### 57.9.8. Opening source ports using GUI

To permit traffic through the firewall from a certain port:

1. Start the firewall-config tool and select the network zone whose settings you want to change.
2. Select the **Source Port** tab and click the **Add** button on the right-hand side. The **Source Port** window opens.
3. Enter the port number or range of ports to permit. Select **tcp** or **udp** from the list.

## 57.10. WORKING WITH FIREWALLD ZONES

Zones represent a concept to manage incoming traffic more transparently. The zones are connected to networking interfaces or assigned a range of source addresses. You manage firewall rules for each zone independently, which enables you to define complex firewall settings and apply them to the traffic.

### 57.10.1. Listing zones

#### Procedure

1. To see which zones are available on your system:

```
firewall-cmd --get-zones
```

The **firewall-cmd --get-zones** command displays all zones that are available on the system, but it does not show any details for particular zones.

2. To see detailed information for all zones:

```
firewall-cmd --list-all-zones
```

3. To see detailed information for a specific zone:

```
firewall-cmd --zone=zone-name --list-all
```

### 57.10.2. Modifying firewalld settings for a certain zone

The [Section 57.9.2, “Controlling traffic with predefined services using CLI”](#) and [Section 57.9.5, “Controlling ports using CLI”](#) explain how to add services or modify ports in the scope of the current working zone. Sometimes, it is required to set up rules in a different zone.

#### Procedure

1. To work in a different zone, use the **--zone=zone-name** option. For example, to allow the **SSH** service in the zone *public*:

```
firewall-cmd --add-service=ssh --zone=public
```

### 57.10.3. Changing the default zone

System administrators assign a zone to a networking interface in its configuration files. If an interface is not assigned to a specific zone, it is assigned to the default zone. After each restart of the **firewalld** service, **firewalld** loads the settings for the default zone and makes it active.

## Procedure

To set up the default zone:

1. Display the current default zone:

```
firewall-cmd --get-default-zone
```

2. Set the new default zone:

```
firewall-cmd --set-default-zone zone-name
```



### NOTE

Following this procedure, the setting is a permanent setting, even without the **--permanent** option.

### 57.10.4. Assigning a network interface to a zone

It is possible to define different sets of rules for different zones and then change the settings quickly by changing the zone for the interface that is being used. With multiple interfaces, a specific zone can be set for each of them to distinguish traffic that is coming through them.

## Procedure

To assign the zone to a specific interface:

1. List the active zones and the interfaces assigned to them:

```
firewall-cmd --get-active-zones
```

2. Assign the interface to a different zone:

```
firewall-cmd --zone=zone_name --change-interface=interface_name --permanent
```

### 57.10.5. Assigning a zone to a connection using nmcli

This procedure describes how to add a firewalld zone to a NetworkManager connection using the **nmcli** utility.

## Procedure

1. Assign the zone to the NetworkManager connection profile:

```
nmcli connection profile modify zone.connection zone_name
```

2. Reload the connection:

```
nmcli connection up profile
```

### 57.10.6. Manually assigning a zone to a network connection in an `ifcfg` file

When the connection is managed by **NetworkManager**, it must be aware of a zone that it uses. For every network connection, a zone can be specified, which provides the flexibility of various firewall settings according to the location of the computer with portable devices. Thus, zones and settings can be specified for different locations, such as company or home.

#### Procedure

1. To set a zone for a connection, edit the `/etc/sysconfig/network-scripts/ifcfg-connection_name` file and add a line that assigns a zone to this connection:

```
ZONE=zone_name
```

### 57.10.7. Creating a new zone

To use custom zones, create a new zone and use it just like a predefined zone. New zones require the `--permanent` option, otherwise the command does not work.

#### Procedure

To create a new zone:

1. Create a new zone:

```
firewall-cmd --new-zone=zone-name
```

2. Check if the new zone is added to your permanent settings:

```
firewall-cmd --get-zones
```

3. Make the new settings persistent:

```
firewall-cmd --runtime-to-permanent
```

### 57.10.8. Zone configuration files

Zones can also be created using a *zone configuration file*. This approach can be helpful when you need to create a new zone, but want to reuse the settings from a different zone and only alter them a little.

A **firewall** zone configuration file contains the information for a zone. These are the zone description, services, ports, protocols, icmp-blocks, masquerade, forward-ports and rich language rules in an XML file format. The file name has to be `zone-name.xml` where the length of `zone-name` is currently limited to 17 chars. The zone configuration files are located in the `/usr/lib/firewall/zones/` and `/etc/firewall/zones/` directories.

The following example shows a configuration that allows one service (**SSH**) and one port range, for both the **TCP** and **UDP** protocols:

```
<?xml version="1.0" encoding="utf-8"?>
<zone>
<short>My zone</short>
<description>Here you can describe the characteristic features of the zone.</description>
<service name="ssh"/>
```

```
<port port="1025-65535" protocol="tcp"/>
<port port="1025-65535" protocol="udp"/>
</zone>
```

To change settings for that zone, add or remove sections to add ports, forward ports, services, and so on.

### Additional resources

- For more information, see the **firewalld.zone** manual pages.

## 57.10.9. Using zone targets to set default behavior for incoming traffic

For every zone, you can set a default behavior that handles incoming traffic that is not further specified. Such behaviour is defined by setting the target of the zone. There are three options - **default**, **ACCEPT**, **REJECT**, and **DROP**. By setting the target to **ACCEPT**, you accept all incoming packets except those disabled by a specific rule. If you set the target to **REJECT** or **DROP**, you disable all incoming packets except those that you have allowed in specific rules. When packets are rejected, the source machine is informed about the rejection, while there is no information sent when the packets are dropped.

### Procedure

To set a target for a zone:

- List the information for the specific zone to see the default target:

```
$ firewall-cmd --zone=zone-name --list-all
```

- Set a new target in the zone:

```
firewall-cmd --zone=zone-name --set-target=<default|ACCEPT|REJECT|DROP>
```

## 57.11. USING ZONES TO MANAGE INCOMING TRAFFIC DEPENDING ON A SOURCE

### 57.11.1. Using zones to manage incoming traffic depending on a source

You can use zones to manage incoming traffic based on its source. That enables you to sort incoming traffic and route it through different zones to allow or disallow services that can be reached by that traffic.

If you add a source to a zone, the zone becomes active and any incoming traffic from that source will be directed through it. You can specify different settings for each zone, which is applied to the traffic from the given sources accordingly. You can use more zones even if you only have one network interface.

### 57.11.2. Adding a source

To route incoming traffic into a specific source, add the source to that zone. The source can be an IP address or an IP mask in the Classless Inter-domain Routing (CIDR) notation.

- To set the source in the current zone:

```
firewall-cmd --add-source=<source>
```

- To set the source IP address for a specific zone:

```
firewall-cmd --zone=zone-name --add-source=<source>
```

The following procedure allows all incoming traffic from 192.168.2.15 in the **trusted** zone:

### Procedure

1. List all available zones:

```
firewall-cmd --get-zones
```

2. Add the source IP to the trusted zone in the permanent mode:

```
firewall-cmd --zone=trusted --add-source=192.168.2.15
```

3. Make the new settings persistent:

```
firewall-cmd --runtime-to-permanent
```

### 57.11.3. Removing a source

Removing a source from the zone cuts off the traffic coming from it.

### Procedure

1. List allowed sources for the required zone:

```
firewall-cmd --zone=zone-name --list-sources
```

2. Remove the source from the zone permanently:

```
firewall-cmd --zone=zone-name --remove-source=<source>
```

3. Make the new settings persistent:

```
firewall-cmd --runtime-to-permanent
```

### 57.11.4. Adding a source port

To enable sorting the traffic based on a port of origin, specify a source port using the **--add-source-port** option. You can also combine this with the **--add-source** option to limit the traffic to a certain IP address or IP range.

### Procedure

1. To add a source port:

```
firewall-cmd --zone=zone-name --add-source-port=<port-name>/<tcp|udp|sctp|dccp>
```

### 57.11.5. Removing a source port

By removing a source port you disable sorting the traffic based on a port of origin.

### Procedure

1. To remove a source port:

```
firewall-cmd --zone=zone-name --remove-source-port=<port-name>/<tcp|udp|sctp|dccp>
```

## 57.11.6. Using zones and sources to allow a service for only a specific domain

To allow traffic from a specific network to use a service on a machine, use zones and source. The following procedure allows traffic from 192.168.1.0/24 to be able to reach the *HTTP* service while any other traffic is blocked.

### Procedure

1. List all available zones:

```
firewall-cmd --get-zones
block dmz drop external home internal public trusted work
```

2. Add the source to the trusted zone to route the traffic originating from the source through the zone:

```
firewall-cmd --zone=trusted --add-source=192.168.1.0/24
```

3. Add the *http* service in the trusted zone:

```
firewall-cmd --zone=trusted -add-service=http
```

4. Make the new settings persistent:

```
firewall-cmd --runtime-to-permanent
```

5. Check that the trusted zone is active and that the service is allowed in it:

```
firewall-cmd --zone=trusted --list-all
trusted (active)
target: ACCEPT
sources: 192.168.1.0/24
services: http
```

## 57.11.7. Configuring traffic accepted by a zone based on a protocol

You can allow incoming traffic to be accepted by a zone based on a protocol. All traffic using the specified protocol is accepted by a zone, in which you can apply further rules and filtering.

### 57.11.7.1. Adding a protocol to a zone

By adding a protocol to a certain zone, you allow all traffic with this protocol to be accepted by this zone.

### Procedure

1. To add a protocol to a zone:

```
firewall-cmd --zone=zone-name --add-protocol=port-name/tcp|udp|sctp|dccp|igmp
```



#### NOTE

To receive multicast traffic, use the **igmp** value with the **--add-protocol** option.

### 57.11.7.2. Removing a protocol from a zone

By removing a protocol from a certain zone, you stop accepting all traffic based on this protocol by the zone.

#### Procedure

1. To remove a protocol from a zone:

```
firewall-cmd --zone=zone-name --remove-protocol=port-name/tcp|udp|sctp|dccp|igmp
```

## 57.12. CONFIGURING IP ADDRESS MASQUERADED

The following procedure describes how to enable IP masquerading on your system. IP masquerading hides individual machines behind a gateway when accessing the Internet.

#### Procedure

1. To check if IP masquerading is enabled (for example, for the **external** zone), enter the following command as **root**:

```
firewall-cmd --zone=external --query-masquerade
```

The command prints **yes** with exit status **0** if enabled. It prints **no** with exit status **1** otherwise. If **zone** is omitted, the default zone will be used.

2. To enable IP masquerading, enter the following command as **root**:

```
firewall-cmd --zone=external --add-masquerade
```

3. To make this setting persistent, repeat the command adding the **--permanent** option.

To disable IP masquerading, enter the following command as **root**:

```
firewall-cmd --zone=external --remove-masquerade --permanent
```

## 57.13. PORT FORWARDING

Redirecting ports using this method only works for IPv4-based traffic. For IPv6 redirecting setup, you must use rich rules.

To redirect to an external system, it is necessary to enable masquerading. For more information, see [Configuring IP address masquerading](#).

### 57.13.1. Adding a port to redirect

Using **firewalld**, you can set up ports redirection so that any incoming traffic that reaches a certain port on your system is delivered to another internal port of your choice or to an external port on another machine.

#### Prerequisites

- Before you redirect traffic from one port to another port, or another address, you have to know three things: which port the packets arrive at, what protocol is used, and where you want to redirect them.

#### Procedure

To redirect a port to another port:

```
firewall-cmd --add-forward-port=port=port-number:proto=tcp|udp|sctp|dccp:toport=port-number
```

To redirect a port to another port at a different IP address:

1. Add the port to be forwarded:

```
firewall-cmd --add-forward-port=port=port-number:proto=tcp|udp:toport=port-number:toaddr=IP/mask
```

2. Enable masquerade:

```
firewall-cmd --add-masquerade
```

### 57.13.2. Redirecting TCP port 80 to port 88 on the same machine

Follow the steps to redirect the TCP port 80 to port 88.

#### Procedure

1. Redirect the port 80 to port 88 for TCP traffic:

```
firewall-cmd --add-forward-port=port=80:proto=tcp:toport=88
```

2. Make the new settings persistent:

```
firewall-cmd --runtime-to-permanent
```

3. Check that the port is redirected:

```
firewall-cmd --list-all
```

### 57.13.3. Removing a redirected port

To remove a redirected port:

```
~]# firewall-cmd --remove-forward-port=port=port-number:proto=<tcp|udp>:toport=port-number:toaddr=<IP/mask>
```

To remove a forwarded port redirected to a different address, use the following procedure.

#### Procedure

1. Remove the forwarded port:

```
~]# firewall-cmd --remove-forward-port=port=port-number:proto=<tcp|udp>:toport=port-number:toaddr=<IP/mask>
```

2. Disable masquerade:

```
~]# firewall-cmd --remove-masquerade
```

### 57.13.4. Removing TCP port 80 forwarded to port 88 on the same machine

To remove the port redirection:

#### Procedure

1. List redirected ports:

```
~]# firewall-cmd --list-forward-ports
port=80:proto=tcp:toport=88:toaddr=
```

2. Remove the redirected port from the firewall::

```
~]# firewall-cmd --remove-forward-port=port=80:proto=tcp:toport=88:toaddr=
```

3. Make the new settings persistent:

```
~]# firewall-cmd --runtime-to-permanent
```

## 57.14. MANAGING ICMP REQUESTS

The **Internet Control Message Protocol (ICMP)** is a supporting protocol that is used by various network devices to send error messages and operational information indicating a connection problem, for example, that a requested service is not available. **ICMP** differs from transport protocols such as TCP and UDP because it is not used to exchange data between systems.

Unfortunately, it is possible to use the **ICMP** messages, especially **echo-request** and **echo-reply**, to reveal information about your network and misuse such information for various kinds of fraudulent activities. Therefore, **firewalld** enables blocking the **ICMP** requests to protect your network information.

### 57.14.1. Listing and blocking ICMP requests

#### Listing ICMP requests

The **ICMP** requests are described in individual XML files that are located in the **/usr/lib/firewalld/icmp-types** directory. You can read these files to see a description of the request. The **firewall-cmd** command controls the **ICMP** requests manipulation.

- To list all available **ICMP** types:

```
firewall-cmd --get-icmptypes
```

- The **ICMP** request can be used by IPv4, IPv6, or by both protocols. To see for which protocol the **ICMP** request is used:

```
firewall-cmd --info-icmptype=<icmptype>
```

- The status of an **ICMP** request shows **yes** if the request is currently blocked or **no** if it is not. To see if an **ICMP** request is currently blocked:

```
firewall-cmd --query-icmp-block=<icmptype>
```

## Blocking or unblocking ICMP requests

When your server blocks **ICMP** requests, it does not provide the information that it normally would. However, that does not mean that no information is given at all. The clients receive information that the particular **ICMP** request is being blocked (rejected). Blocking the **ICMP** requests should be considered carefully, because it can cause communication problems, especially with IPv6 traffic.

- To see if an **ICMP** request is currently blocked:

```
firewall-cmd --query-icmp-block=<icmptype>
```

- To block an **ICMP** request:

```
firewall-cmd --add-icmp-block=<icmptype>
```

- To remove the block for an **ICMP** request:

```
firewall-cmd --remove-icmp-block=<icmptype>
```

## Blocking ICMP requests without providing any information at all

Normally, if you block **ICMP** requests, clients know that you are blocking it. So, a potential attacker who is sniffing for live IP addresses is still able to see that your IP address is online. To hide this information completely, you have to drop all **ICMP** requests.

- To block and drop all **ICMP** requests:

1. Set the target of your zone to **DROP**:

```
firewall-cmd --set-target=DROP
```

2. Make the new settings persistent:

```
firewall-cmd --runtime-to-permanent
```

Now, all traffic, including **ICMP** requests, is dropped, except traffic which you have explicitly allowed.

- To block and drop certain **ICMP** requests and allow others:

1. Set the target of your zone to **DROP**:

- # firewall-cmd --set-target=DROP
- 2. Add the ICMP block inversion to block all **ICMP** requests at once:
  - # firewall-cmd --add-icmp-block-inversion
- 3. Add the ICMP block for those **ICMP** requests that you want to allow:
  - # firewall-cmd --add-icmp-block=<icmptype>
- 4. Make the new settings persistent:
  - # firewall-cmd --runtime-to-permanent

The *block inversion* inverts the setting of the **ICMP** requests blocks, so all requests, that were not previously blocked, are blocked. Those that were blocked are not blocked. Which means that if you need to unblock a request, you must use the blocking command.

- To revert the block inversion to a fully permissive setting:
  1. Set the target of your zone to **default** or **ACCEPT**:
    - # firewall-cmd --set-target=default
  2. Remove all added blocks for **ICMP** requests:
    - # firewall-cmd --remove-icmp-block=<icmptype>
  3. Remove the **ICMP** block inversion:
    - # firewall-cmd --remove-icmp-block-inversion
  4. Make the new settings persistent:
    - # firewall-cmd --runtime-to-permanent

### 57.14.2. Configuring the ICMP filter using GUI

- To enable or disable an **ICMP** filter, start the **firewall-config** tool and select the network zone whose messages are to be filtered. Select the **ICMP Filter** tab and select the check box for each type of **ICMP** message you want to filter. Clear the check box to disable a filter. This setting is per direction and the default allows everything.
- To edit an **ICMP** type, start the **firewall-config** tool and select **Permanent** mode from the menu labeled **Configuration**. Additional icons appear at the bottom of the **Services** window. Select **Yes** in the following dialog to enable masquerading and to make forwarding to another machine working.
- To enable inverting the **ICMP Filter**, click the **Invert Filter** check box on the right. Only marked **ICMP** types are now accepted, all other are rejected. In a zone using the **DROP** target, they are dropped.

## 57.15. SETTING AND CONTROLLING IP SETS USING FIREWALLD

To see the list of IP set types supported by **firewalld**, enter the following command as **root**.

```
~]# firewall-cmd --get-ipset-types
hash:ip hash:ip,mark hash:ip,port hash:ip,port,ip hash:ip,port,net hash:mac hash:net hash:net,iface
hash:net,net hash:net,port hash:net,port,net
```

### 57.15.1. Configuring IP set options using CLI

IP sets can be used in **firewalld** zones as sources and also as sources in rich rules. In Red Hat Enterprise Linux, the preferred method is to use the IP sets created with **firewalld** in a direct rule.

- To list the IP sets known to **firewalld** in the permanent environment, use the following command as **root**:

```
firewall-cmd --permanent --get-ipsets
```

- To add a new IP set, use the following command using the permanent environment as **root**:

```
firewall-cmd --permanent --new-ipset=test --type=hash:net
success
```

The previous command creates a new IP set with the name *test* and the **hash:net** type for **IPv4**.

To create an IP set for use with **IPv6**, add the **--option=family=inet6** option. To make the new setting effective in the runtime environment, reload **firewalld**.

- List the new IP set with the following command as **root**:

```
firewall-cmd --permanent --get-ipsets
test
```

- To get more information about the IP set, use the following command as **root**:

```
firewall-cmd --permanent --info-ipset=test
test
type: hash:net
options:
entries:
```

Note that the IP set does not have any entries at the moment.

- To add an entry to the *test* IP set, use the following command as **root**:

```
firewall-cmd --permanent --ipset=test --add-entry=192.168.0.1
success
```

The previous command adds the IP address *192.168.0.1* to the IP set.

- To get the list of current entries in the IP set, use the following command as **root**:

```
firewall-cmd --permanent --ipset=test --get-entries
192.168.0.1
```

- Generate a file containing a list of IP addresses, for example:

```
cat > iplist.txt <<EOL
192.168.0.2
192.168.0.3
192.168.1.0/24
192.168.2.254
EOL
```

The file with the list of IP addresses for an IP set should contain an entry per line. Lines starting with a hash, a semi-colon, or empty lines are ignored.

- To add the addresses from the *iplist.txt* file, use the following command as **root**:

```
firewall-cmd --permanent --ipset=test --add-entries-from-file=iplist.txt
success
```

- To see the extended entries list of the IP set, use the following command as **root**:

```
firewall-cmd --permanent --ipset=test --get-entries
192.168.0.1
192.168.0.2
192.168.0.3
192.168.1.0/24
192.168.2.254
```

- To remove the addresses from the IP set and to check the updated entries list, use the following commands as **root**:

```
firewall-cmd --permanent --ipset=test --remove-entries-from-file=iplist.txt
success
firewall-cmd --permanent --ipset=test --get-entries
192.168.0.1
```

- You can add the IP set as a source to a zone to handle all traffic coming in from any of the addresses listed in the IP set with a zone. For example, to add the *test* IP set as a source to the *drop* zone to drop all packets coming from all entries listed in the *test* IP set, use the following command as **root**:

```
firewall-cmd --permanent --zone=drop --add-source=ipset:test
success
```

The **ipset**: prefix in the source shows **firewalld** that the source is an IP set and not an IP address or an address range.

Only the creation and removal of IP sets is limited to the permanent environment, all other IP set options can be used also in the runtime environment without the **--permanent** option.



## WARNING

Red Hat does not recommend using IP sets that are not managed through **firewalld**. To use such IP sets, a permanent direct rule is required to reference the set, and a custom service must be added to create these IP sets. This service needs to be started before firewalld starts, otherwise **firewalld** is not able to add the direct rules using these sets. You can add permanent direct rules with the `/etc/firewalld/direct.xml` file.

## 57.16. PRIORITIZING RICH RULES

By default, rich rules are organized based on their rule action. For example, **deny** rules have precedence over **allow** rules. The **priority** parameter in rich rules provides administrators fine-grained control over rich rules and their execution order.

### 57.16.1. How the priority parameter organizes rules into different chains

You can set the **priority** parameter in a rich rule to any number between **-32768** and **32767**, and lower values have higher precedence.

The **firewalld** service organizes rules based on their priority value into different chains:

- Priority lower than 0: the rule is redirected into a chain with the **\_pre** suffix.
- Priority higher than 0: the rule is redirected into a chain with the **\_post** suffix.
- Priority equals 0: based on the action, the rule is redirected into a chain with the **\_log**, **\_deny**, or **\_allow** the action.

Inside these sub-chains, **firewalld** sorts the rules based on their priority value.

### 57.16.2. Setting the priority of a rich rule

The procedure describes an example of how to create a rich rule that uses the **priority** parameter to log all traffic that is not allowed or denied by other rules. You can use this rule to flag unexpected traffic.

#### Procedure

1. Add a rich rule with a very low precedence to log all traffic that has not been matched by other rules:

```
firewall-cmd --add-rich-rule='rule priority=32767 log prefix="UNEXPECTED: " limit
value="5/m"'
```

The command additionally limits the number of log entries to **5** per minute.

2. Optionally, display the **nftables** rule that the command in the previous step created:

```
nft list chain inet firewalld filter_IN_public_post
table inet firewalld {
```

```

chain filter_IN_public_post {
 log prefix "UNEXPECTED: " limit rate 5/minute
}

```

## 57.17. CONFIGURING FIREWALL LOCKDOWN

Local applications or services are able to change the firewall configuration if they are running as **root** (for example, **libvirt**). With this feature, the administrator can lock the firewall configuration so that either no applications or only applications that are added to the lockdown whitelist are able to request firewall changes. The lockdown settings default to disabled. If enabled, the user can be sure that there are no unwanted configuration changes made to the firewall by local applications or services.

### 57.17.1. Configuring lockdown with using CLI

- To query whether lockdown is enabled, use the following command as **root**:

```
firewall-cmd --query-lockdown
```

The command prints **yes** with exit status **0** if lockdown is enabled. It prints **no** with exit status **1** otherwise.

- To enable lockdown, enter the following command as **root**:

```
firewall-cmd --lockdown-on
```

- To disable lockdown, use the following command as **root**:

```
firewall-cmd --lockdown-off
```

### 57.17.2. Configuring lockdown whitelist options using CLI

The lockdown whitelist can contain commands, security contexts, users and user IDs. If a command entry on the whitelist ends with an asterisk "\*", then all command lines starting with that command will match. If the "\*" is not there then the absolute command including arguments must match.

- The context is the security (SELinux) context of a running application or service. To get the context of a running application use the following command:

```
$ ps -e --context
```

That command returns all running applications. Pipe the output through the **grep** tool to get the application of interest. For example:

```
$ ps -e --context | grep example_program
```

- To list all command lines that are on the whitelist, enter the following command as **root**:

```
firewall-cmd --list-lockdown-whitelist-commands
```

- To add a command *command* to the whitelist, enter the following command as **root**:

```
firewall-cmd --add-lockdown-whitelist-command='/usr/bin/python3 -Es /usr/bin/command'
```

- To remove a command *command* from the whitelist, enter the following command as **root**:

```
firewall-cmd --remove-lockdown-whitelist-command='/usr/bin/python3 -Es /usr/bin/command'
```

- To query whether the command *command* is on the whitelist, enter the following command as **root**:

```
firewall-cmd --query-lockdown-whitelist-command='/usr/bin/python3 -Es /usr/bin/command'
```

The command prints **yes** with exit status **0** if true. It prints **no** with exit status **1** otherwise.

- To list all security contexts that are on the whitelist, enter the following command as **root**:

```
firewall-cmd --list-lockdown-whitelist-contexts
```

- To add a context *context* to the whitelist, enter the following command as **root**:

```
firewall-cmd --add-lockdown-whitelist-context=context
```

- To remove a context *context* from the whitelist, enter the following command as **root**:

```
firewall-cmd --remove-lockdown-whitelist-context=context
```

- To query whether the context *context* is on the whitelist, enter the following command as **root**:

```
firewall-cmd --query-lockdown-whitelist-context=context
```

Prints **yes** with exit status **0**, if true, prints **no** with exit status **1** otherwise.

- To list all user IDs that are on the whitelist, enter the following command as **root**:

```
firewall-cmd --list-lockdown-whitelist-uids
```

- To add a user ID *uid* to the whitelist, enter the following command as **root**:

```
firewall-cmd --add-lockdown-whitelist-uid=uid
```

- To remove a user ID *uid* from the whitelist, enter the following command as **root**:

```
firewall-cmd --remove-lockdown-whitelist-uid=uid
```

- To query whether the user ID *uid* is on the whitelist, enter the following command:

```
$ firewall-cmd --query-lockdown-whitelist-uid=uid
```

Prints **yes** with exit status **0**, if true, prints **no** with exit status **1** otherwise.

- To list all user names that are on the whitelist, enter the following command as **root**:

```
firewall-cmd --list-lockdown-whitelist-users
```

- To add a user name *user* to the whitelist, enter the following command as **root**:

```
firewall-cmd --add-lockdown-whitelist-user=user
```

- To remove a user name *user* from the whitelist, enter the following command as **root**:

```
firewall-cmd --remove-lockdown-whitelist-user=user
```

- To query whether the user name *user* is on the whitelist, enter the following command:

```
$ firewall-cmd --query-lockdown-whitelist-user=user
```

Prints **yes** with exit status **0**, if true, prints **no** with exit status **1** otherwise.

### 57.17.3. Configuring lockdown whitelist options using configuration files

The default whitelist configuration file contains the **NetworkManager** context and the default context of **libvirt**. The user ID 0 is also on the list.

```
<?xml version="1.0" encoding="utf-8"?>
<whitelist>
 <selinux context="system_u:system_r:NetworkManager_t:s0"/>
 <selinux context="system_u:system_r:virtd_t:s0-s0:c0.c1023"/>
 <user id="0"/>
</whitelist>
```

Following is an example whitelist configuration file enabling all commands for the **firewall-cmd** utility, for a user called *user* whose user ID is **815**:

```
<?xml version="1.0" encoding="utf-8"?>
<whitelist>
 <command name="/usr/libexec/platform-python -s /bin/firewall-cmd*"/>
 <selinux context="system_u:system_r:NetworkManager_t:s0"/>
 <user id="815"/>
 <user name="user"/>
</whitelist>
```

This example shows both **user id** and **user name**, but only one option is required. Python is the interpreter and is prepended to the command line. You can also use a specific command, for example:

```
/usr/bin/python3 /bin/firewall-cmd --lockdown-on
```

In that example, only the **--lockdown-on** command is allowed.

In Red Hat Enterprise Linux, all utilities are placed in the **/usr/bin/** directory and the **/bin/** directory is sym-linked to the **/usr/bin/** directory. In other words, although the path for **firewall-cmd** when entered as **root** might resolve to **/bin/firewall-cmd**, **/usr/bin/firewall-cmd** can now be used. All new scripts should use the new location. But be aware that if scripts that run as **root** are written to use the **/bin/firewall-cmd** path, then that command path must be whitelisted in addition to the **/usr/bin/firewall-cmd** path traditionally used only for non- **root** users.

The \* at the end of the name attribute of a command means that all commands that start with this string match. If the \* is not there then the absolute command including arguments must match.

## 57.18. LOG FOR DENIED PACKETS

With the **LogDenied** option in the **firewalld**, it is possible to add a simple logging mechanism for denied packets. These are the packets that are rejected or dropped. To change the setting of the logging, edit the **/etc/firewalld/firewalld.conf** file or use the command-line or GUI configuration tool.

If **LogDenied** is enabled, logging rules are added right before the reject and drop rules in the INPUT, FORWARD and OUTPUT chains for the default rules and also the final reject and drop rules in zones. The possible values for this setting are: **all**, **unicast**, **broadcast**, **multicast**, and **off**. The default setting is **off**. With the **unicast**, **broadcast**, and **multicast** setting, the **pktype** match is used to match the link-layer packet type. With **all**, all packets are logged.

To list the actual **LogDenied** setting with **firewall-cmd**, use the following command as **root**:

```
firewall-cmd --get-log-denied
off
```

To change the **LogDenied** setting, use the following command as **root**:

```
firewall-cmd --set-log-denied=all
success
```

To change the **LogDenied** setting with the **firewalld** GUI configuration tool, start **firewall-config**, click the **Options** menu and select **Change Log Denied**. The **LogDenied** window appears. Select the new **LogDenied** setting from the menu and click OK.

## 57.19. RELATED INFORMATION

The following sources of information provide additional resources regarding **firewalld**.

### Installed documentation

- **firewalld(1)** man page – describes command options for **firewalld**.
- **firewalld.conf(5)** man page – contains information to configure **firewalld**.
- **firewall-cmd(1)** man page – describes command options for the **firewalld** command-line client.
- **firewall-config(1)** man page – describes settings for the **firewall-config** tool.
- **firewall-offline-cmd(1)** man page – describes command options for the **firewalld** offline command-line client.
- **firewalld.icmptype(5)** man page – describes XML configuration files for **ICMP** filtering.
- **firewalld.ipset(5)** man page – describes XML configuration files for the **firewalld** IP sets.
- **firewalld.service(5)** man page – describes XML configuration files for **firewalld** service.
- **firewalld.zone(5)** man page – describes XML configuration files for **firewalld** zone configuration.

- **firewalld.direct(5)** man page – describes the **firewalld** direct interface configuration file.
- **firewalld.lockdown-whitelist(5)** man page – describes the **firewalld** lockdown whitelist configuration file.
- **firewalld.richlanguage(5)** man page – describes the **firewalld** rich language rule syntax.
- **firewalld.zones(5)** man page – general description of what zones are and how to configure them.
- **firewalld.dbus(5)** man page – describes the **D-Bus** interface of **firewalld**.

## Online documentation

- <http://www.firewalld.org/> – **firewalld** home page.

# CHAPTER 58. GETTING STARTED WITH NFTABLES

The **nftables** framework enables administrators to configure packet-filtering rules used by the Linux kernel firewall.

## 58.1. INTRODUCTION TO NFTABLES

The **nftables** framework provides packet classification facilities and it is the designated successor to the **iptables**, **ip6tables**, **arptables**, and **ebtables** tools. It offers numerous improvements in convenience, features, and performance over previous packet-filtering tools, most notably:

- lookup tables instead of linear processing
- a single framework for both the **IPv4** and **IPv6** protocols
- rules all applied atomically instead of fetching, updating, and storing a complete rule set
- support for debugging and tracing in the rule set (**nfttrace**) and monitoring trace events (in the **nft** tool)
- more consistent and compact syntax, no protocol-specific extensions
- a Netlink API for third-party applications

Similarly to **iptables**, **nftables** use tables for storing chains. The chains contain individual rules for performing actions. The **nft** tool replaces all tools from the previous packet-filtering frameworks. The **libnftnl** library can be used for low-level interaction with **nftables** Netlink API over the **libmnl** library.

Effect of the modules on the **nftables** rules set can be observed using the **nft list rule set** command. Since these tools add tables, chains, rules, sets, and other objects to the **nftables** rule set, be aware that **nftables** rule-set operations, such as the **nft flush ruleset** command, might affect rule sets installed using the formerly separate legacy commands.

### Additional resources

- The **nft(8)** man page provides a comprehensive reference documentation for configuring and inspecting packet filtering with nftables using the **nft** command-line tool.

## 58.2. WHEN TO USE FIREWALLD, NFTABLES, OR IPTABLES

The following is a brief overview in which scenario you should use one of the following utilities:

- **firewalld**: Use the **firewalld** utility to configure a firewall on workstations. The utility is easy to use and covers the typical use cases for this scenario.
- **nftables**: Use the **nftables** utility to set up complex firewalls, such as for a whole network.
- **iptables**: The **iptables** utility is deprecated in Red Hat Enterprise Linux 8. Use instead **nftables**.



### IMPORTANT

To avoid that the different firewall services influence each other, run only one of them on a RHEL host, and disable the other services.

## 58.3. CONVERTING IPTABLES RULES TO NFTABLES RULES

Red Hat Enterprise Linux 8 provides the **iptables-translate** and **ip6tables-translate** tools to convert existing **iptables** or **ip6tables** rules into the equivalent ones for **nftables**.

Note that some extensions lack translation support. If such an extension exists, the tool prints the untranslated rule prefixed with the **#** sign. For example:

```
iptables-translate -A INPUT -j CHECKSUM --checksum-fill
nft # -A INPUT -j CHECKSUM --checksum-fill
```

Additionally, users can use the **iptables-restore-translate** and **ip6tables-restore-translate** tools to translate a dump of rules. Note that before that, users can use the **iptables-save** or **ip6tables-save** commands to print a dump of current rules. For example:

```
iptables-save >/tmp/iptables.dump
iptables-restore-translate -f /tmp/iptables.dump

Translated by iptables-restore-translate v1.8.0 on Wed Oct 17 17:00:13 2018
add table ip nat
...
...
```

For more information and a list of possible options and values, enter the **iptables-translate --help** command.

## 58.4. WRITING AND EXECUTING NFTABLES SCRIPTS

The **nftables** framework provides a native scripting environment that brings a major benefit over using shell scripts to maintain firewall rules: the execution of scripts is atomic. This means that the system either applies the whole script or prevents the execution if an error occurs. This guarantees that the firewall is always in a consistent state.

Additionally, the **nftables** script environment enables administrators to:

- add comments
- define variables
- include other rule set files

This section explains how to use these features, as well as creating and executing **nftables** scripts.

When you install the **nftables** package, Red Hat Enterprise Linux automatically creates **\*.nft** scripts in the **/etc/nftables/** directory. These scripts contain commands that create tables and empty chains for different purposes. You can either extend these files or write your scripts.

### 58.4.1. The required script header in nftables script

Similar to other scripts, **nftables** scripts require a shebang sequence in the first line of the script that sets the interpreter directive.

An **nftables** script must always start with the following line:

```
#!/usr/sbin/nft -f
```



## IMPORTANT

If you omit the **-f** parameter, the **nft** utility does not read the script and displays **Error: syntax error, unexpected newline, expecting string**.

### 58.4.2. Supported nftables script formats

The **nftables** scripting environment supports scripts in the following formats:

- You can write a script in the same format as the **nft list ruleset** command displays the rule set:

```
#!/usr/sbin/nft -f

Flush the rule set
flush ruleset

table inet example_table {
 chain example_chain {
 # Chain for incoming packets that drops all packets that
 # are not explicitly allowed by any rule in this chain
 type filter hook input priority 0; policy drop;

 # Accept connections to port 22 (ssh)
 tcp dport ssh accept
 }
}
```

- You can use the same syntax for commands as in **nft** commands:

```
#!/usr/sbin/nft -f

Flush the rule set
flush ruleset

Create a table
add table inet example_table

Create a chain for incoming packets that drops all packets
that are not explicitly allowed by any rule in this chain
add chain inet example_table example_chain { type filter hook input priority 0 ; policy drop ; }

Add a rule that accepts connections to port 22 (ssh)
add rule inet example_table example_chain tcp dport ssh accept
```

### 58.4.3. Running nftables scripts

To run an **nftables** script, the script must be executable. Only if the script is included in another script, it does not require to be executable. The procedure describes how to make a script executable and run the script.

#### Prerequisites

- The procedure of this section assumes that you stored an **nftables** script in the **/etc/nftables/example\_firewall.nft** file.

## Procedure

1. Steps that are required only once:
  - a. Optionally, set the owner of the script to **root**:
 

```
chown root /etc/nftables/example_firewall.nft
```
  - b. Make the script executable for the owner:
 

```
chmod u+x /etc/nftables/example_firewall.nft
```

2. Run the script:

```
/etc/nftables/example_firewall.nft
```

If no output is displayed, the system executed the script successfully.



### IMPORTANT

Even if **nft** executes the script successfully, incorrectly placed rules, missing parameters, or other problems in the script can cause that the firewall behaves not as expected.

## Additional resources

- For details about setting the owner of a file, see the **chown(1)** man page.
- For details about setting permissions of a file, see the **chmod(1)** man page.
- [Section 58.4.7, “Automatically loading nftables rules when the system boots”](#)

### 58.4.4. Using comments in nftables scripts

The **nftables** scripting environment interprets everything to the right of a **#** character as a comment.

#### Example 58.1. Comments in an nftables script

Comments can start at the beginning of a line, as well as next to a command:

```
...
Flush the rule set
flush ruleset

add table inet example_table # Create a table
...
```

### 58.4.5. Using variables in an nftables script

To define a variable in an **nftables** script, use the **define** keyword. You can store single values and anonymous sets in a variable. For more complex scenarios, use sets or verdict maps.

## Variables with a single value

The following example defines a variable named **INET\_DEV** with the value **enp1s0**:

```
define INET_DEV = enp1s0
```

You can use the variable in the script by writing the **\$** sign followed by the variable name:

```
...
add rule inet example_table example_chain iifname $INET_DEV tcp dport ssh accept
...
```

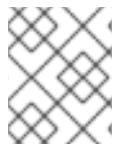
## Variables that contain an anonymous set

The following example defines a variable that contains an anonymous set:

```
define DNS_SERVERS = { 192.0.2.1, 192.0.2.2 }
```

You can use the variable in the script by writing the **\$** sign followed by the variable name:

```
add rule inet example_table example_chain ip daddr $DNS_SERVERS accept
```



### NOTE

Note that curly braces have special semantics when you use them in a rule because they indicate that the variable represents a set.

## Additional resources

- For details about sets, see [Section 58.11, “Using sets in nftables commands”](#) .
- For details about verdict maps, see [Section 58.12, “Using verdict maps in nftables commands”](#) .

### 58.4.6. Including files in an nftables script

The **nftables** scripting environment enables administrators to include other scripts by using the **include** statement.

If you specify only a file name without an absolute or relative path, **nftables** includes files from the default search path, which is set to **/etc** on Red Hat Enterprise Linux.

#### Example 58.2. Including files from the default search directory

To include a file from the default search directory:

```
include "example.nft"
```

#### Example 58.3. Including all \*.nft files from a directory

To include all files ending in **.\*.nft** that are stored in the **/etc/nftables/rulesets/** directory:

```
include "/etc/nftables/rulesets/*.nft"
```

Note that the **include** statement does not match files beginning with a dot.

## Additional resources

- For further details, see the **Include files** section in the **nft(8)** man page.

### 58.4.7. Automatically loading nftables rules when the system boots

The **nftables** systemd service loads firewall scripts that are included in the **/etc/sysconfig/nftables.conf** file. This section explains how to load firewall rules when the system boots.

#### Prerequisites

- The **nftables** scripts are stored in the **/etc/nftables/** directory.

#### Procedure

1. Edit the **/etc/sysconfig/nftables.conf** file.

- If you enhance **\*.nft** scripts created in **/etc/nftables/** when you installed the **nftables** package, uncomment the **include** statement for these scripts.
- If you write scripts from scratch, add **include** statements to include these scripts. For example, to load the **/etc/nftables/example.nft** script when the **nftables** service starts, add:

```
include "/etc/nftables/example.nft"
```

2. Enable the **nftables** service.

```
systemctl enable nftables
```

3. Optionally, start the **nftables** service to load the firewall rules without rebooting the system:

```
systemctl start nftables
```

#### Additional resources

- [Section 58.4.2, "Supported nftables script formats"](#)

## 58.5. DISPLAYING NFTABLES RULE SETS

Rule sets of **nftables** contain tables, chains, and rules. This section explains how to display these rule sets.

#### Procedure

1. To display all rule sets, enter:

```
nft list ruleset
table inet example_table {
 chain example_chain {
 type filter hook input priority 0; policy accept;
```

```
 tcp dport http accept
 tcp dport ssh accept
}
```



## NOTE

By default, **nftables** does not pre-create tables. As a consequence, displaying the rule set on a host without any tables, the **nft list ruleset** command shows no output.

## 58.6. CREATING AN NFTABLES TABLE

A table in **nftables** is a name space that contains a collection of chains, rules, sets, and other objects. This section explains how to create a table.

Each table must have an address family defined. The address family of a table defines what address types the table processes. You can set one of the following address families when you create a table:

- **ip**: Matches only IPv4 packets. This is the default if you do not specify an address family.
  - **ip6**: Matches only IPv6 packets.
  - **inet**: Matches both IPv4 and IPv6 packets.
  - **arp**: Matches IPv4 address resolution protocol (ARP) packets.
  - **bridge**: Matches packets that traverse a bridge device.
  - **netdev**: Matches packets from ingress.

## Procedure

1. Use the **nft add table** command to create a new table. For example, to create a table named **example\_table** that processes IPv4 and IPv6 packets:

```
nft add table inet example table
```

2. Optionally, list all tables in the rule set.

```
nft list tables
```

## Additional resources

- For further details about address families, see the **Address families** section in the **nft(8)** man page.
  - For details on other actions you can run on tables, see the **Tables** section in the **nft(8)** man page.

## 58.7. CREATING AN NFTABLES CHAIN

Chains are containers for rules. The following two rule types exists:

- Base chain: You can use base chains as an entry point for packets from the networking stack.
- Regular chain: You can use regular chains as a **jump** target and to better organize rules.

The procedure describes how to add a base chain to an existing table.

## Prerequisites

- The table to which you want to add the new chain exists.

## Procedure

1. Use the **nft add chain** command to create a new chain. For example, to create a chain named **example\_chain** in **example\_table**:

```
nft add chain inet example_table example_chain { type filter hook input priority 0; policy accept \; }
```



### IMPORTANT

To avoid that the shell interprets the semicolons as the end of the command, you must escape the semicolons with a backslash.

This chain filters incoming packets. The **priority** parameter specifies the order in which **nftables** processes chains with the same hook value. A lower priority value has precedence over higher ones. The **policy** parameter sets the default action for rules in this chain. Note that if you are logged in to the server remotely and you set the default policy to **drop**, you are disconnected immediately if no other rule allows the remote access.

2. Optionally, display all chains:

```
nft list chains
table inet example_table {
 chain example_chain {
 type filter hook input priority 0; policy accept;
 }
}
```

## Additional resources

- For further details about address families, see the **Address families** section in the **nft(8)** man page.
- For details on other actions you can run on chains, see the **Chains** section in the **nft(8)** man page.

## 58.8. ADDING A RULE TO AN NFTABLES CHAIN

This section explains how to add a rule to an existing **nftables** chain. By default, the **nftables add rule** command appends a new rule to the end of the chain.

If you instead want to insert a rule at the beginning of chain, see [Section 58.9, “Inserting a rule into an nftables chain”](#).

## Prerequisites

- The chain to which you want to add the rule exists.

## Procedure

1. To add a new rule, use the **nft add rule** command. For example, to add a rule to the **example\_chain** in the **example\_table** that allows TCP traffic on port 22:

```
nft add rule inet example_table example_chain tcp dport 22 accept
```

Instead of the port number, you can alternatively specify the name of the service. In the example, you could use **ssh** instead of the port number **22**. Note that a service name is resolved to a port number based on its entry in the **/etc/services** file.

2. Optionally, display all chains and their rules in **example\_table**:

```
nft list table inet example_table
table inet example_table {
 chain example_chain {
 type filter hook input priority 0; policy accept;
 ...
 tcp dport ssh accept
 }
}
```

## Additional resources

- For further details about address families, see the **Address families** section in the **nft(8)** man page.
- For details on other actions you can run on rules, see the **Rules** section in the **nft(8)** man page.

## 58.9. INSERTING A RULE INTO AN NFTABLES CHAIN

This section explains how to insert a rule at the beginning of an existing **nftables** chain using the **nftables insert rule** command. If you instead want to add a rule to the end of a chain, see [Section 58.8, "Adding a rule to an nftables chain"](#).

## Prerequisites

- The chain to which you want to add the rule exists.

## Procedure

1. To insert a new rule, use the **nft insert rule** command. For example, to insert a rule to the **example\_chain** in the **example\_table** that allows TCP traffic on port 22:

```
nft add rule inet example_table example_chain tcp dport 22 accept
```

You can alternatively specify the name of the service instead of the port number. In the example, you could use **ssh** instead of the port number **22**. Note that a service name is resolved to a port number based on its entry in the **/etc/services** file.

2. Optionally, display all chains and their rules in **example\_table**:

```
nft list table inet example_table
table inet example_table {
 chain example_chain {
 type filter hook input priority 0; policy accept;
 tcp dport ssh accept
 ...
 }
}
```

## Additional resources

- For further details about address families, see the **Address families** section in the **nft(8)** man page.
- For details on other actions you can run on rules, see the **Rules** section in the **nft(8)** man page.

## 58.10. CONFIGURING NAT USING NFTABLES

With **nftables**, you can configure the following network address translation (NAT) types:

- Masquerading
- Source NAT (SNAT)
- Destination NAT (DNAT)

### 58.10.1. The different NAT types: masquerading, source NAT, and destination NAT

These are the different network address translation (NAT) types:

#### Masquerading and source NAT (SNAT)

Use one of these NAT types to change the source IP address of packets. For example, internet providers do not route reserved IP ranges, such as **10.0.0.0/8**. If you use reserved IP ranges in your network and users should be able to reach servers on the internet, map the source IP address of packets from these ranges to a public IP address.

Both masquerading and SNAT are very similar. The differences are:

- Masquerading automatically uses the IP address of the outgoing interface. Therefore, use masquerading if the outgoing interface uses a dynamic IP address.
- SNAT sets the source IP address of packets to a specified IP and does not dynamically look up the IP of the outgoing interface. Therefore, SNAT is faster than masquerading. Use SNAT if the outgoing interface uses a fixed IP address.

#### Destination NAT (DNAT)

Use this NAT type to route incoming traffic to a different host. For example, if your web server uses an IP address from a reserved IP range and is, therefore, not directly accessible from the internet, you can set a DNAT rule on the router to redirect incoming traffic to this server.

### 58.10.2. Configuring masquerading using nftables

Masquerading enables a router to dynamically change the source IP of packets sent through an interface to the IP address of the interface. This means that if the interface gets a new IP assigned, **nftables** automatically uses the new IP when replacing the source IP.

The following procedure describes how to replace the source IP of packets leaving the host through the **ens3** interface to the IP set on **ens3**.

### Procedure

1. Create a table:

```
nft add table nat
```

2. Add the **prerouting** and **postrouting** chains to the table:

```
nft -- add chain nat prerouting { type nat hook prerouting priority -100 \; }
nft add chain nat postrouting { type nat hook postrouting priority 100 \; }
```



#### IMPORTANT

Even if you do not add a rule to the **prerouting** chain, the **nftables** framework requires this chain to match incoming packet replies.

Note that you must pass the **--** option to the **nft** command to avoid that the shell interprets the negative priority value as an option of the **nft** command.

3. Add a rule to the **postrouting** chain that matches outgoing packets on the **ens3** interface:

```
nft add rule nat postrouting oifname "ens3" masquerade
```

### 58.10.3. Configuring source NAT using nftables

On a router, Source NAT (SNAT) enables you to change the IP of packets sent through an interface to a specific IP address.

The following procedure describes how to replace the source IP of packets leaving the router through the **ens3** interface to **192.0.2.1**.

### Procedure

1. Create a table:

```
nft add table nat
```

2. Add the **prerouting** and **postrouting** chains to the table:

```
nft -- add chain nat prerouting { type nat hook prerouting priority -100 \; }
nft add chain nat postrouting { type nat hook postrouting priority 100 \; }
```



## IMPORTANT

Even if you do not add a rule to the **postrouting** chain, the **nftables** framework requires this chain to match outgoing packet replies.

Note that you must pass the **--** option to the **nft** command to avoid that the shell interprets the negative priority value as an option of the **nft** command.

3. Add a rule to the **postrouting** chain that replaces the source IP of outgoing packets through **ens3** with **192.0.2.1**:

```
nft add rule nat postrouting oifname "ens3" snat to 192.0.2.1
```

### Additional resources

- [Section 58.13.2, “Forwarding incoming packets on a specific local port to a different host”](#)

#### 58.10.4. Configuring destination NAT using nftables

Destination NAT enables you to redirect traffic on a router to a host that is not directly accessible from the internet.

The following procedure describes how to redirect incoming traffic sent to port **80** and **443** of the router to the host with the **192.0.2.1** IP address.

### Procedure

1. Create a table:

```
nft add table nat
```

2. Add the **prerouting** and **postrouting** chains to the table:

```
nft -- add chain nat prerouting { type nat hook prerouting priority -100 \; }
nft add chain nat postrouting { type nat hook postrouting priority 100 \; }
```



## IMPORTANT

Even if you do not add a rule to the **postrouting** chain, the **nftables** framework requires this chain to match outgoing packet replies.

Note that you must pass the **--** option to the **nft** command to avoid that the shell interprets the negative priority value as an option of the **nft** command.

3. Add a rule to the **prerouting** chain that redirects incoming traffic on the **ens3** interface sent to port **80** and **443** to the host with the **192.0.2.1** IP:

```
nft add rule nat prerouting iifname ens3 tcp dport { 80, 443 } dnat to 192.0.2.1
```

4. Depending on your environment, add either a SNAT or masquerading rule to change the source address:

- a. If the **ens3** interface used dynamic IP addresses, add a masquerading rule:

```
nft add rule nat postrouting oifname "ens3" masquerade
b. If the ens3 interface uses a static IP address, add a SNAT rule. For example, if the ens3 uses the 198.51.100.1 IP address:
 nft add rule nat postrouting oifname "ens3" snat to 198.51.100.1
```

## Additional resources

- [Section 58.10.1, “The different NAT types: masquerading, source NAT, and destination NAT”](#)

## 58.11. USING SETS IN NFTABLES COMMANDS

The **nftables** framework natively supports sets. You can use sets, for example, if a rule should match multiple IP addresses, port numbers, interfaces, or any other match criteria.

### 58.11.1. Using an anonymous sets in nftables

An anonymous set contain comma-separated values enclosed in curly brackets, such as **{ 22, 80, 443 }**, that you use directly in a rule. You can also use anonymous sets also for IP addresses or any other match criteria.

The drawback of anonymous sets is that if you want to change the set, you must replace the rule. For a dynamic solution, use named sets as described in [Section 58.11.2, “Using named sets in nftables”](#).

#### Prerequisites

- The **example\_chain** chain and the **example\_table** table in the **inet** family exists.

#### Procedure

1. For example, to add a rule to **example\_chain** in **example\_table** that allows incoming traffic to port **22**, **80**, and **443**:

```
nft add rule inet example_table example_chain tcp dport { 22, 80, 443 } accept
```

2. Optionally, display all chains and their rules in **example\_table**:

```
nft list table inet example_table
table inet example_table {
 chain example_chain {
 type filter hook input priority 0; policy accept;
 tcp dport { ssh, http, https } accept
 }
}
```

### 58.11.2. Using named sets in nftables

The **nftables** framework supports mutable named sets. A named set is a list or range of elements that you can use in multiple rules within a table. Another benefit over anonymous sets is that you can update a named set without replacing the rules that use the set.

When you create a named set, you must specify the type of elements the set contains. You can set the following types:

- **ipv4\_addr** for a set that contains IPv4 addresses or ranges, such as **192.0.2.1** or **192.0.2.0/24**.
- **ipv6\_addr** for a set that contains IPv6 addresses or ranges, such as **2001:db8::1** or **2001:db8::1/24**.
- **ether\_addr** for a set that contains a list of media access control (MAC) addresses, such as **52:54:00:6b:66:42**.
- **inet\_proto** for a set that contains a list of internet protocol types, such as **tcp**.
- **inet\_service** for a set that contains a list of internet services, such as **ssh**.
- **mark** for a set that contains a list of packet marks. Packet marks can be any positive 32-bit integer value (**0** to **2147483647**).

## Prerequisites

- The **example\_chain** chain and the **example\_table** table exists.

## Procedure

1. Create an empty set. The following examples create a set for IPv4 addresses:

- To create a set that can store multiple individual IPv4 addresses:

```
nft add set inet example_table example_set { type ipv4_addr \; }
```

- To create a set that can store IPv4 address ranges:

```
nft add set inet example_table example_set { type ipv4_addr \; flags interval \; }
```



### IMPORTANT

To avoid that the shell interprets the semicolons as the end of the command, you must escape the semicolons with a backslash.

2. Optionally, create rules that use the set. For example, the following command adds a rule to the **example\_chain** in the **example\_table** that will drop all packets from IPv4 addresses in **example\_set**.

```
nft add rule inet example_table example_chain ip saddr @example_set drop
```

Because **example\_set** is still empty, the rule has currently no effect.

3. Add IPv4 addresses to **example\_set**:

- If you create a set that stores individual IPv4 addresses, enter:

```
nft add element inet example_table example_set { 192.0.2.1, 192.0.2.2 }
```

- If you create a set that stores IPv4 ranges, enter:

```
nft add element inet example_table example_set{ 192.0.2.0-192.0.2.255}
```

When you specify an IP address range, you can alternatively use the Classless Inter-Domain Routing (CIDR) notation, such as **192.0.2.0/24** in the above example.

### 58.11.3. Related information

- For further details about sets, see the **Sets** section in the **nft(8)** man page.

## 58.12. USING VERDICT MAPS IN NFTABLES COMMANDS

Verdict maps, which are also known as dictionaries, enable **nft** to perform an action based on packet information by mapping match criteria to an action.

### 58.12.1. Using literal maps in nftables

A literal map is a **{ match\_criteria : action }** statement that you use directly in a rule. The statement can contain multiple comma-separated mappings.

The drawback of a literal map is that if you want to change the map, you must replace the rule. For a dynamic solution, use named verdict maps as described in [Section 58.12.2, “Using mutable verdict maps in nftables”](#).

The example describes how to use a literal map to route both TCP and UDP packets of the IPv4 and IPv6 protocol to different chains to count incoming TCP and UDP packets separately.

#### Procedure

1. Create the **example\_table**:

```
nft add table inet example_table
```

2. Create the **tcp\_packets** chain in **example\_table**:

```
nft add chain inet example_table tcp_packets
```

3. Add a rule to **tcp\_packets** that counts the traffic in this chain:

```
nft add rule inet example_table tcp_packets counter
```

4. Create the **udp\_packets** chain in **example\_table**

```
nft add chain inet example_table udp_packets
```

5. Add a rule to **udp\_packets** that counts the traffic in this chain:

```
nft add rule inet example_table udp_packets counter
```

6. Create a chain for incoming traffic. For example, to create a chain named **incoming\_traffic** in **example\_table** that filters incoming traffic:

```
nft add chain inet example_table incoming_traffic { type filter hook input priority 0 \; }
```

7. Add a rule with a literal map to **incoming\_traffic**:

```
nft add rule inet example_table incoming_traffic ip protocol vmap { tcp : jump tcp_packets,
 udp : jump udp_packets }
```

The literal map distinguishes the packets and sends them to the different counter chains based on their protocol.

8. To list the traffic counters, display **example\_table**:

```
nft list table inet example_table
table inet example_table {
 chain tcp_packets {
 counter packets 36379 bytes 2103816
 }

 chain udp_packets {
 counter packets 10 bytes 1559
 }

 chain incoming_traffic {
 type filter hook input priority 0; policy accept;
 ip protocol vmap { tcp : jump tcp_packets, udp : jump udp_packets }
 }
}
```

The counters in the **tcp\_packets** and **udp\_packets** chain display both the number of received packets and bytes.

### 58.12.2. Using mutable verdict maps in nftables

The **nftables** framework supports mutable verdict maps. You can use these maps in multiple rules within a table. Another benefit over literal maps is that you can update a mutable map without replacing the rules that use it.

When you create a mutable verdict map, you must specify the type of elements

- **ipv4\_addr** for a map whose match part contains an IPv4 address, such as **192.0.2.1**.
- **ipv6\_addr** for a map whose match part contains an IPv6 address, such as **2001:db8::1**.
- **ether\_addr** for a map whose match part contains a media access control (MAC) address, such as **52:54:00:6b:66:42**.
- **inet\_proto** for a map whose match part contains an internet protocol type, such as **tcp**.
- **inet\_service** for a map whose match part contains an internet services name port number, such as **ssh** or **22**.
- **mark** for a map whose match part contains a packet mark. A packet mark can be any positive 32-bit integer value (**0** to **2147483647**).
- **counter** for a map whose match part contains a counter value. The counter value can be any positive 64-bit integer value.

- **quota** for a map whose match part contains a quota value. The quota value can be any positive 64-bit integer value.

The example describes how to allow or drop incoming packets based on their source IP address. Using a mutable verdict map, you require only a single rule to configure this scenario while the IP addresses and actions are dynamically stored in the map. The procedure also describes how to add and remove entries from the map.

## Procedure

1. Create a table. For example, to create a table named **example\_table** that processes IPv4 packets:

```
nft add table ip example_table
```

2. Create a chain. For example, to create a chain named **example\_chain** in **example\_table**:

```
nft add chain ip example_table example_chain { type filter hook input priority 0 \; }
```



### IMPORTANT

To avoid that the shell interprets the semicolons as the end of the command, you must escape the semicolons with a backslash.

3. Create an empty map. For example, to create a map for IPv4 addresses:

```
nft add map ip example_table example_map { type ipv4_addr : verdict \; }
```

4. Create rules that use the map. For example, the following command adds a rule to **example\_chain** in **example\_table** that applies actions to IPv4 addresses which are both defined in **example\_map**:

```
nft add rule example_table example_chain ip saddr vmap @example_map
```

5. Add IPv4 addresses and corresponding actions to **example\_map**:

```
nft add element ip example_table example_map { 192.0.2.1 : accept, 192.0.2.2 : drop }
```

This example defines the mappings of IPv4 addresses to actions. In combination with the rule created above, the firewall accepts packet from **192.0.2.1** and drops packets from **192.0.2.2**.

6. Optionally, enhance the map by adding another IP address and action statement:

```
nft add element ip example_table example_map { 192.0.2.3 : accept }
```

7. Optionally, remove an entry from the map:

```
nft delete element ip example_table example_map { 192.0.2.1 }
```

8. Optionally, display the rule set:

```
nft list ruleset
```

```

table ip example_table {
 map example_map {
 type ipv4_addr : verdict
 elements = { 192.0.2.2 : drop, 192.0.2.3 : accept }
 }

 chain example_chain {
 type filter hook input priority 0; policy accept;
 ip saddr vmap @example_map
 }
}

```

### 58.12.3. Related information

- For further details about verdict maps, see the **Maps** section in the **nft(8)** man page.

## 58.13. CONFIGURING PORT FORWARDING USING NFTABLES

Port forwarding enables administrators to forward packets sent to a specific destination port to a different local or remote port.

For example, if your web server does not have a public IP address, you can set a port forwarding rule on your firewall that forwards incoming packets on port **80** and **443** on the firewall to the web server. With this firewall rule, users on the internet can access the web server using the IP or host name of the firewall.

### 58.13.1. Forwarding incoming packets to a different local port

This section describes an example of how to forward incoming IPv4 packets on port **8022** to port **22** on the local system.

#### Procedure

- Create a table named **nat** with the **ip** address family:

```
nft add table ip nat
```

- Add the **prerouting** and **postrouting** chains to the table:

```
nft -- add chain ip nat prerouting { type nat hook prerouting priority -100 \; }
```



#### NOTE

Pass the **--** option to the **nft** command to avoid that the shell interprets the negative priority value as an option of the **nft** command.

- Add a rule to the **prerouting** chain that redirects incoming packets on port **8022** to the local port **22**:

```
nft add rule ip nat prerouting tcp dport 8022 redirect to :22
```

### 58.13.2. Forwarding incoming packets on a specific local port to a different host

You can use a destination network address translation (DNAT) rule to forward incoming packets on a local port to a remote host. This enables users on the internet to access a service that runs on a host with a private IP address.

The procedure describes how to forward incoming IPv4 packets on the local port **443** to the same port number on the remote system with the **192.0.2.1** IP address.

### Prerequisite

- You are logged in as the **root** user on the system that should forward the packets.

### Procedure

1. Create a table named **nat** with the **ip** address family:

```
nft add table ip nat
```

2. Add the **prerouting** and **postrouting** chains to the table:

```
nft -- add chain ip nat prerouting { type nat hook prerouting priority -100 \; }
nft add chain ip nat postrouting { type nat hook postrouting priority 100 \; }
```



#### NOTE

Pass the **--** option to the **nft** command to avoid that the shell interprets the negative priority value as an option of the **nft** command.

3. Add a rule to the **prerouting** chain that redirects incoming packets on port **443** to the same port on **192.0.2.1**:

```
nft add rule ip nat prerouting tcp dport 443 dnat to 192.0.2.1
```

4. Add a rule to the **postrouting** chain to masquerade outgoing traffic:

```
nft add rule ip daddr 192.0.2.1 masquerade
```

5. Enable packet forwarding:

```
echo "net.ipv4.ip_forward=1" > /etc/sysctl.d/95-IPv4-forwarding.conf
sysctl -p /etc/sysctl.d/95-IPv4-forwarding.conf
```

## 58.14. LIMITING THE NUMBER OF CONNECTIONS USING NFTABLES

The **ct count** parameter of the **nft** utility enables administrators to limit the number of connections. The procedure describes a basic example of how to limit incoming connections.

### Prerequisites

- The base **example\_chain** in **example\_table** exists.

### Procedure

1. Add a rule that allows only two simultaneous connections to the SSH port (22) from an IPv4 address and rejects all further connections from the same IP:

```
nft add rule ip example_table example_chain tcp dport ssh meter example_meter { ip saddr ct count over 2 } counter reject
```

2. Optionally, display the meter created in the previous step:

```
nft list meter ip example_table example_meter
table ip example_table {
 meter example_meter {
 type ipv4_addr
 size 65535
 elements = { 192.0.2.1 : ct count over 2 , 192.0.2.2 : ct count over 2 }
 }
}
```

The **elements** entry displays addresses that currently match the rule. In this example, **elements** lists IP addresses that have active connections to the SSH port. Note that the output does not display the number of active connections or if connections were rejected.

## 58.15. BLOCKING IP ADDRESSES THAT ATTEMPT MORE THAN TEN NEW INCOMING TCP CONNECTIONS WITHIN ONE MINUTE

The **nftables** framework enables administrators to dynamically update sets. This section explains how you use this feature to temporarily block hosts that are establishing more than ten IPv4 TCP connections within one minute. After five minutes, **nftables** automatically removes the IP address from the blacklist.

### Procedure

1. Create the **filter** table with the **ip** address family:

```
nft add table ip filter
```

2. Add the **input** chain to the **filter** table:

```
nft add chain ip filter input { type filter hook input priority 0 };
```

3. Add a set named **blacklist** to the **filter** table:

```
nft add set ip filter blacklist { type ipv4_addr ; flags dynamic, timeout ; timeout 5m ; }
```

This command creates a dynamic set for IPv4 addresses. The **timeout 5m** parameter defines that **nftables** automatically removes entries after 5 minutes from the set.

4. Add a rule that automatically adds the source IP address of hosts that attempt to establish more than ten new TCP connections within one minute to the **blacklist** set:

```
nft add rule ip filter input ip protocol tcp ct state new, untracked limit rate over 10/minute
add @blacklist { ip saddr }
```

5. Add a rule that drops all connections from IP addresses in the **blacklist** set:

```
nft add rule ip filter input ip saddr @blacklist drop
```

## Additional resources

- [Section 58.11.2, “Using named sets in nftables”](#)

# 58.16. DEBUGGING NFTABLES RULES

The **nftables** framework provides different options for administrators to debug rules and if packets match them. This section describes these options.

## 58.16.1. Creating a rule with a counter

To identify if a rule is matched, you can use a counter. This section describes how to create a new rule with a counter.

For a procedure that adds a counter to an existing rule, see [Section 58.16.2, “Adding a counter to an existing rule”](#).

### Prerequisites

- The chain to which you want to add the rule exists.

### Procedure

1. Add a new rule with the **counter** parameter to the chain. The following example adds a rule with a counter that allows TCP traffic on port 22 and counts the packets and traffic that match this rule:

```
nft add rule inet example_table example_chain tcp dport 22 counter accept
```

2. To display the counter values:

```
nft list ruleset
table inet example_table {
 chain example_chain {
 type filter hook input priority 0; policy accept;
 tcp dport ssh counter packets 6872 bytes 105448565 accept
 }
}
```

## 58.16.2. Adding a counter to an existing rule

To identify if a rule is matched, you can use a counter. This section describes how to add a counter to an existing rule.

For a procedure to add a new rule with a counter, see [Section 58.16.1, “Creating a rule with a counter”](#).

### Prerequisites

- The rule to which you want to add the counter exists.

## Procedure

1. Display the rules in the chain including their handles:

```
nft --handle list chain inet example_table example_chain
table inet example_table {
 chain example_chain { # handle 1
 type filter hook input priority 0; policy accept;
 tcp dport ssh accept # handle 4
 }
}
```

2. Add the counter by replacing the rule but with the **counter** parameter. The following example replaces the rule displayed in the previous step and adds a counter:

```
nft replace rule inet example_table example_chain handle 4 tcp dport 22 counter accept
```

3. To display the counter values:

```
nft list ruleset
table inet example_table {
 chain example_chain {
 type filter hook input priority 0; policy accept;
 tcp dport ssh counter packets 6872 bytes 105448565 accept
 }
}
```

### 58.16.3. Monitoring packets that match an existing rule

The tracing feature in **nftables** in combination with the **nft monitor** command enables administrators to display packets that match a rule. The procedure describes how to enable tracing for a rule as well as monitoring packets that match this rule.

## Prerequisites

- The rule to which you want to add the counter exists.

## Procedure

1. Display the rules in the chain including their handles:

```
nft --handle list chain inet example_table example_chain
table inet example_table {
 chain example_chain { # handle 1
 type filter hook input priority 0; policy accept;
 tcp dport ssh accept # handle 4
 }
}
```

2. Add the tracing feature by replacing the rule but with the **meta nftrace set 1** parameters. The following example replaces the rule displayed in the previous step and enables tracing:

```
nft replace rule inet example_table example_chain handle 4 tcp dport 22 meta nftrace set
1 accept
```

3. Use the **nft monitor** command to display the tracing. The following example filters the output of the command to display only entries that contain **inet example\_table example\_chain**:

```
nft monitor | grep "inet example_table example_chain"
trace id 3c5eb15e inet example_table example_chain packet: iif "enp1s0" ether saddr
52:54:00:17:ff:e4 ether daddr 52:54:00:72:2f:6e ip saddr 192.0.2.1 ip daddr 192.0.2.2 ip dscp
cs0 ip ecn not-ect ip ttl 64 ip id 49710 ip protocol tcp ip length 60 tcp sport 56728 tcp dport
ssh tcp flags == syn tcp window 64240
trace id 3c5eb15e inet example_table example_chain rule tcp dport ssh ntrace set 1 accept
(verdict accept)
...
```



### WARNING

Depending on the number of rules with tracing enabled and the amount of matching traffic, the **nft monitor** command can display a lot of output. Use **grep** or other utilities to filter the output.

## 58.17. BACKING UP AND RESTORING NFTABLES RULE SETS

This section describes how to backup **nftables** rules to a file, as well as restoring rules from a file.

Administrators can use a file with the rules to, for example, transfer the rules to a different server.

### 58.17.1. Backing up nftables rule sets to a file

This section describes how to back up **nftables** rule sets to a file.

#### Procedure

1. To backup **nftables** rules:

- In **nft list ruleset** format:

```
nft list ruleset > file.nft
```

- In JSON format:

```
nft -j list ruleset > file.json
```

### 58.17.2. Restoring nftables rule sets from a file

This section describes how to restore **nftables** rule sets.

#### Procedure

1. To restore **nftables** rules:

- If the file to restore is in **nft list ruleset** format or contains **nft** commands:

```
nft -f file.nft
```

- If the file to restore is in JSON format:

```
nft -j -f file.json
```

## 58.18. RELATED INFORMATION

- The [Using nftables in Red Hat Enterprise Linux 8](#) blog post provides an overview about using **nftables** features.
- The [What comes after iptables? Its successor, of course: nftables](#) article explains why **nftables** replaces **iptables**.
- The [Firewalld: The Future is nftables](#) article provides additional information on **nftables** as a default back end for **firewalld**.

## PART IV. DESIGN OF HARD DISK

# CHAPTER 59. OVERVIEW OF AVAILABLE FILE SYSTEMS

Choosing the file system that is appropriate for your application is an important decision due to the large number of options available and the trade-offs involved. This chapter describes some of the file systems that ship with Red Hat Enterprise Linux 8 and provides historical background and recommendations on the right file system to suit your application.

## 59.1. TYPES OF FILE SYSTEMS

Red Hat Enterprise Linux 8 supports a variety of file systems (FS). Different types of file systems solve different kinds of problems, and their usage is application specific. At the most general level, available file systems can be grouped into the following major types:

**Table 59.1. Types of file systems and their use cases**

Type	File system	Attributes and use cases
Disk or local FS	XFS	XFS is the default file system in RHEL. Because it lays out files as extents, it is less vulnerable to fragmentation than ext4. Red Hat recommends deploying XFS as your local file system unless there are specific reasons to do otherwise; for example, compatibility or corner cases around performance.
	ext4	ext4 has the benefit of longevity in Linux. Therefore, it is supported by almost all Linux applications. In most cases, it rivals XFS on performance. ext4 is commonly used for home directories.
Network or client-and-server FS	NFS	Use NFS to share files between multiple systems on the same network.
	SMB	Use SMB for file sharing with Microsoft Windows systems.
Shared storage or shared disk FS	GFS2	GFS2 provides shared write access to members of a compute cluster. The emphasis is on stability and reliability, with the functional experience of a local file system as possible. SAS Grid, Tibco MQ, IBM Websphere MQ, and Red Hat Active MQ have been deployed successfully on GFS2.
Volume-managing FS	Stratis (Technology Preview)	Stratis is a volume manager built on a combination of XFS and LVM. The purpose of Stratis is to emulate capabilities offered by volume-managing file systems like Btrfs and ZFS. It is possible to build this stack manually, but Stratis reduces configuration complexity, implements best practices, and consolidates error information.

## 59.2. LOCAL FILE SYSTEMS

Local file systems are file systems that run on a single, local server and are directly attached to storage.

For example, a local file system is the only choice for internal SATA or SAS disks, and is used when your server has internal hardware RAID controllers with local drives. Local file systems are also the most common file systems used on SAN attached storage when the device exported on the SAN is not shared.

All local file systems are POSIX-compliant and are fully compatible with all supported Red Hat Enterprise Linux releases. POSIX-compliant file systems provide support for a well-defined set of system calls, such as **read()**, **write()**, and **seek()**.

From the application programmer's point of view, there are relatively few differences between local file systems. The most notable differences from a user's perspective are related to scalability and performance. When considering a file system choice, consider how large the file system needs to be, what unique features it should have, and how it performs under your workload.

## Available local file systems

- XFS
- ext4

### 59.3. THE XFS FILE SYSTEM

XFS is a highly scalable, high-performance, robust, and mature 64-bit journaling file system that supports very large files and file systems on a single host. It is the default file system in Red Hat Enterprise Linux 8. XFS was originally developed in the early 1990s by SGI and has a long history of running on extremely large servers and storage arrays.

The features of XFS include:

#### Reliability

- Metadata journaling, which ensures file system integrity after a system crash by keeping a record of file system operations that can be replayed when the system is restarted and the file system remounted
- Extensive run-time metadata consistency checking
- Scalable and fast repair utilities
- Quota journaling. This avoids the need for lengthy quota consistency checks after a crash.

#### Scalability and performance

- Supported file system size up to 1024 TiB
- Ability to support a large number of concurrent operations
- B-tree indexing for scalability of free space management
- Sophisticated metadata read-ahead algorithms
- Optimizations for streaming video workloads

#### Allocation schemes

— . . . . .

- Extent-based allocation
- Stripe-aware allocation policies
- Delayed allocation
- Space pre-allocation
- Dynamically allocated inodes

### Other features

- Reflink-based file copies (new in Red Hat Enterprise Linux 8)
- Tightly integrated backup and restore utilities
- Online defragmentation
- Online file system growing
- Comprehensive diagnostics capabilities
- Extended attributes (**xattr**). This allows the system to associate several additional name/value pairs per file.
- Project or directory quotas. This allows quota restrictions over a directory tree.
- Subsecond timestamps

### Performance characteristics

XFS has a high performance on large systems with enterprise workloads. A large system is one with a relatively high number of CPUs, multiple HBAs, and connections to external disk arrays. XFS also performs well on smaller systems that have a multi-threaded, parallel I/O workload.

XFS has a relatively low performance for single threaded, metadata-intensive workloads: for example, a workload that creates or deletes large numbers of small files in a single thread.

## 59.4. THE EXT4 FILE SYSTEM

The ext4 file system is the fourth generation of the ext file system family. It was the default file system in Red Hat Enterprise Linux 6.

The ext4 driver can read and write to ext2 and ext3 file systems, but the ext4 file system format is not compatible with ext2 and ext3 drivers.

ext4 adds several new and improved features, such as:

- Supported file system size up to 50 TiB
- Extent-based metadata
- Delayed allocation
- Journal checksumming
- Large storage support

The extent-based metadata and the delayed allocation features provide a more compact and efficient way to track utilized space in a file system. These features improve file system performance and reduce the space consumed by metadata. Delayed allocation allows the file system to postpone selection of the permanent location for newly written user data until the data is flushed to disk. This enables higher performance since it can allow for larger, more contiguous allocations, allowing the file system to make decisions with much better information.

File system repair time using the **fsck** utility in ext4 is much faster than in ext2 and ext3. Some file system repairs have demonstrated up to a six-fold increase in performance.

## 59.5. COMPARISON OF XFS AND EXT4

XFS is the default file system in RHEL. This section compares the usage and features of XFS and ext4.

### Metadata error behavior

In ext4, you can configure the behavior when the file system encounters metadata errors. The default behavior is to simply continue the operation. When XFS encounters an unrecoverable metadata error, it shuts down the file system and returns the **EFSCORRUPTED** error.

### Quotas

In ext4, you can enable quotas when creating the file system or later on an existing file system. You can then configure the quota enforcement using a mount option.

XFS quotas are not a remountable option. You must activate quotas on the initial mount.

Running the **quotacheck** command on an XFS file system has no effect. The first time you turn on quota accounting, XFS checks quotas automatically.

### File system resize

XFS has no utility to reduce the size of a file system. You can only increase the size of an XFS file system. In comparison, ext4 supports both extending and reducing the size of a file system.

### Inode numbers

The ext4 file system does not support more than  $2^{32}$  inodes.

XFS dynamically allocates inodes. An XFS file system cannot run out of inodes as long as there is free space on the file system.

Certain applications cannot properly handle inode numbers larger than  $2^{32}$  on an XFS file system. These applications might cause the failure of 32-bit stat calls with the **E OVERFLOW** return value.

Inode number exceed  $2^{32}$  under the following conditions:

- The file system is larger than 1 TiB with 256-byte inodes.
- The file system is larger than 2 TiB with 512-byte inodes.

If your application fails with large inode numbers, mount the XFS file system with the **-o inode32** option to enforce inode numbers below  $2^{32}$ . Note that using **inode32** does not affect inodes that are already allocated with 64-bit numbers.



### IMPORTANT

*Do not use the **inode32** option unless a specific environment requires it. The **inode32** option changes allocation behavior. As a consequence, the **ENOSPC** error might occur if no space is available to allocate inodes in the lower disk blocks.*

## 59.6. CHOOSING A LOCAL FILE SYSTEM

To choose a file system that meets your application requirements, you need to understand the target system on which you are going to deploy the file system. You can use the following questions to inform your decision:

- Do you have a large server?
- Do you have large storage requirements or have a local, slow SATA drive?
- What kind of I/O workload do you expect your application to present?
- What are your throughput and latency requirements?
- How stable is your server and storage hardware?
- What is the typical size of your files and data set?
- If the system fails, how much downtime can you suffer?

If both your server and your storage device are large, XFS is the best choice. Even with smaller storage arrays, XFS performs very well when the average file sizes are large (for example, hundreds of megabytes in size).

If your existing workload has performed well with ext4, staying with ext4 should provide you and your applications with a very familiar environment.

The ext4 file system tends to perform better on systems that have limited I/O capability. It performs better on limited bandwidth (less than 200MB/s) and up to around 1000 IOPS capability. For anything with higher capability, XFS tends to be faster.

XFS consumes about twice the CPU-per-metadata operation compared to ext4, so if you have a CPU-bound workload with little concurrency, then ext4 will be faster. In general, ext4 is better if an application uses a single read/write thread and small files, while XFS shines when an application uses multiple read/write threads and bigger files.

You cannot shrink an XFS file system. If you need to be able to shrink the file system, consider using ext4, which supports offline shrinking.

In general, Red Hat recommends that you use XFS unless you have a specific use case for ext4. You should also measure the performance of your specific application on your target server and storage system to make sure that you choose the appropriate type of file system.

**Table 59.2. Summary of local file system recommendations**

Scenario	Recommended file system
No special use case	XFS
Large server	XFS
Large storage devices	XFS
Large files	XFS

Scenario	Recommended file system
Multi-threaded I/O	XFS
Single-threaded I/O	ext4
Limited I/O capability (under 1000 IOPS)	ext4
Limited bandwidth (under 200MB/s)	ext4
CPU-bound workload	ext4
Support for offline shrinking	ext4

## 59.7. NETWORK FILE SYSTEMS

Network file systems, also referred to as client/server file systems, enable client systems to access files that are stored on a shared server. This makes it possible for multiple users on multiple systems to share files and storage resources.

Such file systems are built from one or more servers that export a set of file systems to one or more clients. The client nodes do not have access to the underlying block storage, but rather interact with the storage using a protocol that allows for better access control.

### Available network file systems

- The most common client/server file system for RHEL customers is the NFS file system. RHEL provides both an NFS server component to export a local file system over the network and an NFS client to import these file systems.
- RHEL also includes a CIFS client that supports the popular Microsoft SMB file servers for Windows interoperability. The userspace Samba server provides Windows clients with a Microsoft SMB service from a RHEL server.

## 59.8. SHARED STORAGE FILE SYSTEMS

Shared storage file systems, sometimes referred to as cluster file systems, give each server in the cluster direct access to a shared block device over a local storage area network (SAN).

### Comparison with network file systems

Like client/server file systems, shared storage file systems work on a set of servers that are all members of a cluster. Unlike NFS, however, no single server provides access to data or metadata to other members: each member of the cluster has direct access to the same storage device (the *shared storage*), and all cluster member nodes access the same set of files.

### Concurrency

Cache coherency is key in a clustered file system to ensure data consistency and integrity. There must be a single version of all files in a cluster visible to all nodes within a cluster. The file system must prevent members of the cluster from updating the same storage block at the same time and causing data corruption. In order to do that, shared storage file systems use a cluster wide-locking mechanism to

arbitrate access to the storage as a concurrency control mechanism. For example, before creating a new file or writing to a file that is opened on multiple servers, the file system component on the server must obtain the correct lock.

The requirement of cluster file systems is to provide a highly available service like an Apache web server. Any member of the cluster will see a fully coherent view of the data stored in their shared disk file system, and all updates will be arbitrated correctly by the locking mechanisms.

### Performance characteristics

Shared disk file systems do not always perform as well as local file systems running on the same system due to the computational cost of the locking overhead. Shared disk file systems perform well with workloads where each node writes almost exclusively to a particular set of files that are not shared with other nodes or where a set of files is shared in an almost exclusively read-only manner across a set of nodes. This results in a minimum of cross-node cache invalidation and can maximize performance.

Setting up a shared disk file system is complex, and tuning an application to perform well on a shared disk file system can be challenging.

### Available shared storage file systems

- Red Hat Enterprise Linux provides the GFS2 file system. GFS2 comes tightly integrated with the Red Hat Enterprise Linux High Availability Add-On and the Resilient Storage Add-On. Red Hat Enterprise Linux supports GFS2 on clusters that range in size from 2 to 16 nodes.

## 59.9. CHOOSING BETWEEN NETWORK AND SHARED STORAGE FILE SYSTEMS

When choosing between network and shared storage file systems, consider the following points:

- NFS-based network file systems are an extremely common and popular choice for environments that provide NFS servers.
- Network file systems can be deployed using very high-performance networking technologies like Infiniband or 10 Gigabit Ethernet. This means that you should not turn to shared storage file systems just to get raw bandwidth to your storage. If the speed of access is of prime importance, then use NFS to export a local file system like XFS.
- Shared storage file systems are not easy to set up or to maintain, so you should deploy them only when you cannot provide your required availability with either local or network file systems.
- A shared storage file system in a clustered environment helps reduce downtime by eliminating the steps needed for unmounting and mounting that need to be done during a typical fail-over scenario involving the relocation of a high-availability service.

Red Hat recommends that you use network file systems unless you have a specific use case for shared storage file systems. Use shared storage file systems primarily for deployments that need to provide high-availability services with minimum downtime and have stringent service-level requirements.

## 59.10. VOLUME-MANAGING FILE SYSTEMS

Volume-managing file systems integrate the entire storage stack for the purposes of simplicity and in-stack optimization.

### Available volume-managing file systems

- Red Hat Enterprise Linux 8 provides the Stratis volume manager as a Technology Preview. Stratis uses XFS for the file system layer and integrates it with LVM, Device Mapper, and other components. Stratis was first released in Red Hat Enterprise Linux 8.0. It is conceived to fill the gap created when Red Hat deprecated Btrfs. Stratis 1.0 is an intuitive, command line-based volume manager that can perform significant storage management operations while hiding the complexity from the user:
  - Volume management
  - Pool creation
  - Thin storage pools
  - Snapshots
  - Automated read cache

Stratis offers powerful features, but currently lacks certain capabilities of other offerings that it might be compared to, such as Btrfs or ZFS. Most notably, it does not support CRCs with self healing.

# CHAPTER 60. MOUNTING NFS SHARES

As a system administrator, you can mount remote NFS shares on your system to access shared data.

## 60.1. INTRODUCTION TO NFS

This section explains the basic concepts of the NFS service.

A Network File System (NFS) allows remote hosts to mount file systems over a network and interact with those file systems as though they are mounted locally. This enables you to consolidate resources onto centralized servers on the network.

The NFS server refers to the **/etc/exports** configuration file to determine whether the client is allowed to access any exported file systems. Once verified, all file and directory operations are available to the user.

## 60.2. SUPPORTED NFS VERSIONS

This section lists versions of NFS supported in Red Hat Enterprise Linux and their features.

Currently, Red Hat Enterprise Linux 8 supports the following major versions of NFS:

- NFS version 3 (NFSv3) supports safe asynchronous writes and is more robust at error handling than the previous NFSv2; it also supports 64-bit file sizes and offsets, allowing clients to access more than 2 GB of file data.
- NFS version 4 (NFSv4) works through firewalls and on the Internet, no longer requires an **rpcbind** service, supports Access Control Lists (ACLs), and utilizes stateful operations.

NFS version 2 (NFSv2) is no longer supported by Red Hat.

### Default NFS version

The default NFS version in Red Hat Enterprise Linux 8 is 4.2. NFS clients attempt to mount using NFSv4.2 by default, and fall back to NFSv4.1 when the server does not support NFSv4.2. The mount later falls back to NFSv4.0 and then to NFSv3.

### Features of minor NFS versions

Following are the features of NFSv4.2 in Red Hat Enterprise Linux 8:

#### Server-side copy

Enables the NFS client to efficiently copy data without wasting network resources using the **copy\_file\_range()** system call.

#### Sparse files

Enables files to have one or more *holes*, which are unallocated or uninitialized data blocks consisting only of zeroes. The **lseek()** operation in NFSv4.2 supports **seek\_hole()** and **seek\_data()**, which enables applications to map out the location of holes in the sparse file.

#### Space reservation

Permits storage servers to reserve free space, which prohibits servers to run out of space. NFSv4.2 supports the **allocate()** operation to reserve space, the **deallocate()** operation to unreserve space, and the **fallocate()** operation to preallocate or deallocate space in a file.

#### Labeled NFS

Enforces data access rights and enables SELinux labels between a client and a server for individual files on an NFS file system.

## Layout enhancements

Provides the **layoutstats()** operation, which enables some Parallel NFS (pNFS) servers to collect better performance statistics.

Following are the features of NFSv4.1:

- Enhances performance and security of network, and also includes client-side support for pNFS.
- No longer requires a separate TCP connection for callbacks, which allows an NFS server to grant delegations even when it cannot contact the client: for example, when NAT or a firewall interferes.
- Provides exactly once semantics (except for reboot operations), preventing a previous issue whereby certain operations sometimes returned an inaccurate result if a reply was lost and the operation was sent twice.

## 60.3. SERVICES REQUIRED BY NFS

This section lists system services that are required for running an NFS server or mounting NFS shares. Red Hat Enterprise Linux starts these services automatically.

Red Hat Enterprise Linux uses a combination of kernel-level support and service processes to provide NFS file sharing. All NFS versions rely on Remote Procedure Calls (RPC) between clients and servers. To share or mount NFS file systems, the following services work together depending on which version of NFS is implemented:

### **nfsd**

The NFS server kernel module that services requests for shared NFS file systems.

### **rpcbind**

Accepts port reservations from local RPC services. These ports are then made available (or advertised) so the corresponding remote RPC services can access them. The **rpcbind** service responds to requests for RPC services and sets up connections to the requested RPC service. This is not used with NFSv4.

### **rpc.mountd**

This process is used by an NFS server to process **MOUNT** requests from NFSv3 clients. It checks that the requested NFS share is currently exported by the NFS server, and that the client is allowed to access it. If the mount request is allowed, the **nfs-mountd** service replies with a Success status and provides the File-Handle for this NFS share back to the NFS client.

### **rpc.nfsd**

This process enables explicit NFS versions and protocols the server advertises to be defined. It works with the Linux kernel to meet the dynamic demands of NFS clients, such as providing server threads each time an NFS client connects. This process corresponds to the **nfs-server** service.

### **lockd**

This is a kernel thread that runs on both clients and servers. It implements the Network Lock Manager (NLM) protocol, which enables NFSv3 clients to lock files on the server. It is started automatically whenever the NFS server is run and whenever an NFS file system is mounted.

### **rpc.statd**

This process implements the Network Status Monitor (NSM) RPC protocol, which notifies NFS clients when an NFS server is restarted without being gracefully brought down. The **rpc-statd** service is started automatically by the **nfs-server** service, and does not require user configuration. This is not used with NFSv4.

## rpc.rquotad

This process provides user quota information for remote users. The **rpc-rquotad** service is started automatically by the **nfs-server** service and does not require user configuration.

## rpc.idmapd

This process provides NFSv4 client and server upcalls, which map between on-the-wire NFSv4 names (strings in the form of **user@domain**) and local UIDs and GIDs. For **idmapd** to function with NFSv4, the **/etc/idmapd.conf** file must be configured. At a minimum, the **Domain** parameter should be specified, which defines the NFSv4 mapping domain. If the NFSv4 mapping domain is the same as the DNS domain name, this parameter can be skipped. The client and server must agree on the NFSv4 mapping domain for ID mapping to function properly.

Only the NFSv4 server uses **rpc.idmapd**, which is started by the **nfs-idmapd** service. The NFSv4 client uses the keyring-based **nfsidmap** utility, which is called by the kernel on-demand to perform ID mapping. If there is a problem with **nfsidmap**, the client falls back to using **rpc.idmapd**.

## The RPC services with NFSv4

The mounting and locking protocols have been incorporated into the NFSv4 protocol. The server also listens on the well-known TCP port 2049. As such, NFSv4 does not need to interact with **rpcbind**, **lockd**, and **rpc-statd** services. The **nfs-mountd** service is still required on the NFS server to set up the exports, but is not involved in any over-the-wire operations.

## Additional resources

- To configure an NFSv4-only server, which does not require **rpcbind**, see [Section 61.14, “Configuring an NFSv4-only server”](#).

## 60.4. NFS HOST NAME FORMATS

This section describes different formats that you can use to specify a host when mounting or exporting an NFS share.

You can specify the host in the following formats:

### Single machine

Either of the following:

- A fully-qualified domain name (that can be resolved by the server)
- Host name (that can be resolved by the server)
- An IP address.

### Series of machines specified with wildcards

You can use the **\*** or **?** characters to specify a string match.

Wildcards are not to be used with IP addresses; however, they might accidentally work if reverse DNS lookups fail. When specifying wildcards in fully qualified domain names, dots (.) are not included in the wildcard. For example, **\*.example.com** includes **one.example.com** but does not include **one.two.example.com**.

### IP networks

Either of the following formats is valid:

- ***a.b.c.d/z***, where ***a.b.c.d*** is the network and ***z*** is the number of bits in the netmask; for example **192.168.0.0/24**.
- ***a.b.c.d/netmask***, where ***a.b.c.d*** is the network and ***netmask*** is the netmask; for example, **192.168.100.8/255.255.255.0**.

### Netgroups

The **@group-name** format, where **group-name** is the NIS netgroup name.

## 60.5. INSTALLING NFS

This procedure installs all packages necessary to mount or export NFS shares.

### Procedure

- Install the **nfs-utils** package:

```
yum install nfs-utils
```

## 60.6. DISCOVERING NFS EXPORTS

This procedure discovers which file systems a given NFSv3 or NFSv4 server exports.

### Procedure

- With any server that supports NFSv3, use the **showmount** utility:

```
$ showmount --exports my-server
```

```
Export list for my-server
/exports/foo
/exports/bar
```

- With any server that supports NFSv4, mount the root directory and look around:

```
mount my-server:/ /mnt/
ls /mnt/
```

```
exports
```

```
ls /mnt/exports/
```

```
foo
```

```
bar
```

On servers that support both NFSv4 and NFSv3, both methods work and give the same results.

### Additional resources

- The **showmount(8)** man page.

## 60.7. MOUNTING AN NFS SHARE WITH MOUNT

This procedure mounts an NFS share exported from a server using the **mount** utility.

## Procedure

- To mount an NFS share, use the following command:

```
mount -t nfs -o options host:/remote/export /local/directory
```

This command uses the following variables:

### *options*

A comma-delimited list of mount options.

### *host*

The host name, IP address, or fully qualified domain name of the server exporting the file system you wish to mount.

### */remote/export*

The file system or directory being exported from the server, that is, the directory you wish to mount.

### */local/directory*

The client location where */remote/export* is mounted.

## Additional resources

- [Section 60.8, “Common NFS mount options”](#)
- [Section 60.4, “NFS host name formats”](#)
- [Section 66.3, “Mounting a file system with mount”](#)
- The **mount(8)** man page

## 60.8. COMMON NFS MOUNT OPTIONS

This section lists options commonly used when mounting NFS shares. These options can be used with manual mount commands, **/etc/fstab** settings, and **autofs**.

### Common NFS mount options

#### **lookupcache=mode**

Specifies how the kernel should manage its cache of directory entries for a given mount point. Valid arguments for *mode* are **all**, **none**, or **positive**.

#### **nfsvers=version**

Specifies which version of the NFS protocol to use, where *version* is **3**, **4**, **4.0**, **4.1**, or **4.2**. This is useful for hosts that run multiple NFS servers, or to disable retrying a mount with lower versions. If no version is specified, NFS uses the highest version supported by the kernel and the **mount** utility.

The option **vers** is identical to **nfsvers**, and is included in this release for compatibility reasons.

#### **noacl**

Turns off all ACL processing. This may be needed when interfacing with older versions of Red Hat Enterprise Linux, Red Hat Linux, or Solaris, because the most recent ACL technology is not compatible with older systems.

**nolock**

Disables file locking. This setting is sometimes required when connecting to very old NFS servers.

**noexec**

Prevents execution of binaries on mounted file systems. This is useful if the system is mounting a non-Linux file system containing incompatible binaries.

**nosuid**

Disables the **set-user-identifier** and **set-group-identifier** bits. This prevents remote users from gaining higher privileges by running a **setuid** program.

**port=num**

Specifies the numeric value of the NFS server port. If *num* is **0** (the default value), then **mount** queries the **rpcbind** service on the remote host for the port number to use. If the NFS service on the remote host is not registered with its **rpcbind** service, the standard NFS port number of TCP 2049 is used instead.

**rsize=num and wsize=num**

These options set the maximum number of bytes to be transferred in a single NFS read or write operation.

There is no fixed default value for **rsize** and **wsize**. By default, NFS uses the largest possible value that both the server and the client support. In Red Hat Enterprise Linux 8, the client and server maximum is 1,048,576 bytes. For more details, see the [What are the default and maximum values for rsize and wsize with NFS mounts?](#) KBase article.

**sec=mode**

Security flavors to use for accessing files on the mounted export.

The default setting is **sec=sys**, which uses local UNIX UIDs and GIDs. These use **AUTH\_SYS** to authenticate NFS operations.

Other options include:

- **sec=krb5** uses Kerberos V5 instead of local UNIX UIDs and GIDs to authenticate users.
- **sec=krb5i** uses Kerberos V5 for user authentication and performs integrity checking of NFS operations using secure checksums to prevent data tampering.
- **sec=krb5p** uses Kerberos V5 for user authentication, integrity checking, and encrypts NFS traffic to prevent traffic sniffing. This is the most secure setting, but it also involves the most performance overhead.

**tcp**

Instructs the NFS mount to use the TCP protocol.

**Additional resources**

- The **mount(8)** man page
- The **nfs(5)** man page

## 60.9. RELATED INFORMATION

- The Linux NFS wiki: <https://linux-nfs.org>
- To mount NFS shares persistently, see [Section 66.8, “Persistently mounting file systems”](#).

- To mount NFS shares on demand, see [Section 66.9, “Mounting file systems on demand”](#).

# CHAPTER 61. EXPORTING NFS SHARES

As a system administrator, you can use the NFS server to share a directory on your system over network.

## 61.1. INTRODUCTION TO NFS

This section explains the basic concepts of the NFS service.

A Network File System (NFS) allows remote hosts to mount file systems over a network and interact with those file systems as though they are mounted locally. This enables you to consolidate resources onto centralized servers on the network.

The NFS server refers to the **/etc(exports** configuration file to determine whether the client is allowed to access any exported file systems. Once verified, all file and directory operations are available to the user.

## 61.2. SUPPORTED NFS VERSIONS

This section lists versions of NFS supported in Red Hat Enterprise Linux and their features.

Currently, Red Hat Enterprise Linux 8 supports the following major versions of NFS:

- NFS version 3 (NFSv3) supports safe asynchronous writes and is more robust at error handling than the previous NFSv2; it also supports 64-bit file sizes and offsets, allowing clients to access more than 2 GB of file data.
- NFS version 4 (NFSv4) works through firewalls and on the Internet, no longer requires an **rpcbind** service, supports Access Control Lists (ACLs), and utilizes stateful operations.

NFS version 2 (NFSv2) is no longer supported by Red Hat.

### Default NFS version

The default NFS version in Red Hat Enterprise Linux 8 is 4.2. NFS clients attempt to mount using NFSv4.2 by default, and fall back to NFSv4.1 when the server does not support NFSv4.2. The mount later falls back to NFSv4.0 and then to NFSv3.

### Features of minor NFS versions

Following are the features of NFSv4.2 in Red Hat Enterprise Linux 8:

#### Server-side copy

Enables the NFS client to efficiently copy data without wasting network resources using the **copy\_file\_range()** system call.

#### Sparse files

Enables files to have one or more *holes*, which are unallocated or uninitialized data blocks consisting only of zeroes. The **lseek()** operation in NFSv4.2 supports **seek\_hole()** and **seek\_data()**, which enables applications to map out the location of holes in the sparse file.

#### Space reservation

Permits storage servers to reserve free space, which prohibits servers to run out of space. NFSv4.2 supports the **allocate()** operation to reserve space, the **deallocate()** operation to unreserve space, and the **fallocate()** operation to preallocate or deallocate space in a file.

#### Labeled NFS

Enforces data access rights and enables SELinux labels between a client and a server for individual files on an NFS file system.

## Layout enhancements

Provides the **layoutstats()** operation, which enables some Parallel NFS (pNFS) servers to collect better performance statistics.

Following are the features of NFSv4.1:

- Enhances performance and security of network, and also includes client-side support for pNFS.
- No longer requires a separate TCP connection for callbacks, which allows an NFS server to grant delegations even when it cannot contact the client: for example, when NAT or a firewall interferes.
- Provides exactly once semantics (except for reboot operations), preventing a previous issue whereby certain operations sometimes returned an inaccurate result if a reply was lost and the operation was sent twice.

## 61.3. THE TCP AND UDP PROTOCOLS IN NFSV3 AND NFSV4

NFSv4 requires the Transmission Control Protocol (TCP) running over an IP network.

NFSv3 could also use the User Datagram Protocol (UDP) in earlier Red Hat Enterprise Linux versions. In Red Hat Enterprise Linux 8, NFS over UDP is no longer supported. By default, UDP is disabled in the NFS server.

## 61.4. SERVICES REQUIRED BY NFS

This section lists system services that are required for running an NFS server or mounting NFS shares. Red Hat Enterprise Linux starts these services automatically.

Red Hat Enterprise Linux uses a combination of kernel-level support and service processes to provide NFS file sharing. All NFS versions rely on Remote Procedure Calls (RPC) between clients and servers. To share or mount NFS file systems, the following services work together depending on which version of NFS is implemented:

### **nfsd**

The NFS server kernel module that services requests for shared NFS file systems.

### **rpcbind**

Accepts port reservations from local RPC services. These ports are then made available (or advertised) so the corresponding remote RPC services can access them. The **rpcbind** service responds to requests for RPC services and sets up connections to the requested RPC service. This is not used with NFSv4.

### **rpc.mountd**

This process is used by an NFS server to process **MOUNT** requests from NFSv3 clients. It checks that the requested NFS share is currently exported by the NFS server, and that the client is allowed to access it. If the mount request is allowed, the **nfs-mountd** service replies with a Success status and provides the File-Handle for this NFS share back to the NFS client.

### **rpc.nfsd**

This process enables explicit NFS versions and protocols the server advertises to be defined. It works with the Linux kernel to meet the dynamic demands of NFS clients, such as providing server threads each time an NFS client connects. This process corresponds to the **nfs-server** service.

### **lockd**

This is a kernel thread that runs on both clients and servers. It implements the Network Lock Manager (NLM) protocol, which enables NFSv3 clients to lock files on the server. It is started automatically whenever the NFS server is run and whenever an NFS file system is mounted.

### rpc.statd

This process implements the Network Status Monitor (NSM) RPC protocol, which notifies NFS clients when an NFS server is restarted without being gracefully brought down. The **rpc-statd** service is started automatically by the **nfs-server** service, and does not require user configuration. This is not used with NFSv4.

### rpc.rquotad

This process provides user quota information for remote users. The **rpc-rquotad** service is started automatically by the **nfs-server** service and does not require user configuration.

### rpc.idmapd

This process provides NFSv4 client and server upcalls, which map between on-the-wire NFSv4 names (strings in the form of **user@domain**) and local UIDs and GIDs. For **idmapd** to function with NFSv4, the **/etc/idmapd.conf** file must be configured. At a minimum, the **Domain** parameter should be specified, which defines the NFSv4 mapping domain. If the NFSv4 mapping domain is the same as the DNS domain name, this parameter can be skipped. The client and server must agree on the NFSv4 mapping domain for ID mapping to function properly.

Only the NFSv4 server uses **rpc.idmapd**, which is started by the **nfs-idmapd** service. The NFSv4 client uses the keyring-based **nfsidmap** utility, which is called by the kernel on-demand to perform ID mapping. If there is a problem with **nfsidmap**, the client falls back to using **rpc.idmapd**.

## The RPC services with NFSv4

The mounting and locking protocols have been incorporated into the NFSv4 protocol. The server also listens on the well-known TCP port 2049. As such, NFSv4 does not need to interact with **rpcbind**, **lockd**, and **rpc-statd** services. The **nfs-mountd** service is still required on the NFS server to set up the exports, but is not involved in any over-the-wire operations.

## Additional resources

- To configure an NFSv4-only server, which does not require **rpcbind**, see [Section 61.14, “Configuring an NFSv4-only server”](#).

## 61.5. NFS HOST NAME FORMATS

This section describes different formats that you can use to specify a host when mounting or exporting an NFS share.

You can specify the host in the following formats:

### Single machine

Either of the following:

- A fully-qualified domain name (that can be resolved by the server)
- Host name (that can be resolved by the server)
- An IP address.

### Series of machines specified with wildcards

You can use the **\*** or **?** characters to specify a string match.

Wildcards are not to be used with IP addresses; however, they might accidentally work if reverse DNS

lookups fail. When specifying wildcards in fully qualified domain names, dots (.) are not included in the wildcard. For example, **\*.example.com** includes **one.example.com** but does not include **one.two.example.com**.

## IP networks

Either of the following formats is valid:

- **a.b.c.d/z**, where **a.b.c.d** is the network and **z** is the number of bits in the netmask; for example **192.168.0.0/24**.
- **a.b.c.d/netmask**, where **a.b.c.d** is the network and **netmask** is the netmask; for example, **192.168.100.8/255.255.255.0**.

## Netgroups

The **@group-name** format, where **group-name** is the NIS netgroup name.

## 61.6. NFS SERVER CONFIGURATION

This section describes the syntax and options of two ways to configure exports on an NFS server:

- Manually editing the **/etc/exports** configuration file
- Using the **exportfs** utility on the command line

### 61.6.1. The **/etc/exports** configuration file

The **/etc/exports** file controls which file systems are exported to remote hosts and specifies options. It follows the following syntax rules:

- Blank lines are ignored.
- To add a comment, start a line with the hash mark (#).
- You can wrap long lines with a backslash (\).
- Each exported file system should be on its own individual line.
- Any lists of authorized hosts placed after an exported file system must be separated by space characters.
- Options for each of the hosts must be placed in parentheses directly after the host identifier, without any spaces separating the host and the first parenthesis.

#### Export entry

Each entry for an exported file system has the following structure:

**export host(options)**

It is also possible to specify multiple hosts, along with specific options for each host. To do so, list them on the same line as a space-delimited list, with each host name followed by its respective options (in parentheses), as in:

**export host1(options1) host2(options2) host3(options3)**

In this structure:

### **export**

The directory being exported

### **host**

The host or network to which the export is being shared

### **options**

The options to be used for host

#### **Example 61.1. A simple /etc/exports file**

In its simplest form, the **/etc/exports** file only specifies the exported directory and the hosts permitted to access it:

/exported/directory bob.example.com

Here, **bob.example.com** can mount **/exported/directory/** from the NFS server. Because no options are specified in this example, NFS uses default options.



### **IMPORTANT**

The format of the **/etc/exports** file is very precise, particularly in regards to use of the space character. Remember to always separate exported file systems from hosts and hosts from one another with a space character. However, there should be no other space characters in the file except on comment lines.

For example, the following two lines do not mean the same thing:

/home bob.example.com(rw)  
 /home bob.example.com (rw)

The first line allows only users from **bob.example.com** read and write access to the **/home** directory. The second line allows users from **bob.example.com** to mount the directory as read-only (the default), while the rest of the world can mount it read/write.

### **Default options**

The default options for an export entry are:

#### **ro**

The exported file system is read-only. Remote hosts cannot change the data shared on the file system. To allow hosts to make changes to the file system (that is, read and write), specify the **rw** option.

#### **sync**

The NFS server will not reply to requests before changes made by previous requests are written to disk. To enable asynchronous writes instead, specify the option **async**.

#### **wdelay**

The NFS server will delay writing to the disk if it suspects another write request is imminent. This can improve performance as it reduces the number of times the disk must be accessed by separate write commands, thereby reducing write overhead. To disable this, specify the **no\_wdelay** option, which is available only if the default **sync** option is also specified.

## root\_squash

This prevents root users connected remotely (as opposed to locally) from having root privileges; instead, the NFS server assigns them the user ID **nfsnobody**. This effectively "squashes" the power of the remote root user to the lowest local user, preventing possible unauthorized writes on the remote server. To disable root squashing, specify the **no\_root\_squash** option.

To squash every remote user (including root), use the **all\_squash** option. To specify the user and group IDs that the NFS server should assign to remote users from a particular host, use the **anonuid** and **anongid** options, respectively, as in:

```
export host(anonuid=uid,anongid=gid)
```

Here, *uid* and *gid* are user ID number and group ID number, respectively. The **anonuid** and **anongid** options enable you to create a special user and group account for remote NFS users to share.

By default, access control lists (ACLs) are supported by NFS under Red Hat Enterprise Linux. To disable this feature, specify the **no\_acl** option when exporting the file system.

### Default and overridden options

Each default for every exported file system must be explicitly overridden. For example, if the **rw** option is not specified, then the exported file system is shared as read-only. The following is a sample line from **/etc(exports** which overrides two default options:

```
/another/exported/directory 192.168.0.3(rw,async)
```

In this example, **192.168.0.3** can mount **/another/exported/directory** read and write, and all writes to disk are asynchronous.

## 61.6.2. The exportfs utility

The **exportfs** utility enables the root user to selectively export or unexport directories without restarting the NFS service. When given the proper options, the **exportfs** utility writes the exported file systems to **/var/lib/nfs/xtab**. Because the **nfs-mountd** service refers to the **xtab** file when deciding access privileges to a file system, changes to the list of exported file systems take effect immediately.

### Common exportfs options

The following is a list of commonly-used options available for **exportfs**:

**-r**

Causes all directories listed in **/etc(exports** to be exported by constructing a new export list in **/etc/lib/nfs/xtab**. This option effectively refreshes the export list with any changes made to **/etc(exports**.

**-a**

Causes all directories to be exported or unexported, depending on what other options are passed to **exportfs**. If no other options are specified, **exportfs** exports all file systems specified in **/etc(exports**.

### **-o file-systems**

Specifies directories to be exported that are not listed in **/etc(exports**. Replace *file-systems* with additional file systems to be exported. These file systems must be formatted in the same way they are specified in **/etc(exports**. This option is often used to test an exported file system before adding it permanently to the list of exported file systems.

**-i**

Ignores **/etc(exports**; only options given from the command line are used to define exported file systems.

**-u**

Unexports all shared directories. The command **exportfs -ua** suspends NFS file sharing while keeping all NFS services up. To re-enable NFS sharing, use **exportfs -r**.

**-v**

Verbose operation, where the file systems being exported or unexported are displayed in greater detail when the **exportfs** command is executed.

If no options are passed to the **exportfs** utility, it displays a list of currently exported file systems.

#### Additional resources

- For information on different methods for specifying host names, see [Section 61.5, “NFS host name formats”](#).
- For a complete list of export options, see the **exports(5)** man page.
- For more information about the **exportfs** utility, see the **exportfs(8)** man page.

## 61.7. NFS AND RPCBIND

This section explains the purpose of the **rpcbind** service, which is required by NFSv3.

The **rpcbind** service maps Remote Procedure Call (RPC) services to the ports on which they listen. RPC processes notify **rpcbind** when they start, registering the ports they are listening on and the RPC program numbers they expect to serve. The client system then contacts **rpcbind** on the server with a particular RPC program number. The **rpcbind** service redirects the client to the proper port number so it can communicate with the requested service.

Because RPC-based services rely on **rpcbind** to make all connections with incoming client requests, **rpcbind** must be available before any of these services start.

Access control rules for **rpcbind** affect all RPC-based services. Alternatively, it is possible to specify access control rules for each of the NFS RPC daemons.

#### Additional resources

- For the precise syntax of access control rules, see the **rpc.mountd(8)** and **rpc.statd(8)** man pages.

## 61.8. INSTALLING NFS

This procedure installs all packages necessary to mount or export NFS shares.

#### Procedure

- Install the **nfs-utils** package:

```
yum install nfs-utils
```

## 61.9. STARTING THE NFS SERVER

This procedure describes how to start the NFS server, which is required to export NFS shares.

### Prerequisites

- For servers that support NFSv2 or NFSv3 connections, the **rpcbind** service must be running. To verify that **rpcbind** is active, use the following command:

```
$ systemctl status rpcbind
```

If the service is stopped, start and enable it:

```
$ systemctl enable --now rpcbind
```

### Procedure

- To start the NFS server and enable it to start automatically at boot, use the following command:

```
systemctl enable --now nfs-server
```

### Additional resources

- To configure an NFSv4-only server, which does not require **rpcbind**, see [Section 61.14, “Configuring an NFSv4-only server”](#).

## 61.10. TROUBLESHOOTING NFS AND RPCBIND

Because the **rpcbind** service provides coordination between RPC services and the port numbers used to communicate with them, it is useful to view the status of current RPC services using **rpcbind** when troubleshooting. The **rpcinfo** utility shows each RPC-based service with port numbers, an RPC program number, a version number, and an IP protocol type (TCP or UDP).

### Procedure

- To make sure the proper NFS RPC-based services are enabled for **rpcbind**, use the following command:

```
rpcinfo -p
```

#### Example 61.2. **rpcinfo -p** command output

The following is sample output from this command:

```
program vers proto port service
100000 4 tcp 111 portmapper
100000 3 tcp 111 portmapper
100000 2 tcp 111 portmapper
100000 4 udp 111 portmapper
100000 3 udp 111 portmapper
100000 2 udp 111 portmapper
100005 1 udp 20048 mountd
100005 1 tcp 20048 mountd
100005 2 udp 20048 mountd
100005 2 tcp 20048 mountd
```

```

100005 3 udp 20048 mountd
100005 3 tcp 20048 mountd
100024 1 udp 37769 status
100024 1 tcp 49349 status
100003 3 tcp 2049 nfs
100003 4 tcp 2049 nfs
100227 3 tcp 2049 nfs_acl
100021 1 udp 56691 nlockmgr
100021 3 udp 56691 nlockmgr
100021 4 udp 56691 nlockmgr
100021 1 tcp 46193 nlockmgr
100021 3 tcp 46193 nlockmgr
100021 4 tcp 46193 nlockmgr

```

If one of the NFS services does not start up correctly, **rpcbind** will be unable to map RPC requests from clients for that service to the correct port.

2. In many cases, if NFS is not present in **rpcinfo** output, restarting NFS causes the service to correctly register with **rpcbind** and begin working:

```
systemctl restart nfs-server
```

#### Additional resources

- For more information and a list of **rpcinfo** options, see the **rpcinfo(8)** man page.
- To configure an NFSv4-only server, which does not require **rpcbind**, see [Section 61.14, "Configuring an NFSv4-only server"](#).

## 61.11. CONFIGURING THE NFS SERVER TO RUN BEHIND A FIREWALL

NFS requires the **rpcbind** service, which dynamically assigns ports for RPC services and can cause issues for configuring firewall rules. This procedure describes how to configure the NFS server to work behind a firewall.

#### Procedure

1. To allow clients to access NFS shares behind a firewall, set which ports the RPC services run on in the **[mountd]** section of the **/etc/nfs.conf** file:

```

[mountd]
port=port-number

```

This adds the **-p port-number** option to the **rpc.mount** command line: **rpc.mount -p port-number**.

2. To allow clients to access NFS shares behind a firewall, configure the firewall by running the following commands on the NFS server:

```

firewall-cmd --permanent --add-service mountd
firewall-cmd --permanent --add-service rpc-bind
firewall-cmd --permanent --add-service nfs

```

```
firewall-cmd --permanent --add-port=<mountd-port>/tcp
firewall-cmd --permanent --add-port=<mountd-port>/udp
firewall-cmd --reload
```

In the commands, replace `<mountd-port>` with the intended port or a port range. When specifying a port range, use the `--add-port=<mountd-port>-<mountd-port>/udp` syntax.

3. To allow NFSv4.0 callbacks to pass through firewalls, set `/proc/sys/fs/nfs/nfs_callback_tcpport` and allow the server to connect to that port on the client.

This step is not needed for NFSv4.1 or higher, and the other ports for `mountd`, `statd`, and `lockd` are not required in a pure NFSv4 environment.

4. To specify the ports to be used by the RPC service `nlockmgr`, set the port number for the `nlm_tcpport` and `nlm_udpport` options in the `/etc/modprobe.d/lockd.conf` file.
5. Restart the NFS server:

```
systemctl restart nfs-server
```

If NFS fails to start, check `/var/log/messages`. Commonly, NFS fails to start if you specify a port number that is already in use.

6. Confirm the changes have taken effect:

```
rpcinfo -p
```

#### Additional resources

- To configure an NFSv4-only server, which does not require `rpcbind`, see [Section 61.14, “Configuring an NFSv4-only server”](#).

## 61.12. EXPORTING RPC QUOTA THROUGH A FIREWALL

If you export a file system that uses disk quotas, you can use the quota Remote Procedure Call (RPC) service to provide disk quota data to NFS clients.

#### Procedure

1. Enable and start the `rpc-rquotad` service:

```
systemctl enable --now rpc-rquotad
```



#### NOTE

The `rpc-rquotad` service is, if enabled, started automatically after starting the `nfs-server` service.

2. To make the quota RPC service accessible behind a firewall, the TCP (or UDP, if UDP is enabled) port 875 need to be open. The default port number is defined in the `/etc/services` file. You can override the default port number by appending `-p port-number` to the `RPCRQUOTADOPTS` variable in the `/etc/sysconfig/rpc-rquotad` file.

3. By default, remote hosts can only read quotas. If you want to allow clients to set quotas, append the **-S** option to the **RPCRQUOTADOPTS** variable in the **/etc/sysconfig/rpc-rquotad** file.
4. Restart **rpc-rquotad** for the changes in the **/etc/sysconfig/rpc-rquotad** file to take effect:

```
systemctl restart rpc-rquotad
```

## 61.13. ENABLING NFS OVER RDMA (NFSORDMA)

The remote direct memory access (RDMA) service works automatically in Red Hat Enterprise Linux 8 if there is RDMA-capable hardware present.

### Procedure

1. Install the **rdma-core** package:

```
yum install rdma-core
```

2. To enable automatic loading of NFSoRDMA server modules, add the **SVCRDMA\_LOAD=yes** option on a new line in the **/etc/rdma/rdma.conf** configuration file.

The **rdma=20049** option in the **[nfsd]** section of the **/etc/nfs.conf** file specifies the port number on which the NFSoRDMA service listens for clients. The RFC 5667 standard specifies that servers must listen on port **20049** when providing NFSv4 services over RDMA.

The **/etc/rdma/rdma.conf** file contains a line that sets the **XPRTRDMA\_LOAD=yes** option by default, which requests the **rdma** service to load the NFSoRDMA *client* module.

3. Restart the **nfs-server** service:

```
systemctl restart nfs-server
```

### Additional resources

- The RFC 5667 standard: <https://tools.ietf.org/html/rfc5667>.

## 61.14. CONFIGURING AN NFSV4-ONLY SERVER

As an NFS server administrator, you can configure the NFS server to support only NFSv4, which minimizes the number of open ports and running services on the system.

### 61.14.1. Benefits and drawbacks of an NFSv4-only server

This section explains the benefits and drawbacks of configuring the NFS server to only support NFSv4.

By default, the NFS server supports NFSv2, NFSv3, and NFSv4 connections in Red Hat Enterprise Linux 8. However, you can also configure NFS to support only NFS version 4.0 and later. This minimizes the number of open ports and running services on the system, because NFSv4 does not require the **rpcbind** service to listen on the network.

When your NFS server is configured as NFSv4-only, clients attempting to mount shares using NFSv2 or NFSv3 fail with an error like the following:

```
Requested NFS version or transport protocol is not supported.
```

Optionally, you can also disable listening for the **RPCBIND**, **MOUNT**, and **NSM** protocol calls, which are not necessary in the NFSv4-only case.

The effects of disabling these additional options are:

- Clients that attempt to mount shares from your server using NFSv2 or NFSv3 become unresponsive.
- The NFS server itself is unable to mount NFSv2 and NFSv3 file systems.

### 61.14.2. NFS and **rpcbind**

This section explains the purpose of the **rpcbind** service, which is required by NFSv3.

The **rpcbind** service maps Remote Procedure Call (RPC) services to the ports on which they listen. RPC processes notify **rpcbind** when they start, registering the ports they are listening on and the RPC program numbers they expect to serve. The client system then contacts **rpcbind** on the server with a particular RPC program number. The **rpcbind** service redirects the client to the proper port number so it can communicate with the requested service.

Because RPC-based services rely on **rpcbind** to make all connections with incoming client requests, **rpcbind** must be available before any of these services start.

Access control rules for **rpcbind** affect all RPC-based services. Alternatively, it is possible to specify access control rules for each of the NFS RPC daemons.

#### Additional resources

- For the precise syntax of access control rules, see the **rpc.mountd(8)** and **rpc.statd(8)** man pages.

### 61.14.3. Configuring the NFS server to support only NFSv4

This procedure describes how to configure your NFS server to support only NFS version 4.0 and later.

#### Procedure

1. Disable NFSv2 and NFSv3 by adding the following lines to the **[nfsd]** section of the **/etc/nfs.conf** configuration file:

```
[nfsd]
vers2=no
vers3=no
```

2. Optionally, disable listening for the **RPCBIND**, **MOUNT**, and **NSM** protocol calls, which are not necessary in the NFSv4-only case. Disable related services:

```
systemctl mask --now rpc-statd.service rpcbind.service rpcbind.socket
```

3. Restart the NFS server:

```
systemctl restart nfs-server
```

The changes take effect as soon as you start or restart the NFS server.

#### 61.14.4. Verifying the NFSv4-only configuration

This procedure describes how to verify that your NFS server is configured in the NFSv4-only mode by using the **netstat** utility.

##### Procedure

- Use the **netstat** utility to list services listening on the TCP and UDP protocols:

```
netstat --listening --tcp --udp
```

#### Example 61.3. Output on an NFSv4-only server

The following is an example **netstat** output on an NFSv4-only server; listening for **RPCBIND**, **MOUNT**, and **NSM** is also disabled. Here, **nfs** is the only listening NFS service:

```
netstat --listening --tcp --udp
```

Active Internet connections (only servers)					
Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	0.0.0.0:ssh	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:nfs	0.0.0.0:*	LISTEN
tcp6	0	0	[:]ssh	[:]/*	LISTEN
tcp6	0	0	[:]nfs	[:]/*	LISTEN
udp	0	0	localhost.locald:bootpc	0.0.0.0:*	

#### Example 61.4. Output before configuring an NFSv4-only server

In comparison, the **netstat** output before configuring an NFSv4-only server includes the **sunrpc** and **mountd** services:

```
netstat --listening --tcp --udp
```

Active Internet connections (only servers)					
Proto	Recv-Q	Send-Q	Local Address	Foreign Address	State
tcp	0	0	0.0.0.0:ssh	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:40189	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:46813	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:nfs	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:sunrpc	0.0.0.0:*	LISTEN
tcp	0	0	0.0.0.0:mountd	0.0.0.0:*	LISTEN
tcp6	0	0	[:]ssh	[:]/*	LISTEN
tcp6	0	0	[:]51227	[:]/*	LISTEN
tcp6	0	0	[:]nfs	[:]/*	LISTEN
tcp6	0	0	[:]sunrpc	[:]/*	LISTEN
tcp6	0	0	[:]mountd	[:]/*	LISTEN
tcp6	0	0	[:]45043	[:]/*	LISTEN
udp	0	0	localhost:1018	0.0.0.0:*	
udp	0	0	localhost.locald:bootpc	0.0.0.0:*	
udp	0	0	0.0.0.0:mountd	0.0.0.0:*	
udp	0	0	0.0.0.0:46672	0.0.0.0:*	

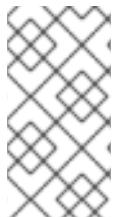
```
 udp 0 0 0.0.0.0:sunrpc 0.0.0.0:*
 udp 0 0 0.0.0.0:33494 0.0.0.0:*
 udp6 0 0 [::]:33734 [::]:*
 udp6 0 0 [::]:mountd [::]:*
 udp6 0 0 [::]:sunrpc [::]:*
 udp6 0 0 [::]:40243 [::]:*
```

## 61.15. RELATED INFORMATION

- The Linux NFS wiki: <https://linux-nfs.org>

# CHAPTER 62. MOUNTING AN SMB SHARE ON RED HAT ENTERPRISE LINUX

The Server Message Block (SMB) protocol implements an application-layer network protocol used to access resources on a server, such as file shares and shared printers.



## NOTE

In the context of SMB, you can find mentions about the Common Internet File System (CIFS) protocol, which is a dialect of SMB. Both the SMB and CIFS protocol are supported, and the kernel module and utilities involved in mounting SMB and CIFS shares both use the name **cifs**.

This section describes how to mount shares from an SMB server. For details about setting up an SMB server on Red Hat Enterprise Linux using Samba, see [Using Samba as a server](#).

## Prerequisites

On Microsoft Windows, SMB is implemented by default. On Red Hat Enterprise Linux, the **cifs.ko** file system module of the kernel provides support for mounting SMB shares. Therefor install the **cifs-utils** package:

```
yum install cifs-utils
```

The **cifs-utils** package provides utilities to:

- Mount SMB and CIFS shares
- Manage NT Lan Manager (NTLM) credentials in the kernel's keyring
- Set and display Access Control Lists (ACL) in a security descriptor on SMB and CIFS shares

## 62.1. SUPPORTED SMB PROTOCOL VERSIONS

The **cifs.ko** kernel module supports the following SMB protocol versions:

- SMB 1
- SMB 2.0
- SMB 2.1
- SMB 3.0
- SMB 3.1.1



## NOTE

Depending on the protocol version, not all SMB features are implemented.

## 62.2. UNIX EXTENSIONS SUPPORT

Samba uses the **CAP\_UNIX** capability bit in the SMB protocol to provide the UNIX extensions feature. These extensions are also supported by the **cifs.ko** kernel module. However, both Samba and the kernel module support UNIX extensions only in the SMB 1 protocol.

To use UNIX extensions:

1. Set the **server min protocol** parameter in the **[global]** section in the **/etc/samba/smb.conf** file to **NT1**.
2. Mount the share using the SMB 1 protocol by providing the **-o vers=1.0** option to the **mount** command. For example:

```
mount -t cifs -o vers=1.0,username=user_name //server_name/share_name /mnt/
```

By default, the kernel module uses SMB 2 or the highest later protocol version supported by the server. Passing the **-o vers=1.0** option to the **mount** command forces that the kernel module uses the SMB 1 protocol that is required for using UNIX extensions.

To verify if UNIX extensions are enabled, display the options of the mounted share:

```
mount
...
//server/share on /mnt type cifs (...,unix,...)
```

If the **unix** entry is displayed in the list of mount options, UNIX extensions are enabled.

## 62.3. MANUALLY MOUNTING AN SMB SHARE

If you only require an SMB share to be temporary mounted, you can mount it manually using the **mount** utility.



### NOTE

Manually mounted shares are not mounted automatically again when you reboot the system. To configure that Red Hat Enterprise Linux automatically mounts the share when the system boots, see [Section 62.4, “Mounting an SMB share automatically when the system boots”](#).

### Prerequisites

- The **cifs-utils** package is installed.

### Procedure

To manually mount an SMB share, use the **mount** utility with the **-t cifs** parameter:

```
mount -t cifs -o username=user_name //server_name/share_name /mnt/
Password for user_name@//server_name/share_name: password
```

In the **-o** parameter, you can specify options that are used to mount the share. For details, see [Section 62.7, “Frequently used mount options”](#) and the **OPTIONS** section in the **mount.cifs(8)** man page.

**Example 62.1. Mounting a share using an encrypted SMB 3.0 connection**

To mount the `\server\example` share as the **DOMA\administrator** user over an encrypted SMB 3.0 connection into the `/mnt` directory:

```
mount -t cifs -o username=DOMA\administrator,seal,vers=3.0 //server/example /mnt/
Password for DOMA\administrator@//server_name/share_name: password
```

## 62.4. MOUNTING AN SMB SHARE AUTOMATICALLY WHEN THE SYSTEM BOOTS

If access to a mounted SMB share is permanently required on a server, mount the share automatically at boot time.

### Prerequisites

- The **cifs-utils** package is installed.

### Procedure

To mount an SMB share automatically when the system boots, add an entry for the share to the **/etc/fstab** file. For example:

```
//server_name/share_name /mnt cifs credentials=/root/smb.cred 0 0
```



### IMPORTANT

To enable the system to mount a share automatically, you must store the user name, password, and domain name in a credentials file. For details, see [Section 62.5, “Authenticating to an SMB share using a credentials file”](#).

In the fourth field of the row in the **/etc/fstab**, specify mount options, such as the path to the credentials file. For details, see [Section 62.7, “Frequently used mount options”](#) and the **OPTIONS** section in the **mount.cifs(8)** man page.

To verify that the share mounts successfully, enter:

```
mount /mnt/
```

## 62.5. AUTHENTICATING TO AN SMB SHARE USING A CREDENTIALS FILE

In certain situations, such as when mounting a share automatically at boot time, a share should be mounted without entering the user name and password. To implement this, create a credentials file.

### Prerequisites

- The **cifs-utils** package is installed.

### Procedure

1. Create a file, such as **/root/smb.cred**, and specify the user name, password, and domain name that file:

```
username=user_name
password=password
domain=domain_name
```

2. Set the permissions to only allow the owner to access the file:

```
chown user_name /root/smb.cred
chmod 600 /root/smb.cred
```

You can now pass the **credentials=file\_name** mount option to the **mount** utility or use it in the **/etc/fstab** file to mount the share without being prompted for the user name and password.

## 62.6. PERFORMING A MULTI-USER SMB MOUNT

The credentials you provide to mount a share determine the access permissions on the mount point by default. For example, if you use the **DOMAIN\example** user when you mount a share, all operations on the share will be executed as this user, regardless which local user performs the operation.

However, in certain situations, the administrator wants to mount a share automatically when the system boots, but users should perform actions on the share's content using their own credentials. The **multiuser** mount options lets you configure this scenario.



### IMPORTANT

To use the **multiuser** mount option, you must additionally set the **sec** mount option to a security type that supports providing credentials in a non-interactive way, such as **krb5** or the **ntlmssp** option with a credentials file. For details, see [Section 62.6.3, "Accessing a share as a user"](#).

The **root** user mounts the share using the **multiuser** option and an account that has minimal access to the contents of the share. Regular users can then provide their user name and password to the current session's kernel keyring using the **cifscreds** utility. If the user accesses the content of the mounted share, the kernel uses the credentials from the kernel keyring instead of the one initially used to mount the share.

Using this feature consists of the following steps:

- Mount a share with the **multiuser** option.
- Optionally, verify if the share was successfully mounted with the **multiuser** option.
- Access the share as a user .

### Prerequisites

- The **cifs-utils** package is installed.

#### 62.6.1. Mounting a share with the multiuser option

Before users can access the share with their own credentials, mount the share as the **root** user using an account with limited permissions.

## Procedure

To mount a share automatically with the **multiuser** option when the system boots:

1. Create the entry for the share in the **/etc/fstab** file. For example:

```
//server_name/share_name /mnt cifs multiuser,sec=ntlmssp,credentials=/root/smb.cred
0 0
```

2. Mount the share:

```
mount /mnt/
```

If you do not want to mount the share automatically when the system boots, mount it manually by passing **-o multiuser,sec=security\_type** to the **mount** command. For details about mounting an SMB share manually, see [Section 62.3, "Manually mounting an SMB share"](#).

### 62.6.2. Verifying if an SMB share is mounted with the multiuser option

To verify if a share is mounted with the **multiuser** option, display the mount options.

## Procedure

```
mount
...
//server_name/share_name on /mnt type cifs (sec=ntlmssp,multiuser,...)
```

If the **multiuser** entry is displayed in the list of mount options, the feature is enabled.

### 62.6.3. Accessing a share as a user

If an SMB share is mounted with the **multiuser** option, users can provide their credentials for the server to the kernel's keyring:

```
cifscreds add -u SMB_user_name server_name
Password: password
```

When the user performs operations in the directory that contains the mounted SMB share, the server applies the file system permissions for this user, instead of the one initially used when the share was mounted.



#### NOTE

Multiple users can perform operations using their own credentials on the mounted share at the same time.

## 62.7. FREQUENTLY USED MOUNT OPTIONS

When you mount an SMB share, the mount options determine:

- How the connection will be established with the server. For example, which SMB protocol version is used when connecting to the server.

- How the share will be mounted into the local file system. For example, if the system overrides the remote file and directory permissions to enable multiple local users to access the content on the server.

To set multiple options in the fourth field of the `/etc/fstab` file or in the `-o` parameter of a mount command, separate them with commas. For example, see [Section 62.6.1, “Mounting a share with the multiuser option”](#).

The following list gives frequently used mount options:

Option	Description
<code>credentials=file_name</code>	Sets the path to the credentials file. See <a href="#">Section 62.5, “Authenticating to an SMB share using a credentials file”</a>
<code>dir_mode=mode</code>	Sets the directory mode if the server does not support CIFS UNIX extensions.
<code>file_mode=mode</code>	Sets the file mode if the server does not support CIFS UNIX extensions.
<code>password=password</code>	Sets the password used to authenticate to the SMB server. Alternatively, specify a credentials file using the <b>credentials</b> option.
<code>seal</code>	Enables encryption support for connections using SMB 3.0 or a later protocol version. Therefore, use <b>seal</b> together with the <b>vers</b> mount option set to <b>3.0</b> or later. See <a href="#">Example 62.1, “Mounting a share using an encrypted SMB 3.0 connection”</a> .
<code>sec=security_mode</code>	<p>Sets the security mode, such as <b>ntlmssp</b>, to enable NTLMv2 password hashing and enabled packet signing. For a list of supported values, see the option’s description in the <b>mount.cifs(8)</b> man page.</p> <p>If the server does not support the <b>ntlmv2</b> security mode, use <b>sec=ntlmssp</b>, which is the default.</p> <p>For security reasons, do not use the insecure <b>ntlm</b> security mode.</p>
<code>username=user_name</code>	Sets the user name used to authenticate to the SMB server. Alternatively, specify a credentials file using the <b>credentials</b> option.
<code>vers=SMB_protocol_version</code>	Sets the SMB protocol version used for the communication with the server.

For a complete list, see the **OPTIONS** section in the **mount.cifs(8)** man page.

# CHAPTER 63. OVERVIEW OF PERSISTENT NAMING ATTRIBUTES

As a system administrator, you need to refer to storage volumes using persistent naming attributes to build storage setups that are reliable over multiple system boots.

## 63.1. DISADVANTAGES OF NON-PERSISTENT NAMING ATTRIBUTES

Red Hat Enterprise Linux provides a number of ways to identify storage devices. It is important to use the correct option to identify each device when used in order to avoid inadvertently accessing the wrong device, particularly when installing to or reformatting drives.

Traditionally, non-persistent names in the form of **/dev/sd(major number)(minor number)** are used on Linux to refer to storage devices. The major and minor number range and associated **sd** names are allocated for each device when it is detected. This means that the association between the major and minor number range and associated **sd** names can change if the order of device detection changes.

Such a change in the ordering might occur in the following situations:

- The parallelization of the system boot process detects storage devices in a different order with each system boot.
- A disk fails to power up or respond to the SCSI controller. This results in it not being detected by the normal device probe. The disk is not accessible to the system and subsequent devices will have their major and minor number range, including the associated **sd** names shifted down. For example, if a disk normally referred to as **sdb** is not detected, a disk that is normally referred to as **sdc** would instead appear as **sdb**.
- A SCSI controller (host bus adapter, or HBA) fails to initialize, causing all disks connected to that HBA to not be detected. Any disks connected to subsequently probed HBAs are assigned different major and minor number ranges, and different associated **sd** names.
- The order of driver initialization changes if different types of HBAs are present in the system. This causes the disks connected to those HBAs to be detected in a different order. This might also occur if HBAs are moved to different PCI slots on the system.
- Disks connected to the system with Fibre Channel, iSCSI, or FCoE adapters might be inaccessible at the time the storage devices are probed, due to a storage array or intervening switch being powered off, for example. This might occur when a system reboots after a power failure, if the storage array takes longer to come online than the system takes to boot. Although some Fibre Channel drivers support a mechanism to specify a persistent SCSI target ID to WWPN mapping, this does not cause the major and minor number ranges, and the associated **sd** names to be reserved; it only provides consistent SCSI target ID numbers.

These reasons make it undesirable to use the major and minor number range or the associated **sd** names when referring to devices, such as in the **/etc/fstab** file. There is the possibility that the wrong device will be mounted and data corruption might result.

Occasionally, however, it is still necessary to refer to the **sd** names even when another mechanism is used, such as when errors are reported by a device. This is because the Linux kernel uses **sd** names (and also SCSI host/channel/target/LUN tuples) in kernel messages regarding the device.

## 63.2. FILE SYSTEM AND DEVICE IDENTIFIERS

This section explains the difference between persistent attributes identifying file systems and block devices.

## File system identifiers

File system identifiers are tied to a particular file system created on a block device. The identifier is also stored as part of the file system. If you copy the file system to a different device, it still carries the same file system identifier. On the other hand, if you rewrite the device, such as by formatting it with the **mkfs** utility, the device loses the attribute.

File system identifiers include:

- Unique identifier (UUID)
- Label

## Device identifiers

Device identifiers are tied to a block device: for example, a disk or a partition. If you rewrite the device, such as by formatting it with the **mkfs** utility, the device keeps the attribute, because it is not stored in the file system.

Device identifiers include:

- World Wide Identifier (WWID)
- Partition UUID
- Serial number

## Recommendations

- Some file systems, such as logical volumes, span multiple devices. Red Hat recommends accessing these file systems using file system identifiers rather than device identifiers.

## 63.3. DEVICE NAMES MANAGED BY THE UDEV MECHANISM IN /DEV/DISK/

This section lists different kinds of persistent naming attributes that the **udev** service provides in the **/dev/disk/** directory.

The **udev** mechanism is used for all types of devices in Linux, not just for storage devices. In the case of storage devices, Red Hat Enterprise Linux contains **udev** rules that create symbolic links in the **/dev/disk/** directory. This enables you to refer to storage devices by:

- Their content
- A unique identifier
- Their serial number.

Although **udev** naming attributes are persistent, in that they do not change on their own across system reboots, some are also configurable.

### 63.3.1. File system identifiers

#### The UUID attribute in /dev/disk/by-uuid/

Entries in this directory provide a symbolic name that refers to the storage device by a **unique identifier** (UUID) in the content (that is, the data) stored on the device. For example:

`/dev/disk/by-uuid/3e6be9de-8139-11d1-9106-a43f08d823a6`

You can use the UUID to refer to the device in the **/etc/fstab** file using the following syntax:

`UUID=3e6be9de-8139-11d1-9106-a43f08d823a6`

You can configure the UUID attribute when creating a file system, and you can also change it later on.

#### The Label attribute in `/dev/disk/by-label/`

Entries in this directory provide a symbolic name that refers to the storage device by a **label** in the content (that is, the data) stored on the device.

For example:

`/dev/disk/by-label/Boot`

You can use the label to refer to the device in the **/etc/fstab** file using the following syntax:

`LABEL=Boot`

You can configure the Label attribute when creating a file system, and you can also change it later on.

### 63.3.2. Device identifiers

#### The WWID attribute in `/dev/disk/by-id/`

The World Wide Identifier (WWID) is a persistent, **system-independent identifier** that the SCSI Standard requires from all SCSI devices. The WWID identifier is guaranteed to be unique for every storage device, and independent of the path that is used to access the device. The identifier is a property of the device but is not stored in the content (that is, the data) on the devices.

This identifier can be obtained by issuing a SCSI Inquiry to retrieve the Device Identification Vital Product Data (page **0x83**) or Unit Serial Number (page **0x80**).

Red Hat Enterprise Linux automatically maintains the proper mapping from the WWID-based device name to a current `/dev/sd` name on that system. Applications can use the `/dev/disk/by-id/` name to reference the data on the disk, even if the path to the device changes, and even when accessing the device from different systems.

#### Example 63.1. WWID mappings

WWID symlink	Non-persistent device	Note
<code>/dev/disk/by-id/scsi-3600508b400105e210000900000490000</code>	<code>/dev/sda</code>	A device with a page <b>0x83</b> identifier
<code>/dev/disk/by-id/scsi-SSEAGATE_ST373453LW_3HW1RHM6</code>	<code>/dev/sdb</code>	A device with a page <b>0x80</b> identifier

WWID symlink	Non-persistent device	Note
<code>/dev/disk/by-id/ata-SAMSUNG_MZNLN256HMHQ-000L7_S2WDNX0J336519-part3</code>	<code>/dev/sdc3</code>	A disk partition

In addition to these persistent names provided by the system, you can also use **udev** rules to implement persistent names of your own, mapped to the WWID of the storage.

### The Partition UUID attribute in `/dev/disk/by-partuuid`

The Partition UUID (PARTUUID) attribute identifies partitions as defined by GPT partition table.

#### Example 63.2. Partition UUID mappings

PARTUUID symlink	Non-persistent device
<code>/dev/disk/by-partuuid/4cd1448a-01</code>	<code>/dev/sda1</code>
<code>/dev/disk/by-partuuid/4cd1448a-02</code>	<code>/dev/sda2</code>
<code>/dev/disk/by-partuuid/4cd1448a-03</code>	<code>/dev/sda3</code>

### The Path attribute in `/dev/disk/by-path/`

This attribute provides a symbolic name that refers to the storage device by the **hardware path** used to access the device.



#### WARNING

The Path attribute is unreliable, and Red Hat does not recommend using it.

## 63.4. THE WORLD WIDE IDENTIFIER WITH DM MULTIPATH

This section describes the mapping between the World Wide Identifier (WWID) and non-persistent device names in a Device Mapper Multipath configuration.

If there are multiple paths from a system to a device, DM Multipath uses the WWID to detect this. DM Multipath then presents a single "pseudo-device" in the `/dev/mapper/wwid` directory, such as `/dev/mapper/3600508b400105df70000e00000ac0000`.

The command **multipath -l** shows the mapping to the non-persistent identifiers:

- **Host:Channel:Target:LUN**
- **/dev/sd** name
- **major:minor** number

### Example 63.3. WWID mappings in a multipath configuration

An example output of the **multipath -l** command:

```
3600508b400105df70000e00000ac0000 dm-2 vendor,product
[size=20G][features=1 queue_if_no_path][hwandler=0][rw]
\ round-robin 0 [prio=0][active]
 \ 5:0:1:1 sdc 8:32 [active][undef]
 \ 6:0:1:1 sdg 8:96 [active][undef]
\ round-robin 0 [prio=0][enabled]
 \ 5:0:0:1 sdb 8:16 [active][undef]
 \ 6:0:0:1 sdf 8:80 [active][undef]
```

DM Multipath automatically maintains the proper mapping of each WWID-based device name to its corresponding **/dev/sd** name on the system. These names are persistent across path changes, and they are consistent when accessing the device from different systems.

When the **user\_friendly\_names** feature of DM Multipath is used, the WWID is mapped to a name of the form **/dev/mapper/mpathN**. By default, this mapping is maintained in the file **/etc/multipath/bindings**. These **mpathN** names are persistent as long as that file is maintained.



#### IMPORTANT

If you use **user\_friendly\_names**, then additional steps are required to obtain consistent names in a cluster.

## 63.5. LIMITATIONS OF THE UDEV DEVICE NAMING CONVENTION

The following are some limitations of the **udev** naming convention:

- It is possible that the device might not be accessible at the time the query is performed because the **udev** mechanism might rely on the ability to query the storage device when the **udev** rules are processed for a **udev** event. This is more likely to occur with Fibre Channel, iSCSI or FCoE storage devices when the device is not located in the server chassis.
- The kernel might send **udev** events at any time, causing the rules to be processed and possibly causing the **/dev/disk/by-\*/** links to be removed if the device is not accessible.
- There might be a delay between when the **udev** event is generated and when it is processed, such as when a large number of devices are detected and the user-space **udevd** service takes some amount of time to process the rules for each one. This might cause a delay between when the kernel detects the device and when the **/dev/disk/by-\*/** names are available.
- External programs such as **blkid** invoked by the rules might open the device for a brief period of time, making the device inaccessible for other uses.

## 63.6. LISTING PERSISTENT NAMING ATTRIBUTES

This procedure describes how to find out the persistent naming attributes of non-persistent storage devices.

## Procedure

- To list the UUID and Label attributes, use the **lsblk** utility:

```
$ lsblk --fs storage-device
```

For example:

### Example 63.4. Viewing the UUID and Label of a file system

```
$ lsblk --fs /dev/sda1
```

NAME	FSTYPE	LABEL	UUID	MOUNTPOINT
sda1	xfs	Boot	afa5d5e3-9050-48c3-acc1-bb30095f3dc4	/boot

- To list the PARTUUID attribute, use the **lsblk** utility with the **--output +PARTUUID** option:

```
$ lsblk --output +PARTUUID
```

For example:

### Example 63.5. Viewing the PARTUUID attribute of a partition

```
$ lsblk --output +PARTUUID /dev/sda1
```

NAME	MAJ:MIN	RM	SIZE	RO	TYPE	MOUNTPOINT	PARTUUID
sda1	8:1	0	512M	0	part	/boot	4cd1448a-01

- To list the WWID attribute, examine the targets of symbolic links in the **/dev/disk/by-id** directory. For example:

### Example 63.6. Viewing the WWID of all storage devices on the system

```
$ file /dev/disk/by-id/*

/dev/disk/by-id/ata-QEMU_HARDDISK_QM00001
symbolic link to ../../sda
/dev/disk/by-id/ata-QEMU_HARDDISK_QM00001-part1
symbolic link to ../../sda1
/dev/disk/by-id/ata-QEMU_HARDDISK_QM00001-part2
symbolic link to ../../sda2
/dev/disk/by-id/dm-name-rhel_rhel8-root
symbolic link to ../../dm-0
/dev/disk/by-id/dm-name-rhel_rhel8-swap
symbolic link to ../../dm-1
/dev/disk/by-id/dm-uuid-LVM-
QIWtEHtXGobe5bewIIUDivKOz5ofkgFhP0RMFsNyySVihqEl2cWWbR7MjXJolD6g
symbolic link to ../../dm-1
/dev/disk/by-id/dm-uuid-LVM-
```

```
QIWtEHtXGobe5bewIIUDivKOz5ofkgFhXqH2M45hD2H9nAf2qfWSrlRLhzfMyOKd
symbolic link to ../../dm-0
/dev/disk/by-id/lvm-pv-uuid-atlr2Y-vuMo-ueoH-CpMG-4JuH-AhEF-wu4QQm
symbolic link to ../../sda2
```

## 63.7. MODIFYING PERSISTENT NAMING ATTRIBUTES

This procedure describes how to change the UUID or Label persistent naming attribute of a file system.



### NOTE

Changing **udev** attributes happens in the background and might take a long time. The **udevadm settle** command waits until the change is fully registered, which ensures that your next command will be able to utilize the new attribute correctly.

In the following commands:

- Replace *new-uuid* with the UUID you want to set; for example, **1cdfbc07-1c90-4984-b5ec-f61943f5ea50**. You can generate a UUID using the **uuidgen** command.
- Replace *new-label* with a label; for example, **backup\_data**.

### Prerequisites

- If you are modifying the attributes of an XFS file system, unmount it first.

### Procedure

- To change the UUID or Label attributes of an **XFS** file system, use the **xfs\_admin** utility:

```
xfs_admin -U new-uuid -L new-label storage-device
udevadm settle
```

- To change the UUID or Label attributes of an **ext4**, **ext3**, or **ext2** file system, use the **tune2fs** utility:

```
tune2fs -U new-uuid -L new-label storage-device
udevadm settle
```

- To change the UUID or Label attributes of a swap volume, use the **swaplabel** utility:

```
swaplabel --uuid new-uuid --label new-label swap-device
udevadm settle
```

# CHAPTER 64. GETTING STARTED WITH PARTITIONS

As a system administrator, you can use the following procedures to create, delete, and modify various types of disk partitions.

For an overview of the advantages and disadvantages to using partitions on block devices, see the following KBase article: <https://access.redhat.com/solutions/163853>.

## 64.1. VIEWING THE PARTITION TABLE

As a system administrator, you can display the partition table of a block device to see the partition layout and details about individual partitions.

### 64.1.1. Viewing the partition table with parted

This procedure describes how to view the partition table on a block device using the **parted** utility.

#### Procedure

1. Start the interactive **parted** shell:

```
parted block-device
```

- Replace *block-device* with the path to the device you want to examine: for example, **/dev/sda**.

2. View the partition table:

```
(parted) print
```

3. Optionally, use the following command to switch to another device you want to examine next:

```
(parted) select block-device
```

#### Additional resources

- The **parted(8)** man page.

### 64.1.2. Example output of parted print

This section provides an example output of the **print** command in the **parted** shell and describes fields in the output.

#### Example 64.1. Output of the **print** command

```
Model: ATA SAMSUNG MZNLN256 (scsi)
Disk /dev/sda: 256GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:
```

Number	Start	End	Size	Type	File system	Flags
--------	-------	-----	------	------	-------------	-------

```

1 1049kB 269MB 268MB primary xfs boot
2 269MB 34.6GB 34.4GB primary
3 34.6GB 45.4GB 10.7GB primary
4 45.4GB 256GB 211GB extended
5 45.4GB 256GB 211GB logical

```

Following is a description of the fields:

**Model: ATA SAMSUNG MZNLN256 (scsi)**

The disk type, manufacturer, model number, and interface.

**Disk /dev/sda: 256GB**

The file path to the block device and the storage capacity.

**Partition Table: msdos**

The disk label type.

**Number**

The partition number. For example, the partition with minor number 1 corresponds to **/dev/sda1**.

**Start and End**

The location on the device where the partition starts and ends.

**Type**

Valid types are metadata, free, primary, extended, or logical.

**File system**

The file system type. If the **File system** field of a device shows no value, this means that its file system type is unknown. The **parted** utility cannot recognize the file system on encrypted devices.

**Flags**

Lists the flags set for the partition. Available flags are **boot**, **root**, **swap**, **hidden**, **raid**, **lvm**, or **lba**.

## 64.2. CREATING A PARTITION TABLE ON A DISK

As a system administrator, you can format a block device with different types of partition tables to enable using partitions on the device.

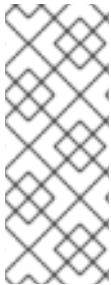


**WARNING**

Formatting a block device with a partition table deletes all data stored on the device.

### 64.2.1. Considerations before modifying partitions on a disk

This section lists key points to consider before creating, removing, or resizing partitions.



## NOTE

This section does not cover the DASD partition table, which is specific to the IBM Z architecture. For information on DASD, see:

- [Configuring a Linux instance on IBM Z](#)
- The [What you should know about DASD](#) article at the IBM Knowledge Center

### The maximum number of partitions

The number of partitions on a device is limited by the type of the partition table:

- On a device formatted with the **Master Boot Record (MBR)** partition table, you can have either:
  - Up to four primary partitions, or
  - Up to three primary partitions, one extended partition, and multiple logical partitions within the extended.
- On a device formatted with the **GUID Partition Table (GPT)** the maximum number of partitions is 128. While the GPT specification allows for more partitions by growing the area reserved for the partition table, common practice used by the **parted** utility is to limit it to enough area for 128 partitions.

### The maximum size of a partition

The size of a partition on a device is limited by the type of the partition table:

- On a device formatted with the **Master Boot Record (MBR)** partition table, the maximum size is 2TiB.
- On a device formatted with the **GUID Partition Table (GPT)** the maximum size is 8ZiB.

If you want to create a partition larger than 2TiB, the disk must be formatted with GPT.

### Size alignment

The **parted** utility enables you to specify partition size using multiple different suffixes:

#### MiB, GiB, or TiB

Size expressed in powers of 2.

- The starting point of the partition is aligned to the exact sector specified by size.
- The ending point is aligned to the specified size minus 1 sector.

#### MB, GB, or TB

Size expressed in powers of 10.

The starting and ending point is aligned within one half of the specified unit: for example,  $\pm 500\text{KB}$  when using the MB suffix.

### 64.2.2. Comparison of partition table types

This section compares the properties of different types of partition tables that you can create on a block device.

#### Table 64.1. Partition table types

Partition table	Maximum number of partitions	Maximum partition size
Master Boot Record (MBR)	4 primary, or 3 primary and 12 logical inside an extended partition	2TiB
GUID Partition Table (GPT)	128	8ZiB

### 64.2.3. Creating a partition table on a disk with parted

This procedure describes how to format a block device with a partition table using the **parted** utility.

#### Procedure

1. Start the interactive **parted** shell:

```
parted block-device
```

- Replace *block-device* with the path to the device where you want to create a partition table: for example, **/dev/sda**.

2. Determine if there already is a partition table on the device:

```
(parted) print
```

If the device already contains partitions, they will be deleted in the next steps.

3. Create the new partition table:

```
(parted) mklabel table-type
```

- Replace *table-type* with the intended partition table type:
  - **msdos** for MBR
  - **gpt** for GPT

#### Example 64.2. Creating a GPT table

For example, to create a GPT table on the disk, use:

```
(parted) mklabel gpt
```

The changes start taking place as soon as you enter this command, so review it before executing it.

4. View the partition table to confirm that the partition table exists:

```
(parted) print
```

5. Exit the **parted** shell:

(parted) quit

## Additional resources

- The **parted(8)** man page.

## Next steps

- Create partitions on the device. See [Section 64.3, “Creating a partition”](#) for details.

## 64.3. CREATING A PARTITION

As a system administrator, you can create new partitions on a disk.

### 64.3.1. Considerations before modifying partitions on a disk

This section lists key points to consider before creating, removing, or resizing partitions.



#### NOTE

This section does not cover the DASD partition table, which is specific to the IBM Z architecture. For information on DASD, see:

- [Configuring a Linux instance on IBM Z](#)
- The [What you should know about DASD](#) article at the IBM Knowledge Center

#### The maximum number of partitions

The number of partitions on a device is limited by the type of the partition table:

- On a device formatted with the **Master Boot Record (MBR)** partition table, you can have either:
  - Up to four primary partitions, or
  - Up to three primary partitions, one extended partition, and multiple logical partitions within the extended.
- On a device formatted with the **GUID Partition Table (GPT)** the maximum number of partitions is 128. While the GPT specification allows for more partitions by growing the area reserved for the partition table, common practice used by the **parted** utility is to limit it to enough area for 128 partitions.

#### The maximum size of a partition

The size of a partition on a device is limited by the type of the partition table:

- On a device formatted with the **Master Boot Record (MBR)** partition table, the maximum size is 2TiB.
- On a device formatted with the **GUID Partition Table (GPT)** the maximum size is 8ZiB.

If you want to create a partition larger than 2TiB, the disk must be formatted with GPT.

#### Size alignment

The **parted** utility enables you to specify partition size using multiple different suffixes:

## MiB, GiB, or TiB

Size expressed in powers of 2.

- The starting point of the partition is aligned to the exact sector specified by size.
- The ending point is aligned to the specified size minus 1 sector.

## MB, GB, or TB

Size expressed in powers of 10.

The starting and ending point is aligned within one half of the specified unit: for example,  $\pm 500\text{KB}$  when using the MB suffix.

### 64.3.2. Partition types

This section describes different attributes that specify the type of a partition.

#### Partition types or flags

The partition type, or flag, is used by a running system only rarely. However, the partition type matters to on-the-fly generators, such as **systemd-gpt-auto-generator**, which use the partition type to, for example, automatically identify and mount devices.

- The **parted** utility provides some control of partition types by mapping the partition type to flags. The parted utility can handle only certain partition types: for example LVM, swap, or RAID.
- The **fdisk** utility supports the full range of partition types by specifying hexadecimal codes.

#### Partition file system type

The **parted** utility optionally accepts a file system type argument when creating a partition. The value is used to:

- Set the partition flags on MBR, or
- Set the partition UUID type on GPT. For example, the **swap**, **fat**, or **hfs** file system types set different GUIDs. The default value is the Linux Data GUID.

The argument does not modify the file system on the partition in any way. It only differentiates between the supported flags or GUIDs.

The following file system types are supported:

- **xfs**
- **ext2**
- **ext3**
- **ext4**
- **fat16**
- **fat32**
- **hfs**
- **hfs+**

- **linux-swap**
- **ntfs**
- **reiserfs**

### 64.3.3. Creating a partition with parted

This procedure describes how to create a new partition on a block device using the **parted** utility.

#### Prerequisites

- There is a partition table on the disk. For details on how to format the disk, see [Section 64.2, "Creating a partition table on a disk"](#).
- If the partition you want to create is larger than 2TiB, the disk must be formatted with the GUID Partition Table (GPT).

#### Procedure

1. Start the interactive **parted** shell:

```
parted block-device
```

- Replace *block-device* with the path to the device where you want to create a partition: for example, **/dev/sda**.

2. View the current partition table to determine if there is enough free space:

```
(parted) print
```

- If there is not enough free space, you can resize an existing partition. For more information, see [Section 64.5, "Resizing a partition"](#).
- From the partition table, determine:
  - The start and end points of the new partition
  - On MBR, what partition type it should be.

3. Create the new partition:

```
(parted) mkpart part-type name fs-type start end
```

- Replace *part-type* with **primary**, **logical**, or **extended** based on what you decided from the partition table. This applies only to the MBR partition table.
- Replace *name* with an arbitrary partition name. This is required for GPT partition tables.
- Replace *fs-type* with any one of **xfs**, **ext2**, **ext3**, **ext4**, **fat16**, **fat32**, **hfs**, **hfs+**, **linux-swap**, **ntfs**, or **reiserfs**. The *fs-type* parameter is optional. Note that **parted** does not create the file system on the partition.

- Replace *start* and *end* with the sizes that determine the starting and ending points of the partition, counting from the beginning of the disk. You can use size suffixes, such as **512MiB**, **20GiB**, or **1.5TiB**. The default size megabytes.

### Example 64.3. Creating a small primary partition

For example, to create a primary partition from 1024MiB until 2048MiB on an MBR table, use:

```
(parted) mkpart primary 1024MiB 2048MiB
```

The changes start taking place as soon as you enter this command, so review it before executing it.

4. View the partition table to confirm that the created partition is in the partition table with the correct partition type, file system type, and size:

```
(parted) print
```

5. Exit the **parted** shell:

```
(parted) quit
```

6. Use the following command to wait for the system to register the new device node:

```
udevadm settle
```

7. Verify that the kernel recognizes the new partition:

```
cat /proc/partitions
```

## Additional resources

- The **parted(8)** man page.

### 64.3.4. Setting a partition type with fdisk

This procedure describes how to set a partition type, or flag, using the **fdisk** utility.

#### Prerequisites

- There is a partition on the disk.

#### Procedure

1. Start the interactive **fdisk** shell:

```
fdisk block-device
```

- Replace *block-device* with the path to the device where you want to set a partition type: for example, **/dev/sda**.

2. View the current partition table to determine the minor partition number:

Command (m for help): print

You can see the current partition type in the **Type** column and its corresponding type ID in the **Id** column.

3. Enter the partition type command and select a partition using its minor number:

Command (m for help): type  
Partition number (1,2,3 default 3): 2

4. Optionally, list the available hexadecimal codes:

Hex code (type L to list all codes): L

5. Set the partition type:

Hex code (type L to list all codes): 8e

6. Write your changes and exit the **fdisk** shell:

Command (m for help): write  
The partition table has been altered.  
Syncing disks.

7. Verify your changes:

# fdisk --list *block-device*

## 64.4. REMOVING A PARTITION

As a system administrator, you can remove a disk partition that is no longer used to free up disk space.



### WARNING

Removing a partition deletes all data stored on the partition.

#### 64.4.1. Considerations before modifying partitions on a disk

This section lists key points to consider before creating, removing, or resizing partitions.



## NOTE

This section does not cover the DASD partition table, which is specific to the IBM Z architecture. For information on DASD, see:

- [Configuring a Linux instance on IBM Z](#)
- The [What you should know about DASD](#) article at the IBM Knowledge Center

### The maximum number of partitions

The number of partitions on a device is limited by the type of the partition table:

- On a device formatted with the **Master Boot Record (MBR)** partition table, you can have either:
  - Up to four primary partitions, or
  - Up to three primary partitions, one extended partition, and multiple logical partitions within the extended.
- On a device formatted with the **GUID Partition Table (GPT)** the maximum number of partitions is 128. While the GPT specification allows for more partitions by growing the area reserved for the partition table, common practice used by the **parted** utility is to limit it to enough area for 128 partitions.

### The maximum size of a partition

The size of a partition on a device is limited by the type of the partition table:

- On a device formatted with the **Master Boot Record (MBR)** partition table, the maximum size is 2TiB.
- On a device formatted with the **GUID Partition Table (GPT)** the maximum size is 8ZiB.

If you want to create a partition larger than 2TiB, the disk must be formatted with GPT.

### Size alignment

The **parted** utility enables you to specify partition size using multiple different suffixes:

#### MiB, GiB, or TiB

Size expressed in powers of 2.

- The starting point of the partition is aligned to the exact sector specified by size.
- The ending point is aligned to the specified size minus 1 sector.

#### MB, GB, or TB

Size expressed in powers of 10.

The starting and ending point is aligned within one half of the specified unit: for example,  $\pm 500\text{KB}$  when using the MB suffix.

### 64.4.2. Removing a partition with parted

This procedure describes how to remove a disk partition using the **parted** utility.

#### Procedure

1. Start the interactive **parted** shell:

```
parted block-device
```

- Replace *block-device* with the path to the device where you want to remove a partition: for example, **/dev/sda**.

2. View the current partition table to determine the minor number of the partition to remove:

```
(parted) print
```

3. Remove the partition:

```
(parted) rm minor-number
```

- Replace *minor-number* with the minor number of the partition you want to remove: for example, **3**.

The changes start taking place as soon as you enter this command, so review it before executing it.

4. Confirm that the partition is removed from the partition table:

```
(parted) print
```

5. Exit the **parted** shell:

```
(parted) quit
```

6. Verify that the kernel knows the partition is removed:

```
cat /proc/partitions
```

7. Remove the partition from the **/etc/fstab** file if it is present. Find the line that declares the removed partition, and remove it from the file.

8. Regenerate mount units so that your system registers the new **/etc/fstab** configuration:

```
systemctl daemon-reload
```

9. If you have deleted a swap partition or removed pieces of LVM, remove all references to the partition from the kernel command line in the **/etc/default/grub** file and regenerate GRUB configuration:

- On a BIOS-based system:

```
grub2-mkconfig --output=/etc/grub2.cfg
```

- On a UEFI-based system:

```
grub2-mkconfig --output=/etc/grub2-efi.cfg
```

10. To register the changes in the early boot system, rebuild the **initramfs** file system:

```
dracut --force --verbose
```

## Additional resources

- The **parted(8)** man page

## 64.5. RESIZING A PARTITION

As a system administrator, you can extend a partition to utilize unused disk space, or shrink a partition to use its capacity for different purposes.

### 64.5.1. Considerations before modifying partitions on a disk

This section lists key points to consider before creating, removing, or resizing partitions.



#### NOTE

This section does not cover the DASD partition table, which is specific to the IBM Z architecture. For information on DASD, see:

- [Configuring a Linux instance on IBM Z](#)
- The [What you should know about DASD](#) article at the IBM Knowledge Center

#### The maximum number of partitions

The number of partitions on a device is limited by the type of the partition table:

- On a device formatted with the **Master Boot Record (MBR)** partition table, you can have either:
  - Up to four primary partitions, or
  - Up to three primary partitions, one extended partition, and multiple logical partitions within the extended.
- On a device formatted with the **GUID Partition Table (GPT)** the maximum number of partitions is 128. While the GPT specification allows for more partitions by growing the area reserved for the partition table, common practice used by the **parted** utility is to limit it to enough area for 128 partitions.

#### The maximum size of a partition

The size of a partition on a device is limited by the type of the partition table:

- On a device formatted with the **Master Boot Record (MBR)** partition table, the maximum size is 2TiB.
- On a device formatted with the **GUID Partition Table (GPT)** the maximum size is 8ZiB.

If you want to create a partition larger than 2TiB, the disk must be formatted with GPT.

#### Size alignment

The **parted** utility enables you to specify partition size using multiple different suffixes:

#### MiB, GiB, or TiB

Size expressed in powers of 2.

- The starting point of the partition is aligned to the exact sector specified by size.
- The ending point is aligned to the specified size minus 1 sector.

## MB, GB, or TB

Size expressed in powers of 10.

The starting and ending point is aligned within one half of the specified unit: for example,  $\pm 500\text{KB}$  when using the MB suffix.

### 64.5.2. Resizing a partition with parted

This procedure resizes a disk partition using the **parted** utility.

#### Prerequisites

- If you want to shrink a partition, back up the data that are stored on it.



#### WARNING

Shrinking a partition might result in data loss on the partition.

- If you want to resize a partition to be larger than 2TiB, the disk must be formatted with the GUID Partition Table (GPT). For details on how to format the disk, see [Section 64.2, “Creating a partition table on a disk”](#).

#### Procedure

1. If you want to shrink the partition, shrink the file system on it first so that it is not larger than the resized partition. Note that XFS does not support shrinking.
2. Start the interactive **parted** shell:

```
parted block-device
```

- Replace *block-device* with the path to the device where you want to resize a partition: for example, **/dev/sda**.

3. View the current partition table:

```
(parted) print
```

From the partition table, determine:

- The minor number of the partition
- The location of the existing partition and its new ending point after resizing

4. Resize the partition:

—

```
(parted) resizepart minor-number new-end
```

- Replace *minor-number* with the minor number of the partition that you are resizing: for example, **3**.
- Replace *new-end* with the size that determines the new ending point of the resized partition, counting from the beginning of the disk. You can use size suffixes, such as **512MiB**, **20GiB**, or **1.5TiB**. The default size megabytes.

#### Example 64.4. Extending a partition

For example, to extend a partition located at the beginning of the disk to be 2GiB in size, use:

```
(parted) resizepart 1 2GiB
```

The changes start taking place as soon as you enter this command, so review it before executing it.

5. View the partition table to confirm that the resized partition is in the partition table with the correct size:

```
(parted) print
```

6. Exit the **parted** shell:

```
(parted) quit
```

7. Verify that the kernel recognizes the new partition:

```
cat /proc/partitions
```

8. If you extended the partition, extend the file system on it as well. See (reference) for details.

### Additional resources

- The **parted(8)** man page.

# CHAPTER 65. GETTING STARTED WITH XFS

This is an overview of how to create and maintain XFS file systems.

## 65.1. THE XFS FILE SYSTEM

XFS is a highly scalable, high-performance, robust, and mature 64-bit journaling file system that supports very large files and file systems on a single host. It is the default file system in Red Hat Enterprise Linux 8. XFS was originally developed in the early 1990s by SGI and has a long history of running on extremely large servers and storage arrays.

The features of XFS include:

### Reliability

- Metadata journaling, which ensures file system integrity after a system crash by keeping a record of file system operations that can be replayed when the system is restarted and the file system remounted
- Extensive run-time metadata consistency checking
- Scalable and fast repair utilities
- Quota journaling. This avoids the need for lengthy quota consistency checks after a crash.

### Scalability and performance

- Supported file system size up to 1024 TiB
- Ability to support a large number of concurrent operations
- B-tree indexing for scalability of free space management
- Sophisticated metadata read-ahead algorithms
- Optimizations for streaming video workloads

### Allocation schemes

- Extent-based allocation
- Stripe-aware allocation policies
- Delayed allocation
- Space pre-allocation
- Dynamically allocated inodes

### Other features

- Reflink-based file copies (new in Red Hat Enterprise Linux 8)
- Tightly integrated backup and restore utilities
- Online defragmentation

- Online file system growing
- Comprehensive diagnostics capabilities
- Extended attributes (**xattr**). This allows the system to associate several additional name/value pairs per file.
- Project or directory quotas. This allows quota restrictions over a directory tree.
- Subsecond timestamps

## Performance characteristics

XFS has a high performance on large systems with enterprise workloads. A large system is one with a relatively high number of CPUs, multiple HBAs, and connections to external disk arrays. XFS also performs well on smaller systems that have a multi-threaded, parallel I/O workload.

XFS has a relatively low performance for single threaded, metadata-intensive workloads: for example, a workload that creates or deletes large numbers of small files in a single thread.

## 65.2. CREATING AN XFS FILE SYSTEM

As a system administrator, you can create an XFS file system on a block device to enable it to store files and directories.

### 65.2.1. Creating an XFS file system with **mkfs.xfs**

This procedure describes how to create an XFS file system on a block device.

#### Procedure

1. To create the file system:

- If the device is a regular partition, an LVM volume, an MD volume, a disk, or a similar device, use the following command:

```
mkfs.xfs block-device
```

- Replace *block-device* with the path to the block device. For example, **/dev/sdb1**, **/dev/disk/by-uuid/05e99ec8-def1-4a5e-8a9d-5945339ceb2a**, or **/dev/my-volgroup/my-lv**.
- In general, the default options are optimal for common use.
- When using **mkfs.xfs** on a block device containing an existing file system, add the **-f** option to overwrite that file system.

- To create the file system on a hardware RAID device, check if the system correctly detects the stripe geometry of the device:

- If the stripe geometry information is correct, no additional options are needed. Create the file system:

```
mkfs.xfs block-device
```

- If the information is incorrect, specify stripe geometry manually with the **su** and **sw** parameters of the **-d** option. The **su** parameter specifies the RAID chunk size, and the **sw** parameter specifies the number of data disks in the RAID device. For example:

```
mkfs.xfs -d su=64k,sw=4 /dev/sda3
```

2. Use the following command to wait for the system to register the new device node:

```
udevadm settle
```

## Additional resources

- The **mkfs.xfs(8)** man page.

### 65.2.2. Creating an XFS file system on a block device using RHEL System Roles

This section describes how to create an XFS file system on a block device on multiple target machines using the **storage** role.

#### Prerequisites

- An Ansible playbook that uses the **storage** role exists. For information on how to apply such a playbook, see [Applying a role](#).

#### 65.2.2.1. Example Ansible playbook to create an XFS file system on a block device

This section provides an example Ansible playbook. This playbook applies the **storage** role to create an XFS file system on a block device (**/dev/sdb**) using the default parameters.



#### WARNING

The **storage** role can create file systems only on whole disks. Partitions are not supported.

```

- hosts: all
 vars:
 storage_volumes:
 - name: barefs
 type: disk
 disks:
 - sdb
 fs_type: xfs
 roles:
 - rhel-system-roles.storage
```

The volume name (**barefs** in the example) is currently arbitrary. The volume is identified by the disk device listed under the **disks:** attribute.

XFS is the default file system type in RHEL 8, so **fs\_type: xfs** can be omitted.



### NOTE

To create a file system on a logical volume, do not provide the path to the LV device under the **disks:** attribute. Instead, provide the LVM setup including the enclosing volume group as described in [Configuring and managing logical volumes](#).

## 65.3. BACKING UP AN XFS FILE SYSTEM

As a system administrator, you can use the **xfsdump** to back up an XFS file system into a file or on a tape. This provides a simple backup mechanism.

### 65.3.1. Features of XFS backup

This section describes key concepts and features of backing up an XFS file system with the **xfsdump** utility.

You can use the **xfsdump** utility to:

- Perform backups to regular file images.  
Only one backup can be written to a regular file.
- Perform backups to tape drives.  
The **xfsdump** utility also enables you to write multiple backups to the same tape. A backup can span multiple tapes.

To back up multiple file systems to a single tape device, simply write the backup to a tape that already contains an XFS backup. This appends the new backup to the previous one. By default, **xfsdump** never overwrites existing backups.

- Create incremental backups.  
The **xfsdump** utility uses dump levels to determine a base backup to which other backups are relative. Numbers from 0 to 9 refer to increasing dump levels. An incremental backup only backs up files that have changed since the last dump of a lower level:
  - To perform a full backup, perform a level 0 dump on the file system.
  - A level 1 dump is the first incremental backup after a full backup. The next incremental backup would be level 2, which only backs up files that have changed since the last level 1 dump; and so on, to a maximum of level 9.
- Exclude files from a backup using size, subtree, or inode flags to filter them.

### Additional resources

- The **xfsdump(8)** man page.

### 65.3.2. Backing up an XFS file system with xfsdump

This procedure describes how to back up the content of an XFS file system into a file or a tape.

## Prerequisites

- An XFS file system that you can back up.
- Another file system or a tape drive where you can store the backup.

## Procedure

- Use the following command to back up an XFS file system:

```
xfsdump -l level [-L label] \
-f backup-destination path-to-xfs-filesystem
```

- Replace *level* with the dump level of your backup. Use **0** to perform a full backup or **1** to **9** to perform consequent incremental backups.
- Replace *backup-destination* with the path where you want to store your backup. The destination can be a regular file, a tape drive, or a remote tape device. For example, **/backup-files/Data.xfsdump** for a file or **/dev/st0** for a tape drive.
- Replace *path-to-xfs-filesystem* with the mount point of the XFS file system you want to back up. For example, **/mnt/data/**. The file system must be mounted.
- When backing up multiple file systems and saving them on a single tape device, add a session label to each backup using the **-L *label*** option so that it is easier to identify them when restoring. Replace *label* with any name for your backup: for example, **backup\_data**.

### Example 65.1. Backing up multiple XFS file systems

- To back up the content of XFS file systems mounted on the **/boot/** and **/data/** directories and save them as files in the **/backup-files/** directory:

```
xfsdump -l 0 -f /backup-files/boot.xfsdump /boot
xfsdump -l 0 -f /backup-files/data.xfsdump /data
```

- To back up multiple file systems on a single tape device, add a session label to each backup using the **-L *label*** option:

```
xfsdump -l 0 -L "backup_boot" -f /dev/st0 /boot
xfsdump -l 0 -L "backup_data" -f /dev/st0 /data
```

## Additional resources

- The **xfsdump(8)** man page.

### 65.3.3. Additional resources

- The **xfsdump(8)** man page.

## 65.4. RESTORING AN XFS FILE SYSTEM FROM BACKUP

As a system administrator, you can use the **xfsrestore** utility to restore XFS backup created with the **xfsdump** utility and stored in a file or on a tape.

### 65.4.1. Features of restoring XFS from backup

This section describes key concepts and features of restoring an XFS file system from backup with the **xfsrestore** utility.

The **xfsrestore** utility restores file systems from backups produced by **xfsdump**. The **xfsrestore** utility has two modes:

- The **simple** mode enables users to restore an entire file system from a level 0 dump. This is the default mode.
- The **cumulative** mode enables file system restoration from an incremental backup: that is, level 1 to level 9.

A unique *session ID* or *session label* identifies each backup. Restoring a backup from a tape containing multiple backups requires its corresponding session ID or label.

To extract, add, or delete specific files from a backup, enter the **xfsrestore** interactive mode. The interactive mode provides a set of commands to manipulate the backup files.

#### Additional resources

- The **xfsrestore(8)** man page.

### 65.4.2. Restoring an XFS file system from backup with xfsrestore

This procedure describes how to restore the content of an XFS file system from a file or tape backup.

#### Prerequisites

- A file or tape backup of XFS file systems, as described in [Section 65.3, “Backing up an XFS file system”](#).
- A storage device where you can restore the backup.

#### Procedure

- The command to restore the backup varies depending on whether you are restoring from a full backup or an incremental one, or are restoring multiple backups from a single tape device:

```
xfsrestore [-r] [-S session-id] [-L session-label] [-i]
-f backup-location restoration-path
```

- Replace *backup-location* with the location of the backup. This can be a regular file, a tape drive, or a remote tape device. For example, **/backup-files/Data.xfsdump** for a file or **/dev/st0** for a tape drive.
- Replace *restoration-path* with the path to the directory where you want to restore the file system. For example, **/mnt/data/**.
- To restore a file system from an incremental (level 1 to level 9) backup, add the **-r** option.
- To restore a backup from a tape device that contains multiple backups, specify the backup using the **-S** or **-L** options.

The **-S** option lets you choose a backup by its session ID, while the **-L** option lets you choose by the session label. To obtain the session ID and session labels, use the **xfsrestore -I** command.

Replace *session-id* with the session ID of the backup. For example, **b74a3586-e52e-4a4a-8775-c3334fa8ea2c**. Replace *session-label* with the session label of the backup. For example, **my\_backup\_session\_label**.

- To use **xfsrestore** interactively, use the **-i** option.

The interactive dialog begins after **xfsrestore** finishes reading the specified device.

Available commands in the interactive **xfsrestore** shell include **cd**, **ls**, **add**, **delete**, and **extract**; for a complete list of commands, use the **help** command.

### Example 65.2. Restoring Multiple XFS File Systems

- To restore the XFS backup files and save their content into directories under **/mnt/**:

```
xfsrestore -f /backup-files/boot.xfsdump /mnt/boot/
xfsrestore -f /backup-files/data.xfsdump /mnt/data/
```

- To restore from a tape device containing multiple backups, specify each backup by its session label or session ID:

```
xfsrestore -L "backup_boot" -f /dev/st0 /mnt/boot/
xfsrestore -S "45e9af35-efd2-4244-87bc-4762e476cbab" \
 -f /dev/st0 /mnt/data/
```

### Additional resources

- The **xfsrestore(8)** man page.

#### 65.4.3. Informational messages when restoring an XFS backup from a tape

When restoring a backup from a tape with backups from multiple file systems, the **xfsrestore** utility might issue messages. The messages inform you whether a match of the requested backup has been found when **xfsrestore** examines each backup on the tape in sequential order. For example:

```
xfsrestore: preparing drive
xfsrestore: examining media file 0
xfsrestore: inventory session uuid (8590224e-3c93-469c-a311-fc8f23029b2a) does not match the
media header's session uuid (7eda9f86-f1e9-4dfd-b1d4-c50467912408)
xfsrestore: examining media file 1
xfsrestore: inventory session uuid (8590224e-3c93-469c-a311-fc8f23029b2a) does not match the
media header's session uuid (7eda9f86-f1e9-4dfd-b1d4-c50467912408)
[...]
```

The informational messages keep appearing until the matching backup is found.

#### 65.4.4. Additional resources

- The **xfsrestore(8)** man page.

## 65.5. INCREASING THE SIZE OF AN XFS FILE SYSTEM

As a system administrator, you can increase the size of an XFS file system to utilize larger storage capacity.



### IMPORTANT

It is not currently possible to decrease the size of XFS file systems.

#### 65.5.1. Increasing the size of an XFS file system with `xfs_growfs`

This procedure describes how to grow an XFS file system using the `xfs_growfs` utility.

#### Prerequisites

- Ensure that the underlying block device is of an appropriate size to hold the resized file system later. Use the appropriate resizing methods for the affected block device.
- Mount the XFS file system.

#### Procedure

- While the XFS file system is mounted, use the `xfs_growfs` utility to increase its size:

```
xfs_growfs file-system -D new-size
```

- Replace *file-system* with the mount point of the XFS file system.
- With the **-D** option, replace *new-size* with the desired new size of the file system specified in the number of file system blocks.

To find out the block size in kB of a given XFS file system, use the `xfs_info` utility:

```
xfs_info block-device
...
data = bsize=4096
...
```

- Without the **-D** option, `xfs_growfs` grows the file system to the maximum size supported by the underlying device.

#### Additional resources

- The `xfs_growfs(8)` man page.

## 65.6. COMPARISON OF TOOLS USED WITH EXT4 AND XFS

This section compares which tools to use to accomplish common tasks on the ext4 and XFS file systems.

Task	ext4	XFS
Create a file system	<b>mkfs.ext4</b>	<b>mkfs.xfs</b>
File system check	<b>e2fsck</b>	<b>xfs_repair</b>
Resize a file system	<b>resize2fs</b>	<b>xfs_growfs</b>
Save an image of a file system	<b>e2image</b>	<b>xfs_metadump</b> and <b>xfs_mdrestore</b>
Label or tune a file system	<b>tune2fs</b>	<b>xfs_admin</b>
Back up a file system	<b>dump</b> and <b>restore</b>	<b>xfsdump</b> and <b>xfsrestore</b>
Quota management	<b>quota</b>	<b>xfs_quota</b>
File mapping	<b>filefrag</b>	<b>xfs_bmap</b>

# CHAPTER 66. MOUNTING FILE SYSTEMS

As a system administrator, you can mount file systems on your system to access data on them.

## 66.1. THE LINUX MOUNT MECHANISM

This section explains basic concepts of mounting file systems on Linux.

On Linux, UNIX, and similar operating systems, file systems on different partitions and removable devices (CDs, DVDs, or USB flash drives for example) can be attached to a certain point (the mount point) in the directory tree, and then detached again. While a file system is mounted on a directory, the original content of the directory is not accessible.

Note that Linux does not prevent you from mounting a file system to a directory with a file system already attached to it.

When mounting, you can identify the device by:

- a universally unique identifier (UUID): for example, **UUID=34795a28-ca6d-4fd8-a347-73671d0c19cb**
- a volume label: for example, **LABEL=home**
- a full path to a non-persistent block device: for example, **/dev/sda3**

When you mount a file system using the **mount** command without all required information, that is without the device name, the target directory, or the file system type, the **mount** utility reads the content of the **/etc/fstab** file to check if the given file system is listed there. The **/etc/fstab** file contains a list of device names and the directories in which the selected file systems are set to be mounted as well as the file system type and mount options. Therefore, when mounting a file system that is specified in **/etc/fstab**, the following command syntax is sufficient:

- Mounting by the mount point:

```
mount directory
```

- Mounting by the block device:

```
mount device
```

### Additional resources

- The **mount(8)** man page.
- For information on how to list persistent naming attributes such as the UUID, see [Section 63.6, “Listing persistent naming attributes”](#).

## 66.2. LISTING CURRENTLY MOUNTED FILE SYSTEMS

This procedure describes how to list all currently mounted file systems on the command line.

### Procedure

- To list all mounted file systems, use the **findmnt** utility:

```
$ findmnt
```

- To limit the listed file systems only to a certain file system type, add the **--types** option:

```
$ findmnt --types fs-type
```

For example:

#### Example 66.1. Listing only XFS file systems

```
$ findmnt --types xfs
```

TARGET	SOURCE	FSTYPE	OPTIONS
/	/dev/mapper/luks-5564ed00-6aac-4406-bfb4-c59bf5de48b5	xfs	rw,relatime
└─/boot	/dev/sda1	xfs	rw,relatime
└─/home	/dev/mapper/luks-9d185660-7537-414d-b727-d92ea036051e	xfs	rw,relatime

#### Additional resources

- The **findmnt(8)** man page.

## 66.3. MOUNTING A FILE SYSTEM WITH MOUNT

This procedure describes how to mount a file system using the **mount** utility.

#### Prerequisites

- Make sure that no file system is already mounted on your chosen mount point:

```
$ findmnt mount-point
```

#### Procedure

1. To attach a certain file system, use the **mount** utility:

```
mount device mount-point
```

#### Example 66.2. Mounting an XFS file system

For example, to mount a local XFS file system identified by UUID:

```
mount UUID=ea74bbec-536d-490c-b8d9-5b40bbd7545b /mnt/data
```

2. If **mount** cannot recognize the file system type automatically, specify it using the **--types** option:

```
mount --types type device mount-point
```

#### Example 66.3. Mounting an NFS file system

For example, to mount a remote NFS file system:

```
mount --types nfs4 host:/remote-export /mnt/nfs
```

#### Additional resources

- The **mount(8)** man page.

## 66.4. MOVING A MOUNT POINT

This procedure describes how to change the mount point of a mounted file system to a different directory.

#### Procedure

1. To change the directory in which a file system is mounted:

```
mount --move old-directory new-directory
```

#### Example 66.4. Moving a home file system

For example, to move the file system mounted in the **/mnt/userdirs** directory to the **/home** mount point:

```
mount --move /mnt/userdirs /home
```

2. Verify that the file system has been moved as expected:

```
$ findmnt
$ ls old-directory
$ ls new-directory
```

#### Additional resources

- The **mount(8)** man page.

## 66.5. UNMOUNTING A FILE SYSTEM WITH UOUNT

This procedure describes how to unmount a file system using the **umount** utility.

#### Procedure

1. Try unmounting the file system using either of the following commands:

- By mount point:

```
umount mount-point
```

- By device:

```
umount device
```

If the command fails with an error similar to the following, it means that the file system is in use because of a process is using resources on it:

```
umount: /run/media/user/FlashDrive: target is busy.
```

2. If the file system is in use, use the **fuser** utility to determine which processes are accessing it. For example:

```
$ fuser --mount /run/media/user/FlashDrive
/run/media/user/FlashDrive: 18351
```

Afterwards, terminate the processes using the file system and try unmounting it again.

## 66.6. COMMON MOUNT OPTIONS

This section lists some commonly used options of the **mount** utility.

You can use these options in the following syntax:

```
mount --options option1,option2,option3 device mount-point
```

**Table 66.1. Common mount options**

Option	Description
<b>async</b>	Enables asynchronous input and output operations on the file system.
<b>auto</b>	Enables the file system to be mounted automatically using the <b>mount -a</b> command.
<b>defaults</b>	Provides an alias for the <b>async,auto,dev,exec,nouser,rw,suid</b> options.
<b>exec</b>	Allows the execution of binary files on the particular file system.
<b>loop</b>	Mounts an image as a loop device.
<b>noauto</b>	Default behavior disables the automatic mount of the file system using the <b>mount -a</b> command.
<b>noexec</b>	Disallows the execution of binary files on the particular file system.
<b>nouser</b>	Disallows an ordinary user (that is, other than root) to mount and unmount the file system.
<b>remount</b>	Remounts the file system in case it is already mounted.

Option	Description
<b>ro</b>	Mounts the file system for reading only.
<b>rw</b>	Mounts the file system for both reading and writing.
<b>user</b>	Allows an ordinary user (that is, other than root) to mount and unmount the file system.

## 66.7. SHARING A MOUNT ON MULTIPLE MOUNT POINTS

As a system administrator, you can duplicate mount points to make the file systems accessible from multiple directories.

### 66.7.1. Types of shared mounts

There are multiple types of shared mounts that you can use. The difference between them is what happens when you mount another file system under one of the shared mount points. The shared mounts are implemented using the *shared subtrees* functionality.

The types are:

#### Private mount

This type does not receive or forward any propagation events.

When you mount another file system under either the duplicate or the original mount point, it is not reflected in the other.

#### Shared mount

This type creates an exact replica of a given mount point.

When a mount point is marked as a shared mount, any mount within the original mount point is reflected in it, and vice versa.

This is the default mount type of the root file system.

#### Slave mount

This type creates a limited duplicate of a given mount point.

When a mount point is marked as a slave mount, any mount within the original mount point is reflected in it, but no mount within a slave mount is reflected in its original.

#### Unbindable mount

This type prevents the given mount point from being duplicated whatsoever.

### 66.7.2. Creating a private mount point duplicate

This procedure duplicates a mount point as a private mount. File systems that you later mount under the duplicate or the original mount point are not reflected in the other.

#### Procedure

1. Create a virtual file system (VFS) node from the original mount point:

```
mount --bind original-dir original-dir
```

2. Mark the original mount point as private:

```
mount --make-private original-dir
```

Alternatively, to change the mount type for the selected mount point and all mount points under it, use the **--make-rprivate** option instead of **--make-private**.

3. Create the duplicate:

```
mount --bind original-dir duplicate-dir
```

#### Example 66.5. Duplicating `/media` into `/mnt` as a private mount point

1. Create a VFS node from the `/media` directory:

```
mount --bind /media /media
```

2. Mark the `/media` directory as private:

```
mount --make-private /media
```

3. Create its duplicate in `/mnt`:

```
mount --bind /media /mnt
```

4. It is now possible to verify that `/media` and `/mnt` share content but none of the mounts within `/media` appear in `/mnt`. For example, if the CD-ROM drive contains non-empty media and the `/media/cdrom` directory exists, use:

```
mount /dev/cdrom /media/cdrom
ls /media/cdrom
EFI GPL isolinux LiveOS
ls /mnt/cdrom
#
```

5. It is also possible to verify that file systems mounted in the `/mnt` directory are not reflected in `/media`. For instance, if a non-empty USB flash drive that uses the `/dev/sdc1` device is plugged in and the `/mnt/flashdisk` directory is present, use:

```
mount /dev/sdc1 /mnt/flashdisk
ls /media/flashdisk
ls /mnt/flashdisk
en-US publican.cfg
```

#### Additional resources

- The **mount(8)** man page.

### 66.7.3. Creating a shared mount point duplicate

This procedure duplicates a mount point as a shared mount. File systems that you later mount under the original directory or the duplicate are always reflected in the other.

#### Procedure

1. Create a virtual file system (VFS) node from the original mount point:

```
mount --bind original-dir original-dir
```

2. Mark the original mount point as shared:

```
mount --make-shared original-dir
```

Alternatively, to change the mount type for the selected mount point and all mount points under it, use the **--make-rshared** option instead of **--make-shared**.

3. Create the duplicate:

```
mount --bind original-dir duplicate-dir
```

#### Example 66.6. Duplicating `/media` into `/mnt` as a shared mount point

To make the `/media` and `/mnt` directories share the same content:

1. Create a VFS node from the `/media` directory:

```
mount --bind /media /media
```

2. Mark the `/media` directory as shared:

```
mount --make-shared /media
```

3. Create its duplicate in `/mnt`:

```
mount --bind /media /mnt
```

4. It is now possible to verify that a mount within `/media` also appears in `/mnt`. For example, if the CD-ROM drive contains non-empty media and the `/media/cdrom` directory exists, use:

```
mount /dev/cdrom /media/cdrom
ls /media/cdrom
EFI GPL isolinux LiveOS
ls /mnt/cdrom
EFI GPL isolinux LiveOS
```

5. Similarly, it is possible to verify that any file system mounted in the `/mnt` directory is reflected in `/media`. For instance, if a non-empty USB flash drive that uses the `/dev/sdc1` device is plugged in and the `/mnt/flashdisk` directory is present, use:

```
mount /dev/sdc1 /mnt/flashdisk
ls /media/flashdisk
```

```
en-US publican.cfg
ls /mnt/flashdisk
en-US publican.cfg
```

## Additional resources

- The **mount(8)** man page.

### 66.7.4. Creating a slave mount point duplicate

This procedure duplicates a mount point as a slave mount. File systems that you later mount under the original mount point are reflected in the duplicate but not the other way around.

#### Procedure

1. Create a virtual file system (VFS) node from the original mount point:

```
mount --bind original-dir original-dir
```

2. Mark the original mount point as shared:

```
mount --make-shared original-dir
```

Alternatively, to change the mount type for the selected mount point and all mount points under it, use the **--make-rshared** option instead of **--make-shared**.

3. Create the duplicate and mark it as slave:

```
mount --bind original-dir duplicate-dir
mount --make-slave duplicate-dir
```

#### Example 66.7. Duplicating `/media` into `/mnt` as a slave mount point

This example shows how to get the content of the `/media` directory to appear in `/mnt` as well, but without any mounts in the `/mnt` directory to be reflected in `/media`.

1. Create a VFS node from the `/media` directory:

```
mount --bind /media /media
```

2. Mark the `/media` directory as shared:

```
mount --make-shared /media
```

3. Create its duplicate in `/mnt` and mark it as slave:

```
mount --bind /media /mnt
mount --make-slave /mnt
```

4. Verify that a mount within `/media` also appears in `/mnt`. For example, if the CD-ROM drive contains non-empty media and the `/media/cdrom/` directory exists, use:

```
mount /dev/cdrom /media/cdrom
ls /media/cdrom
EFI GPL isolinux LiveOS
ls /mnt/cdrom
EFI GPL isolinux LiveOS
```

5. Also verify that file systems mounted in the **/mnt** directory are not reflected in **/media**. For instance, if a non-empty USB flash drive that uses the **/dev/sdc1** device is plugged in and the **/mnt/flashdisk** directory is present, use:

```
mount /dev/sdc1 /mnt/flashdisk
ls /media/flashdisk
ls /mnt/flashdisk
en-US publican.cfg
```

## Additional resources

- The **mount(8)** man page.

### 66.7.5. Preventing a mount point from being duplicated

This procedure marks a mount point as unbindable so that it is not possible to duplicate it in another mount point.

#### Procedure

- To change the type of a mount point to an unbindable mount, use:

```
mount --bind mount-point mount-point
mount --make-unbindable mount-point
```

Alternatively, to change the mount type for the selected mount point and all mount points under it, use the **--make-runnable** option instead of **--make-unbindable**.

Any subsequent attempt to make a duplicate of this mount fails with the following error:

```
mount --bind mount-point duplicate-dir
mount: wrong fs type, bad option, bad superblock on mount-point,
missing codepage or helper program, or other error
In some cases useful info is found in syslog - try
dmesg | tail or so
```

#### Example 66.8. Preventing **/media** from being duplicated

- To prevent the **/media** directory from being shared, use:

```
mount --bind /media /media
mount --make-unbindable /media
```

## Additional resources

- The **mount(8)** man page.

### 66.7.6. Related information

- The *Shared subtrees* article on Linux Weekly News: <https://lwn.net/Articles/159077/>.

## 66.8. PERSISTENTLY MOUNTING FILE SYSTEMS

As a system administrator, you can persistently mount file systems to configure non-removable storage.

### 66.8.1. The `/etc/fstab` file

This section describes the **/etc/fstab** configuration file, which controls persistent mount points of file systems. Using **/etc/fstab** is the recommended way to persistently mount file systems.

Each line in the **/etc/fstab** file defines a mount point of a file system. It includes six fields separated by white space:

1. The block device identified by a persistent attribute or a path in the **/dev** directory.
2. The directory where the device will be mounted.
3. The file system on the device.
4. Mount options for the file system. The option **defaults** means that the partition is mounted at boot time with default options. This section also recognizes **systemd** mount unit options in the **x-systemd.option** format.
5. Backup option for the **dump** utility.
6. Check order for the **fsck** utility.

#### Example 66.9. The `/boot` file system in `/etc/fstab`

Block device	Mount point	File system	Options	Backup	Check
UUID=ea74bbec-536d-490c-b8d9-5b40bbd7545b	/boot	xfs	defaults	0	0

The **systemd** service automatically generates mount units from entries in **/etc/fstab**.

## Additional resources

- The **fstab(5)** man page.
- The *fstab* section of the **systemd.mount(5)** man page.

### 66.8.2. Adding a file system to `/etc/fstab`

This procedure describes how to configure persistent mount point for a file system in the **/etc/fstab** configuration file.

## Procedure

- Find out the UUID attribute of the file system:

```
$ lsblk --fs storage-device
```

For example:

### Example 66.10. Viewing the UUID of a partition

```
$ lsblk --fs /dev/sda1
```

NAME	FSTYPE	LABEL	UUID	MOUNTPOINT
sda1	xfs	Boot	ea74bbec-536d-490c-b8d9-5b40bbd7545b	/boot

- If the mount point directory does not exist, create it:

```
mkdir --parents mount-point
```

- As root, edit the **/etc/fstab** file and add a line for the file system, identified by the UUID.  
For example:

### Example 66.11. The /boot mount point in /etc/fstab

```
UUID=ea74bbec-536d-490c-b8d9-5b40bbd7545b /boot xfs defaults 0 0
```

- Regenerate mount units so that your system registers the new configuration:

```
systemctl daemon-reload
```

- Try mounting the file system to verify that the configuration works:

```
mount mount-point
```

## Additional resources

- Other persistent attributes that you can use to identify the file system: [Section 63.3, “Device names managed by the udev mechanism in /dev/disk/”](#)

### 66.8.3. Persistently mounting a file system using RHEL System Roles

This section describes how to persistently mount a file system using the **storage** role.

## Prerequisites

- An Ansible playbook that uses the **storage** role exists.

For information on how to apply such a playbook, see [Applying a role](#).

### 66.8.3.1. Example Ansible playbook to persistently mount a file system

This section provides an example Ansible playbook. This playbook applies the **storage** role to immediately and persistently mount an XFS file system.

```

- hosts: all
 vars:
 storage_volumes:
 - name: barefs
 type: disk
 disks:
 - sdb
 fs_type: xfs
 mount_point: /mnt/data
 roles:
 - rhel-system-roles.storage
```



#### NOTE

Applying the storage role in this playbook adds the file system to the **/etc/fstab** file, and mounts the file system immediately. If the file system does not exist on the **/dev/sdb** disk device, it is created. If the mount point directory does not exist, it is created as well.

## 66.9. MOUNTING FILE SYSTEMS ON DEMAND

As a system administrator, you can configure file systems, such as NFS, to mount automatically on demand.

### 66.9.1. The **autofs** service

This section explains the benefits and basic concepts of the **autofs** service, used to mount file systems on demand.

One drawback of permanent mounting using the **/etc/fstab** configuration is that, regardless of how infrequently a user accesses the mounted file system, the system must dedicate resources to keep the mounted file system in place. This might affect system performance when, for example, the system is maintaining NFS mounts to many systems at one time.

An alternative to **/etc/fstab** is to use the kernel-based **autofs** service. It consists of the following components:

- A kernel module that implements a file system, and
- A user-space service that performs all of the other functions.

The **autofs** service can mount and unmount file systems automatically (on-demand), therefore saving system resources. It can be used to mount file systems such as NFS, AFS, SMBFS, CIFS, and local file systems.

#### Additional resources

- The **autofs(8)** man page.

### 66.9.2. The autofs configuration files

This section describes the usage and syntax of configuration files used by the **autofs** service.

#### The master map file

The **autofs** service uses **/etc/auto.master** (master map) as its default primary configuration file. This can be changed to use another supported network source and name using the **autofs** configuration in the **/etc/autofs.conf** configuration file in conjunction with the Name Service Switch (NSS) mechanism.

All on-demand mount points must be configured in the master map. Mount point, host name, exported directory, and options can all be specified in a set of files (or other supported network sources) rather than configuring them manually for each host.

The master map file lists mount points controlled by **autofs**, and their corresponding configuration files or network sources known as automount maps. The format of the master map is as follows:

**mount-point map-name options**

The variables used in this format are:

#### **mount-point**

The **autofs** mount point; for example, **/mnt/data/**.

#### **map-file**

The map source file, which contains a list of mount points and the file system location from which those mount points should be mounted.

#### **options**

If supplied, these apply to all entries in the given map, if they do not themselves have options specified.

#### Example 66.12. The **/etc/auto.master** file

The following is a sample line from **/etc/auto.master** file:

**/mnt/data /etc/auto.data**

#### Map files

Map files configure the properties of individual on-demand mount points.

The automounter creates the directories if they do not exist. If the directories exist before the automounter was started, the automounter will not remove them when it exits. If a timeout is specified, the directory is automatically unmounted if the directory is not accessed for the timeout period.

The general format of maps is similar to the master map. However, the options field appears between the mount point and the location instead of at the end of the entry as in the master map:

**mount-point options location**

The variables used in this format are:

#### **mount-point**

This refers to the **autofs** mount point. This can be a single directory name for an indirect mount or the full path of the mount point for direct mounts. Each direct and indirect map entry key (*mount-point*) can be followed by a space separated list of offset directories (subdirectory names each beginning with `/`) making them what is known as a multi-mount entry.

### options

When supplied, these are the mount options for the map entries that do not specify their own options. This field is optional.

### location

This refers to the file system location such as a local file system path (preceded with the Sun map format escape character `:` for map names beginning with `/`), an NFS file system or other valid file system location.

#### Example 66.13. A map file

The following is a sample from a map file; for example, `/etc/auto.misc`:

```
payroll -fstype=nfs4 personnel:/dev/disk/by-uuid/52b94495-e106-4f29-b868-fe6f6c2789b1
sales -fstype=xfs :/dev/disk/by-uuid/5564ed00-6aac-4406-bfb4-c59bf5de48b5
```

The first column in the map file indicates the **autofs** mount point: **sales** and **payroll** from the server called **personnel**. The second column indicates the options for the **autofs** mount. The third column indicates the source of the mount.

Following the given configuration, the **autofs** mount points will be `/home/payroll` and `/home/sales`. The `-fstype=` option is often omitted and is generally not needed for correct operation.

Using the given configuration, if a process requires access to an **autofs** unmounted directory such as `/home/payroll/2006/July.sxc`, the **autofs** service automatically mounts the directory.

### The amd map format

The **autofs** service recognizes map configuration in the **amd** format as well. This is useful if you want to reuse existing automounter configuration written for the **am-utils** service, which has been removed from Red Hat Enterprise Linux.

However, Red Hat recommends using the simpler **autofs** format described in the previous sections.

### Additional resources

- The **autofs(5)**, **autofs.conf(5)**, and **auto.master(5)** man pages.
- For details on the **amd** map format, see the `/usr/share/doc/autofs/README.amd-maps` file, which is provided by the **autofs** package.

### 66.9.3. Configuring autofs mount points

This procedure describes how to configure on-demand mount points using the **autofs** service.

#### Prerequisites

- Install the **autofs** package:

```
yum install autofs
```

- Start and enable the **autofs** service:

```
systemctl enable --now autofs
```

## Procedure

- Create a map file for the on-demand mount point, located at **/etc/auto.*identifier***. Replace *identifier* with a name that identifies the mount point.
- In the map file, fill in the mount point, options, and location fields as described in [Section 66.9.2, “The autofs configuration files”](#).
- Register the map file in the master map file, as described in [Section 66.9.2, “The autofs configuration files”](#).
- Try accessing content in the on-demand directory:

```
$ ls automounted-directory
```

### 66.9.4. Automounting NFS server user home directories with autofs service

This procedure describes how to configure the autofs service to mount user home directories automatically.

## Prerequisites

- The **autofs** package is installed.
- The **autofs** service is enabled and running.

## Procedure

- Specify the mount point and location of the map file by editing the **/etc/auto.master** file on a server on which you need to mount user home directories. To do so, add the following line into the **/etc/auto.master** file:

```
/home /etc/auto.home
```

- Create a map file with the name of **/etc/auto.home** on a server on which you need to mount user home directories, and edit the file with the following parameters:

```
* -fstype=nfs,rw,sync host.example.com:/home/&
```

You can skip **fstype** parameter, as it is **nfs** by default. For more information, see **autofs(5)** man page.

- Reload the **autofs** service:

```
systemctl reload autofs
```

### 66.9.5. Overriding or augmenting autofs site configuration files

It is sometimes useful to override site defaults for a specific mount point on a client system.

### Example 66.14. Initial conditions

For example, consider the following conditions:

- Automounter maps are stored in NIS and the **/etc/nsswitch.conf** file has the following directive:
 

```
automount: files nis
```
- The **auto.master** file contains:
 

```
+auto.master
```
- The NIS **auto.master** map file contains:
 

```
/home auto.home
```
- The NIS **auto.home** map contains:
 

```
beth fileserver.example.com:/export/home/beth
joe fileserver.example.com:/export/home/joe
* fileserver.example.com:/export/home/&
```
- The file map **/etc/auto.home** does not exist.

### Example 66.15. Mounting home directories from a different server

Given the preceding conditions, let's assume that the client system needs to override the NIS map **auto.home** and mount home directories from a different server.

- In this case, the client needs to use the following **/etc/auto.master** map:
 

```
/home /etc/auto.home
+auto.master
```
- The **/etc/auto.home** map contains the entry:
 

```
* host.example.com:/export/home/&
```

Because the automounter only processes the first occurrence of a mount point, the **/home** directory contains the content of **/etc/auto.home** instead of the NIS **auto.home** map.

### Example 66.16. Augmenting auto.home with only selected entries

Alternatively, to augment the site-wide **auto.home** map with just a few entries:

- Create an **/etc/auto.home** file map, and in it put the new entries. At the end, include the NIS **auto.home** map. Then the **/etc/auto.home** file map looks similar to:
 

```
mydir someserver:/export/mydir
+auto.home
```

- With these NIS **auto.home** map conditions, listing the content of the **/home** directory outputs:

```
$ ls /home
beth joe mydir
```

This last example works as expected because **autofs** does not include the contents of a file map of the same name as the one it is reading. As such, **autofs** moves on to the next map source in the **nsswitch** configuration.

### 66.9.6. Using LDAP to store automounter maps

This procedure configures **autofs** to store automounter maps in LDAP configuration rather than in **autofs** map files.

#### Prerequisites

- LDAP client libraries must be installed on all systems configured to retrieve automounter maps from LDAP. On Red Hat Enterprise Linux, the **openldap** package should be installed automatically as a dependency of the **autofs** package.

#### Procedure

- To configure LDAP access, modify the **/etc/openldap/ldap.conf** file. Ensure that the **BASE**, **URI**, and **schema** options are set appropriately for your site.
- The most recently established schema for storing automount maps in LDAP is described by the **rfc2307bis** draft. To use this schema, set it in the **/etc/autofs.conf** configuration file by removing the comment characters from the schema definition. For example:

#### Example 66.17. Setting autofs configuration

```
DEFAULT_MAP_OBJECT_CLASS="automountMap"
DEFAULT_ENTRY_OBJECT_CLASS="automount"
DEFAULT_MAP_ATTRIBUTE="automountMapName"
DEFAULT_ENTRY_ATTRIBUTE="automountKey"
DEFAULT_VALUE_ATTRIBUTE="automountInformation"
```

- Ensure that all other schema entries are commented in the configuration. The **automountKey** attribute replaces the **cn** attribute in the **rfc2307bis** schema. Following is an example of an LDAP Data Interchange Format (LDIF) configuration:

#### Example 66.18. LDF Configuration

```
extended LDIF
#
LDAPv3
base <> with scope subtree
filter: (&(objectclass=automountMap)(automountMapName=auto.master))
requesting: ALL
#
```

```
auto.master, example.com
dn: automountMapName=auto.master,dc=example,dc=com
objectClass: top
objectClass: automountMap
automountMapName: auto.master

extended LDIF
#
LDAPv3
base <automountMapName=auto.master,dc=example,dc=com> with scope subtree
filter: (objectclass=automount)
requesting: ALL
#

/home, auto.master, example.com
dn: automountMapName=auto.master,dc=example,dc=com
objectClass: automount
cn: /home

automountKey: /home
automountInformation: auto.home

extended LDIF
#
LDAPv3
base <> with scope subtree
filter: (&(objectclass=automountMap)(automountMapName=auto.home))
requesting: ALL
#

auto.home, example.com
dn: automountMapName=auto.home,dc=example,dc=com
objectClass: automountMap
automountMapName: auto.home

extended LDIF
#
LDAPv3
base <automountMapName=auto.home,dc=example,dc=com> with scope subtree
filter: (objectclass=automount)
requesting: ALL
#

foo, auto.home, example.com
dn: automountKey=foo,automountMapName=auto.home,dc=example,dc=com
objectClass: automount
automountKey: foo
automountInformation: filer.example.com:/export/foo

/, auto.home, example.com
dn: automountKey=/,automountMapName=auto.home,dc=example,dc=com
objectClass: automount
automountKey: /
automountInformation: filer.example.com:/export/amp;
```

## Additional resources

- The **rfc2307bis** draft: <https://tools.ietf.org/html/draft-howard-rfc2307bis>.

## 66.10. SETTING READ-ONLY PERMISSIONS FOR THE ROOT FILE SYSTEM

Sometimes, you need to mount the root file system (/) with read-only permissions. Example use cases include enhancing security or ensuring data integrity after an unexpected system power-off.

### 66.10.1. Files and directories that always retain write permissions

For the system to function properly, some files and directories need to retain write permissions. When the root file system is mounted in read-only mode, these files are mounted in RAM using the **tmpfs** temporary file system.

The default set of such files and directories is read from the **/etc/rwtab** file, which contains:

```
dirs /var/cache/man
dirs /var/gdm
<content truncated>

empty /tmp
empty /var/cache/foomatic
<content truncated>

files /etc/adjtime
files /etc/ntp.conf
<content truncated>
```

Entries in the **/etc/rwtab** file follow this format:

```
copy-method path
```

In this syntax:

- Replace *copy-method* with one of the keywords specifying how the file or directory is copied to **tmpfs**.
- Replace *path* with the path to the file or directory.

The **/etc/rwtab** file recognizes the following ways in which a file or directory can be copied to **tmpfs**:

#### **empty**

An empty path is copied to **tmpfs**. For example:

```
empty /tmp
```

#### **dirs**

A directory tree is copied to **tmpfs**, empty. For example:

```
dirs /var/run
```

## files

A file or a directory tree is copied to **tmpfs** intact. For example:

```
files /etc/resolv.conf
```

The same format applies when adding custom paths to **/etc/rwtab.d/**.

### 66.10.2. Configuring the root file system to mount with read-only permissions on boot

With this procedure, the root file system is mounted read-only on all following boots.

#### Procedure

1. In the **/etc/sysconfig/readonly-root** file, set the **readonly** option to **yes**:

```
Set to 'yes' to mount the file systems as read-only.
readonly=yes
```

2. Add the **ro** option in the root entry ( **/** ) in the **/etc/fstab** file:

```
/dev/mapper/luks-c376919e... / xfs x-systemd.device-timeout=0,ro 1 1
```

3. Add the **ro** option to the **GRUB\_CMDLINE\_LINUX** directive in the **/etc/default/grub** file and ensure that the directive does not contain **rw**:

```
GRUB_CMDLINE_LINUX="rhgb quiet... ro"
```

4. Recreate the GRUB2 configuration file:

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

5. If you need to add files and directories to be mounted with write permissions in the **tmpfs** file system, create a text file in the **/etc/rwtab.d/** directory and put the configuration there. For example, to mount the **/etc/example/file** file with write permissions, add this line to the **/etc/rwtab.d/example** file:

```
files /etc/example/file
```



#### IMPORTANT

Changes made to files and directories in **tmpfs** do not persist across boots.

6. Reboot the system to apply the changes.

#### Troubleshooting

- If you mount the root file system with read-only permissions by mistake, you can remount it with read-and-write permissions again using the following command:

```
mount -o remount,rw /
```

## CHAPTER 67. MANAGING STORAGE DEVICES

# CHAPTER 68. MANAGING LAYERED LOCAL STORAGE WITH STRATIS

You can easily set up and manage complex storage configurations integrated by the Stratis high-level system.



## IMPORTANT

Stratis is available as a Technology Preview. For information on Red Hat scope of support for Technology Preview features, see the [Technology Preview Features Support Scope](#) document.

Customers deploying Stratis are encouraged to provide feedback to Red Hat.

## 68.1. SETTING UP STRATIS FILE SYSTEMS

As a system administrator, you can enable and set up the Stratis volume-managing file system on your system to easily manage layered storage.

### 68.1.1. The purpose and features of Stratis

Stratis is a local storage-management solution for Linux. It is focused on simplicity and ease of use, and gives you access to advanced storage features.

Stratis makes the following activities easier:

- Initial configuration of storage
- Making changes later
- Using advanced storage features

Stratis is a hybrid user-and-kernel local storage management system that supports advanced storage features. The central concept of Stratis is a storage *pool*. This pool is created from one or more local disks or partitions, and volumes are created from the pool.

The pool enables many useful features, such as:

- File system snapshots
- Thin provisioning
- Tiering

### 68.1.2. Components of a Stratis volume

Externally, Stratis presents the following volume components in the command-line interface and the API:

#### **blockdev**

Block devices, such as a disk or a disk partition.

#### **pool**

Composed of one or more block devices.

A pool has a fixed total size, equal to the size of the block devices.

The pool contains most Stratis layers, such as the non-volatile data cache using the **dm-cache** target.

Stratis creates a **/stratis/my-pool** directory for each pool. This directory contains links to devices that represent Stratis file systems in the pool.

## filesystem

Each pool can contain one or more file systems, which store files.

File systems are thinly provisioned and do not have a fixed total size. The actual size of a file system grows with the data stored on it. If the size of the data approaches the virtual size of the file system, Stratis grows the thin volume and the file system automatically.

The file systems are formatted with XFS.



### IMPORTANT

Stratis tracks information about file systems created using Stratis that XFS is not aware of, and changes made using XFS do not automatically create updates in Stratis. Users must not reformat or reconfigure XFS file systems that are managed by Stratis.

Stratis creates links to file systems at the **/stratis/my-pool/my-fs** path.



### NOTE

Stratis uses many Device Mapper devices, which show up in **dmsetup** listings and the **/proc/partitions** file. Similarly, the **lsblk** command output reflects the internal workings and layers of Stratis.

### 68.1.3. Block devices usable with Stratis

This section lists storage devices that you can use for Stratis.

#### Supported devices

Stratis pools have been tested to work on these types of block devices:

- LUKS
- LVM logical volumes
- MD RAID
- DM Multipath
- iSCSI
- HDDs and SSDs
- NVMe devices

**WARNING**

In the current version, Stratis does not handle failures in hard drives or other hardware. If you create a Stratis pool over multiple hardware devices, you increase the risk of data loss because multiple devices must be operational to access the data.

**Unsupported devices**

Because Stratis contains a thin-provisioning layer, Red Hat does not recommend placing a Stratis pool on block devices that are already thinly-provisioned.

**Additional resources**

- For iSCSI and other block devices requiring network, see the **systemd.mount(5)** man page for information on the **\_netdev** mount option.

**68.1.4. Installing Stratis**

This procedure installs all packages necessary to use Stratis.

**Procedure**

- Install packages that provide the Stratis service and command-line utilities:

```
yum install stratisd stratis-cli
```

- Make sure that the **stratisd** service is enabled:

```
systemctl enable --now stratisd
```

**68.1.5. Creating a Stratis pool**

This procedure creates a Stratis pool from one or more block devices.

**Prerequisites**

- Stratis is installed. See [Section 68.1.4, “Installing Stratis”](#).
- The **stratisd** service is running.
- The block devices on which you are creating a Stratis pool are not in use and not mounted.
- The block devices on which you are creating a Stratis pool are at least 1 GiB in size each.
- On the IBM Z architecture, the **/dev/dasd\*** block devices must be partitioned. Use the partition in the Stratis pool.  
For information on partitioning DASD devices, see [Configuring a Linux instance on IBM Z](#).

**Procedure**

1. If the selected block device contains file system, partition table, or RAID signatures, erase them:

```
wipefs --all block-device
```

Replace *block-device* with the path to a block device, such as `/dev/sdb`.

2. To create a Stratis pool on the block device, use:

```
stratis pool create my-pool block-device
```

- Replace *my-pool* with an arbitrary name for the pool.
- Replace *block-device* with the path to the empty or wiped block device, such as `/dev/sdb`.

To create a pool from more than one block device, list them all on the command line:

```
stratis pool create my-pool device-1 device-2 device-n
```

3. To verify, list all pools on your system:

```
stratis pool list
```

## Additional resources

- The **stratis(8)** man page

## Next steps

- Create a Stratis file system on the pool. See [Section 68.1.6, “Creating a Stratis file system”](#).

### 68.1.6. Creating a Stratis file system

This procedure creates a Stratis file system on an existing Stratis pool.

#### Prerequisites

- Stratis is installed. See [Section 68.1.4, “Installing Stratis”](#).
- The **stratisd** service is running.
- You have created a Stratis pool. See [Section 68.1.5, “Creating a Stratis pool”](#).

#### Procedure

1. To create a Stratis file system on a pool, use:

```
stratis fs create my-pool my-fs
```

- Replace *my-pool* with the name of your existing Stratis pool.
- Replace *my-fs* with an arbitrary name for the file system.

2. To verify, list file systems within the pool:

■

```
stratis fs list my-pool
```

## Additional resources

- The **stratis(8)** man page

## Next steps

- Mount the Stratis file system. See [Section 68.1.7, “Mounting a Stratis file system”](#).

### 68.1.7. Mounting a Stratis file system

This procedure mounts an existing Stratis file system to access the content.

#### Prerequisites

- Stratis is installed. See [Section 68.1.4, “Installing Stratis”](#).
- The **stratisd** service is running.
- You have created a Stratis file system. See [Section 68.1.6, “Creating a Stratis file system”](#).

#### Procedure

- To mount the file system, use the entries that Stratis maintains in the **/stratis/** directory:

```
mount /stratis/my-pool/my-fs mount-point
```

The file system is now mounted on the *mount-point* directory and ready to use.

## Additional resources

- The **mount(8)** man page

### 68.1.8. Persistently mounting a Stratis file system

This procedure persistently mounts a Stratis file system so that it is available automatically after booting the system.

#### Prerequisites

- Stratis is installed. See [Section 68.1.4, “Installing Stratis”](#).
- The **stratisd** service is running.
- You have created a Stratis file system. See [Section 68.1.6, “Creating a Stratis file system”](#).

#### Procedure

1. Determine the UUID attribute of the file system:

```
$ lsblk --output=UUID /stratis/my-pool/my-fs
```

For example:

#### Example 68.1. Viewing the UUID of Stratis file system

```
$ lsblk --output=UUID /stratis/my-pool/fs1
UUID
a1f0b64a-4ebb-4d4e-9543-b1d79f600283
```

2. If the mount point directory does not exist, create it:

```
mkdir --parents mount-point
```

3. As root, edit the **/etc/fstab** file and add a line for the file system, identified by the UUID. Use **xfs** as the file system type and add the **x-systemd.requires=stratisd.service** option.

For example:

#### Example 68.2. The */fs1* mount point in */etc/fstab*

```
UUID=a1f0b64a-4ebb-4d4e-9543-b1d79f600283 /fs1 xfs defaults,x-
systemd.requires=stratisd.service 0 0
```

4. Regenerate mount units so that your system registers the new configuration:

```
systemctl daemon-reload
```

5. Try mounting the file system to verify that the configuration works:

```
mount mount-point
```

### Additional resources

- [Section 66.8, “Persistently mounting file systems”](#)

### 68.1.9. Related information

- The *Stratis Storage* website: <https://stratis-storage.github.io/>

## 68.2. EXTENDING A STRATIS VOLUME WITH ADDITIONAL BLOCK DEVICES

You can attach additional block devices to a Stratis pool to provide more storage capacity for Stratis file systems.

### 68.2.1. Components of a Stratis volume

Externally, Stratis presents the following volume components in the command-line interface and the API:

**blockdev**

Block devices, such as a disk or a disk partition.

**pool**

Composed of one or more block devices.

A pool has a fixed total size, equal to the size of the block devices.

The pool contains most Stratis layers, such as the non-volatile data cache using the **dm-cache** target.

Stratis creates a **/stratis/my-pool/** directory for each pool. This directory contains links to devices that represent Stratis file systems in the pool.

**filesystem**

Each pool can contain one or more file systems, which store files.

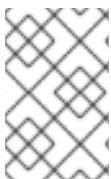
File systems are thinly provisioned and do not have a fixed total size. The actual size of a file system grows with the data stored on it. If the size of the data approaches the virtual size of the file system, Stratis grows the thin volume and the file system automatically.

The file systems are formatted with XFS.

**IMPORTANT**

Stratis tracks information about file systems created using Stratis that XFS is not aware of, and changes made using XFS do not automatically create updates in Stratis. Users must not reformat or reconfigure XFS file systems that are managed by Stratis.

Stratis creates links to file systems at the **/stratis/my-pool/my-fs** path.

**NOTE**

Stratis uses many Device Mapper devices, which show up in **dmsetup** listings and the **/proc/partitions** file. Similarly, the **lsblk** command output reflects the internal workings and layers of Stratis.

### 68.2.2. Adding block devices to a Stratis pool

This procedure adds one or more block devices to a Stratis pool to be usable by Stratis file systems.

#### Prerequisites

- Stratis is installed. See [Section 68.1.4, “Installing Stratis”](#).
- The **stratisd** service is running.
- The block devices that you are adding to the Stratis pool are not in use and not mounted.
- The block devices that you are adding to the Stratis pool are at least 1 GiB in size each.

#### Procedure

- To add one or more block devices to the pool, use:

```
stratis pool add-data my-pool device-1 device-2 device-n
```

## Additional resources

- The **stratis(8)** man page

### 68.2.3. Related information

- The *Stratis Storage* website: <https://stratis-storage.github.io/>

## 68.3. MONITORING STRATIS FILE SYSTEMS

As a Stratis user, you can view information about Stratis volumes on your system to monitor their state and free space.

### 68.3.1. Stratis sizes reported by different utilities

This section explains the difference between Stratis sizes reported by standard utilities such as **df** and the **stratis** utility.

Standard Linux utilities such as **df** report the size of the XFS file system layer on Stratis, which is 1 TiB. This is not useful information, because the actual storage usage of Stratis is less due to thin provisioning, and also because Stratis automatically grows the file system when the XFS layer is close to full.



#### IMPORTANT

Regularly monitor the amount of data written to your Stratis file systems, which is reported as the *Total Physical Used* value. Make sure it does not exceed the *Total Physical Size* value.

## Additional resources

- The **stratis(8)** man page

### 68.3.2. Displaying information about Stratis volumes

This procedure lists statistics about your Stratis volumes, such as the total, used, and free size or file systems and block devices belonging to a pool.

#### Prerequisites

- Stratis is installed. See [Section 68.1.4, “Installing Stratis”](#).
- The **stratisd** service is running.

#### Procedure

- To display information about all **block devices** used for Stratis on your system:

```
stratis blockdev
```

Pool Name	Device Node	Physical Size	State	Tier
<i>my-pool</i>	<i>/dev/sdb</i>	<i>9.10 TiB</i>	<i>In-use</i>	<i>Data</i>

- To display information about all Stratis **pools** on your system:

```
stratis pool

 Name Total Physical Size Total Physical Used
 my-pool 9.10 TiB 598 MiB
```

- To display information about all Stratis **file systems** on your system:

```
stratis filesystem

 Pool Name Name Used Created Device
 my-pool my-fs 546 MiB Nov 08 2018 08:03 /stratis/my-pool/my-fs
```

## Additional resources

- The **stratis(8)** man page

### 68.3.3. Related information

- The *Stratis Storage* website: <https://stratis-storage.github.io/>

## 68.4. USING SNAPSHOTS ON STRATIS FILE SYSTEMS

You can use snapshots on Stratis file systems to capture file system state at arbitrary times and restore it in the future.

### 68.4.1. Characteristics of Stratis snapshots

This section describes the properties and limitations of file system snapshots on Stratis.

In Stratis, a snapshot is a regular Stratis file system created as a copy of another Stratis file system. The snapshot initially contains the same file content as the original file system, but can change as the snapshot is modified. Whatever changes you make to the snapshot will not be reflected in the original file system.

The current snapshot implementation in Stratis is characterized by the following:

- A snapshot of a file system is another file system.
- A snapshot and its origin are not linked in lifetime. A snapshot file system can live longer than the file system it was created from.
- A file system does not have to be mounted to create a snapshot from it.
- Each snapshot uses around half a gigabyte of actual backing storage, which is needed for the XFS log.

### 68.4.2. Creating a Stratis snapshot

This procedure creates a Stratis file system as a snapshot of an existing Stratis file system.

## Prerequisites

- Stratis is installed. See [Section 68.1.4, “Installing Stratis”](#).
- The **stratisd** service is running.
- You have created a Stratis file system. See [Section 68.1.6, “Creating a Stratis file system”](#).

## Procedure

- To create a Stratis snapshot, use:

```
stratis fs snapshot my-pool my-fs my-fs-snapshot
```

## Additional resources

- The **stratis(8)** man page

### 68.4.3. Accessing the content of a Stratis snapshot

This procedure mounts a snapshot of a Stratis file system to make it accessible for read and write operations.

## Prerequisites

- Stratis is installed. See [Section 68.1.4, “Installing Stratis”](#).
- The **stratisd** service is running.
- You have created a Stratis snapshot. See [Section 68.4.2, “Creating a Stratis snapshot”](#).

## Procedure

- To access the snapshot, mount it as a regular file system from the **/stratis/*my-pool*** directory:

```
mount /stratis/my-pool/my-fs-snapshot mount-point
```

## Additional resources

- [Section 68.1.7, “Mounting a Stratis file system”](#)
- The **mount(8)** man page

### 68.4.4. Reverting a Stratis file system to a previous snapshot

This procedure reverts the content of a Stratis file system to the state captured in a Stratis snapshot.

## Prerequisites

- Stratis is installed. See [Section 68.1.4, “Installing Stratis”](#).
- The **stratisd** service is running.
- You have created a Stratis snapshot. See [Section 68.4.2, “Creating a Stratis snapshot”](#).

## Procedure

1. Optionally, back up the current state of the file system to be able to access it later:

```
stratis filesystem snapshot my-pool my-fs my-fs-backup
```

2. Unmount and remove the original file system:

```
umount /stratis/my-pool/my-fs
stratis filesystem destroy my-pool my-fs
```

3. Create a copy of the snapshot under the name of the original file system:

```
stratis filesystem snapshot my-pool my-fs-snapshot my-fs
```

4. Mount the snapshot, which is now accessible with the same name as the original file system:

```
mount /stratis/my-pool/my-fs mount-point
```

The content of the file system named *my-fs* is now identical to the snapshot *my-fs-snapshot*.

## Additional resources

- The **stratis(8)** man page

### 68.4.5. Removing a Stratis snapshot

This procedure removes a Stratis snapshot from a pool. Data on the snapshot are lost.

## Prerequisites

- Stratis is installed. See [Section 68.1.4, “Installing Stratis”](#).
- The **stratisd** service is running.
- You have created a Stratis snapshot. See [Section 68.4.2, “Creating a Stratis snapshot”](#).

## Procedure

1. Unmount the snapshot:

```
umount /stratis/my-pool/my-fs-snapshot
```

2. Destroy the snapshot:

```
stratis filesystem destroy my-pool my-fs-snapshot
```

## Additional resources

- The **stratis(8)** man page

### 68.4.6. Related information

- The *Stratis Storage* website: <https://stratis-storage.github.io/>

## 68.5. REMOVING STRATIS FILE SYSTEMS

You can remove an existing Stratis file system or a Stratis pool, destroying data on them.

### 68.5.1. Components of a Stratis volume

Externally, Stratis presents the following volume components in the command-line interface and the API:

#### **blockdev**

Block devices, such as a disk or a disk partition.

#### **pool**

Composed of one or more block devices.

A pool has a fixed total size, equal to the size of the block devices.

The pool contains most Stratis layers, such as the non-volatile data cache using the **dm-cache** target.

Stratis creates a **/stratis/my-pool** directory for each pool. This directory contains links to devices that represent Stratis file systems in the pool.

#### **filesystem**

Each pool can contain one or more file systems, which store files.

File systems are thinly provisioned and do not have a fixed total size. The actual size of a file system grows with the data stored on it. If the size of the data approaches the virtual size of the file system, Stratis grows the thin volume and the file system automatically.

The file systems are formatted with XFS.

#### **IMPORTANT**

 Stratis tracks information about file systems created using Stratis that XFS is not aware of, and changes made using XFS do not automatically create updates in Stratis. Users must not reformat or reconfigure XFS file systems that are managed by Stratis.

Stratis creates links to file systems at the **/stratis/my-pool/my-fs** path.

#### **NOTE**

 Stratis uses many Device Mapper devices, which show up in **dmsetup** listings and the **/proc/partitions** file. Similarly, the **lsblk** command output reflects the internal workings and layers of Stratis.

### 68.5.2. Removing a Stratis file system

This procedure removes an existing Stratis file system. Data stored on it are lost.

#### Prerequisites

- Stratis is installed. See [Section 68.1.4, “Installing Stratis”](#).
- The **stratisd** service is running.
- You have created a Stratis file system. See [Section 68.1.6, “Creating a Stratis file system”](#).

## Procedure

1. Unmount the file system:

```
umount /stratis/my-pool/my-fs
```

2. Destroy the file system:

```
stratis filesystem destroy my-pool my-fs
```

3. Verify that the file system no longer exists:

```
stratis filesystem list my-pool
```

## Additional resources

- The **stratis(8)** man page

### 68.5.3. Removing a Stratis pool

This procedure removes an existing Stratis pool. Data stored on it are lost.

## Prerequisites

- Stratis is installed. See [Section 68.1.4, “Installing Stratis”](#).
- The **stratisd** service is running.
- You have created a Stratis pool. See [Section 68.1.5, “Creating a Stratis pool”](#).

## Procedure

1. List file systems on the pool:

```
stratis filesystem list my-pool
```

2. Unmount all file systems on the pool:

```
umount /stratis/my-pool/my-fs-1 \
 /stratis/my-pool/my-fs-2 \
 /stratis/my-pool/my-fs-n
```

3. Destroy the file systems:

```
stratis filesystem destroy my-pool my-fs-1 my-fs-2
```

4. Destroy the pool:

```
stratis pool destroy my-pool
```

5. Verify that the pool no longer exists:

```
stratis pool list
```

#### Additional resources

- The **stratis(8)** man page

#### 68.5.4. Related information

- The *Stratis Storage* website: <https://stratis-storage.github.io/>

# CHAPTER 69. GETTING STARTED WITH SWAP

This section describes swap space and how to use it.

## 69.1. SWAP SPACE

Swap space in Linux is used when the amount of physical memory (RAM) is full. If the system needs more memory resources and the RAM is full, inactive pages in memory are moved to the swap space. While swap space can help machines with a small amount of RAM, it should not be considered a replacement for more RAM. Swap space is located on hard drives, which have a slower access time than physical memory. Swap space can be a dedicated swap partition (recommended), a swap file, or a combination of swap partitions and swap files.

In years past, the recommended amount of swap space increased linearly with the amount of RAM in the system. However, modern systems often include hundreds of gigabytes of RAM. As a consequence, recommended swap space is considered a function of system memory workload, not system memory.

[Section 69.2, “Recommended system swap space”](#) illustrates the recommended size of a swap partition depending on the amount of RAM in your system and whether you want sufficient memory for your system to hibernate. The recommended swap partition size is established automatically during installation. To allow for hibernation, however, you need to edit the swap space in the custom partitioning stage.

Recommendations in [Section 69.2, “Recommended system swap space”](#) are especially important on systems with low memory (1 GB and less). Failure to allocate sufficient swap space on these systems can cause issues such as instability or even render the installed system unbootable.

## 69.2. RECOMMENDED SYSTEM SWAP SPACE

This section gives recommendation about swap space.

Amount of RAM in the system	Recommended swap space	Recommended swap space if allowing for hibernation
≤ 2 GB	2 times the amount of RAM	3 times the amount of RAM
> 2 GB – 8 GB	Equal to the amount of RAM	2 times the amount of RAM
> 8 GB – 64 GB	At least 4 GB	1.5 times the amount of RAM
> 64 GB	At least 4 GB	Hibernation not recommended

At the border between each range listed in the table above, for example a system with 2 GB, 8 GB, or 64 GB of system RAM, discretion can be exercised with regard to chosen swap space and hibernation support. If your system resources allow for it, increasing the swap space may lead to better performance. A swap space of at least 100 GB is recommended for systems with over 140 logical processors or over 3 TB of RAM.

Note that distributing swap space over multiple storage devices also improves swap space performance, particularly on systems with fast drives, controllers, and interfaces.



## IMPORTANT

File systems and LVM2 volumes assigned as swap space *should not* be in use when being modified. Any attempts to modify swap fail if a system process or the kernel is using swap space. Use the **free** and **cat /proc/swaps** commands to verify how much and where swap is in use.

You should modify swap space while the system is booted in **rescue** mode, see [Debug boot options](#) in the *Performing an advanced RHEL installation*. When prompted to mount the file system, select **Skip**.

## 69.3. ADDING SWAP SPACE

This section describes how to add more swap space after installation. For example, you may upgrade the amount of RAM in your system from 1 GB to 2 GB, but there is only 2 GB of swap space. It might be advantageous to increase the amount of swap space to 4 GB if you perform memory-intense operations or run applications that require a large amount of memory.

There are three options: create a new swap partition, create a new swap file, or extend swap on an existing LVM2 logical volume. It is recommended that you extend an existing logical volume.

### 69.3.1. Extending swap on an LVM2 logical volume

This procedure describes how to extend swap space on an existing LVM2 logical volume. Assuming **/dev/VolGroup00/LogVol01** is the volume you want to extend by 2 GB.

#### Prerequisites

- Enough disk space.

#### Procedure

- Disable swapping for the associated logical volume:

```
swapoff -v /dev/VolGroup00/LogVol01
```

- Resize the LVM2 logical volume by 2 GB:

```
lvresize /dev/VolGroup00/LogVol01 -L +2G
```

- Format the new swap space:

```
mkswap /dev/VolGroup00/LogVol01
```

- Enable the extended logical volume:

```
swapon -v /dev/VolGroup00/LogVol01
```

- To test if the swap logical volume was successfully extended and activated, inspect active swap space:

```
$ cat /proc/swaps
$ free -h
```

### 69.3.2. Creating an LVM2 logical volume for swap

This procedure describes how to create an LVM2 logical volume for swap. Assuming **/dev/VolGroup00/LogVol02** is the swap volume you want to add.

#### Prerequisites

- Enough disk space.

#### Procedure

- Create the LVM2 logical volume of size 2 GB:

```
lvcreate VolGroup00 -n LogVol02 -L 2G
```

- Format the new swap space:

```
mkswap /dev/VolGroup00/LogVol02
```

- Add the following entry to the **/etc/fstab** file:

```
/dev/VolGroup00/LogVol02 swap swap defaults 0 0
```

- Regenerate mount units so that your system registers the new configuration:

```
systemctl daemon-reload
```

- Activate swap on the logical volume:

```
swapon -v /dev/VolGroup00/LogVol02
```

- To test if the swap logical volume was successfully created and activated, inspect active swap space:

```
$ cat /proc/swaps
$ free -h
```

### 69.3.3. Creating a swap file

This procedure describes how to create a swap file.

#### Prerequisites

- Enough disk space.

#### Procedure

- Determine the size of the new swap file in megabytes and multiply by 1024 to determine the number of blocks. For example, the block size of a 64 MB swap file is 65536.
- Create an empty file:

```
dd if=/dev/zero of=/swapfile bs=1024 count=65536
```

Replace *count* with the value equal to the desired block size.

3. Set up the swap file with the command:

```
mkswap /swapfile
```

4. Change the security of the swap file so it is not world readable.

```
chmod 0600 /swapfile
```

5. To enable the swap file at boot time, edit **/etc/fstab** as root to include the following entry:

```
/swapfile swap swap defaults 0 0
```

The next time the system boots, it activates the new swap file.

6. Regenerate mount units so that your system registers the new **/etc/fstab** configuration:

```
systemctl daemon-reload
```

7. To activate the swap file immediately:

```
swapon /swapfile
```

8. To test if the new swap file was successfully created and activated, inspect active swap space:

```
$ cat /proc/swaps
$ free -h
```

## 69.4. REMOVING SWAP SPACE

This section describes how to reduce swap space after installation. For example, you have downgraded the amount of RAM in your system from 1 GB to 512 MB, but there is 2 GB of swap space still assigned. It might be advantageous to reduce the amount of swap space to 1 GB, since the larger 2 GB could be wasting disk space.

Depending on what you need, you may choose one of three options: reduce swap space on an existing LVM2 logical volume, remove an entire LVM2 logical volume used for swap, or remove a swap file.

### 69.4.1. Reducing swap on an LVM2 logical volume

This procedure describes how to reduce swap on an LVM2 logical volume. Assuming **/dev/VolGroup00/LogVol01** is the volume you want to reduce.

#### Procedure

1. Disable swapping for the associated logical volume:

```
swapoff -v /dev/VolGroup00/LogVol01
```

2. Reduce the LVM2 logical volume by 512 MB:

```
lvreduce /dev/VolGroup00/LogVol01 -L -512M
```

3. Format the new swap space:

```
mkswap /dev/VolGroup00/LogVol01
```

4. Activate swap on the logical volume:

```
swapon -v /dev/VolGroup00/LogVol01
```

5. To test if the swap logical volume was successfully reduced, inspect active swap space:

```
$ cat /proc/swaps
$ free -h
```

#### 69.4.2. Removing an LVM2 logical volume for swap

This procedure describes how to remove an LVM2 logical volume for swap. Assuming **/dev/VolGroup00/LogVol02** is the swap volume you want to remove.

##### Procedure

1. Disable swapping for the associated logical volume:

```
swapoff -v /dev/VolGroup00/LogVol02
```

2. Remove the LVM2 logical volume:

```
lvremove /dev/VolGroup00/LogVol02
```

3. Remove the following associated entry from the **/etc/fstab** file:

```
/dev/VolGroup00/LogVol02 swap swap defaults 0 0
```

4. Regenerate mount units so that your system registers the new configuration:

```
systemctl daemon-reload
```

5. To test if the logical volume was successfully removed, inspect active swap space:

```
$ cat /proc/swaps
$ free -h
```

#### 69.4.3. Removing a swap file

This procedure describes how to remove a swap file.

##### Procedure

1. At a shell prompt, execute the following command to disable the swap file (where **/swapfile** is the swap file):

```
swapoff -v /swapfile
```

2. Remove its entry from the **/etc/fstab** file accordingly.
3. Regenerate mount units so that your system registers the new configuration:

```
systemctl daemon-reload
```

4. Remove the actual file:

```
rm /swapfile
```

## CHAPTER 70. DEDUPLICATING AND COMPRESSING STORAGE

# CHAPTER 71. DEPLOYING VDO

As a system administrator, you can use VDO to create deduplicated and compressed storage pools.

## 71.1. INTRODUCTION TO VDO

Virtual Data Optimizer (VDO) provides inline data reduction for Linux in the form of deduplication, compression, and thin provisioning. When you set up a VDO volume, you specify a block device on which to construct your VDO volume and the amount of logical storage you plan to present.

- When hosting active VMs or containers, Red Hat recommends provisioning storage at a 10:1 logical to physical ratio: that is, if you are utilizing 1 TB of physical storage, you would present it as 10 TB of logical storage.
- For object storage, such as the type provided by Ceph, Red Hat recommends using a 3:1 logical to physical ratio: that is, 1 TB of physical storage would present as 3 TB logical storage.

In either case, you can simply put a file system on top of the logical device presented by VDO and then use it directly or as part of a distributed cloud storage architecture.

Because VDO is thinly provisioned, the file system and applications only see the logical space in use and are not aware of the actual physical space available. Scripting should be used to monitor the actual available space and generate an alert if use exceeds a threshold: for example, when the VDO volume is 80% full. See [Section 72.1, “Managing free space on VDO volumes”](#) for details.

## 71.2. VDO DEPLOYMENT SCENARIOS

You can deploy VDO in a variety of ways to provide deduplicated storage for:

- both block and file access
- both local and remote storage

Because VDO exposes its deduplicated storage as a standard Linux block device, you can use it with standard file systems, iSCSI and FC target drivers, or as unified storage.

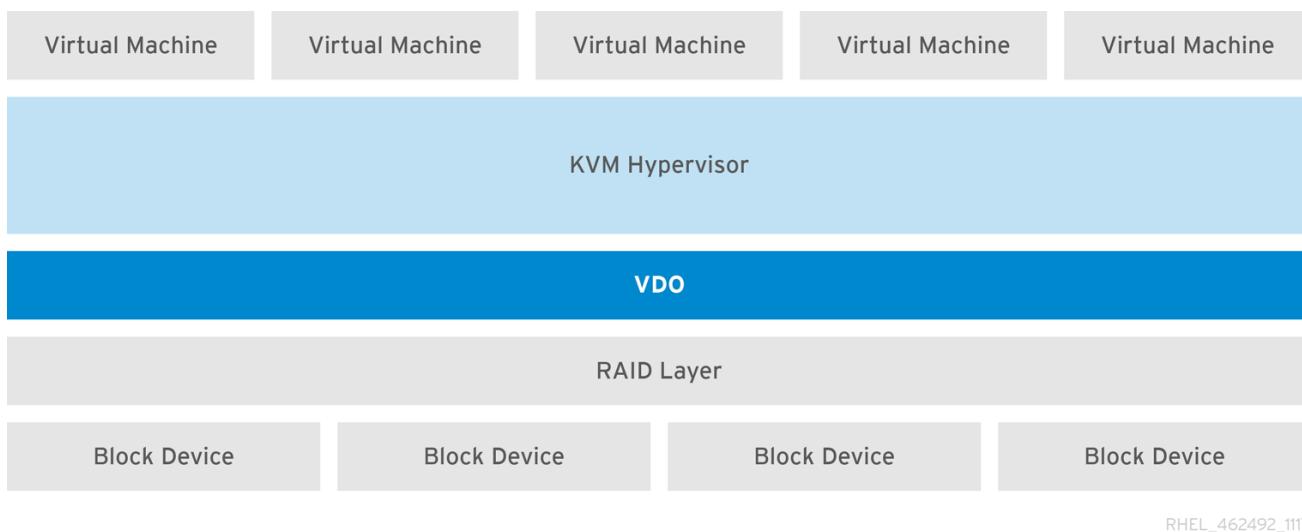


### NOTE

VDO deployment with Ceph Storage is currently not supported.

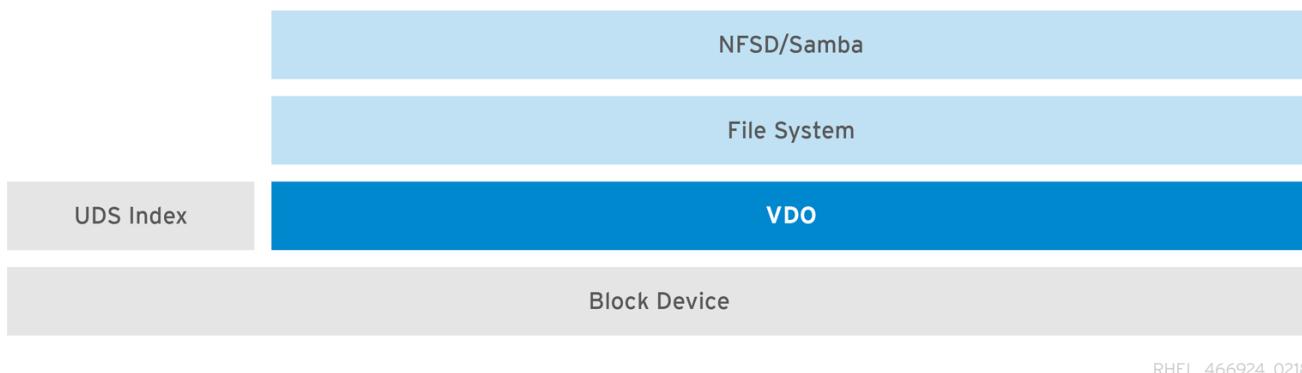
### KVM

You can deploy VDO on a KVM server configured with Direct Attached Storage.



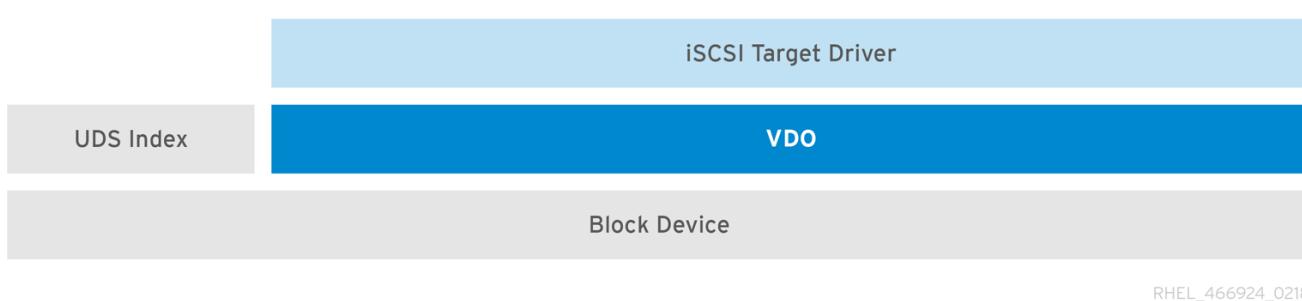
## File systems

You can create file systems on top of VDO and expose them to NFS or CIFS users with the NFS server or Samba.



## iSCSI target

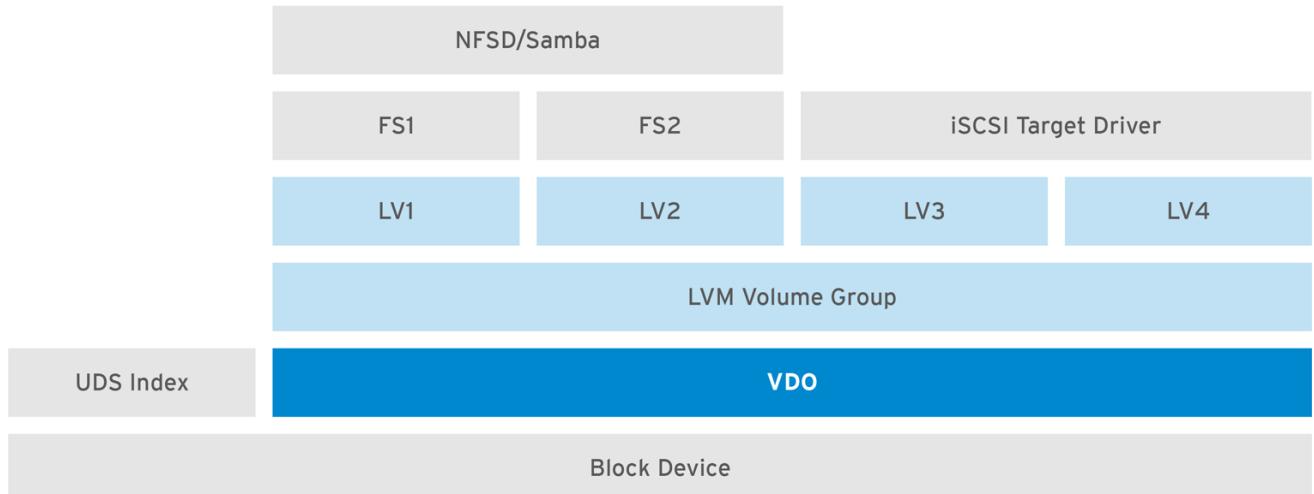
You can export the entirety of the VDO storage target as an iSCSI target to remote iSCSI initiators.



## LVM

On more feature-rich systems, you can use LVM to provide multiple logical unit numbers (LUNs) that are all backed by the same deduplicated storage pool.

In the following diagram, the VDO target is registered as a physical volume so that it can be managed by LVM. Multiple logical volumes (*LV1* to *LV4*) are created out of the deduplicated storage pool. In this way, VDO can support multiprotocol unified block or file access to the underlying deduplicated storage pool.

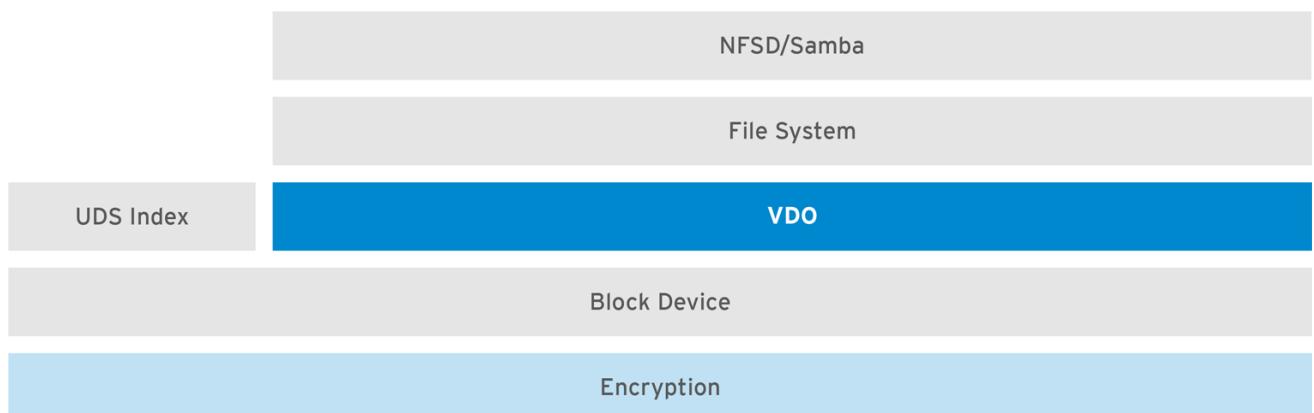


RHEL\_466924\_0218

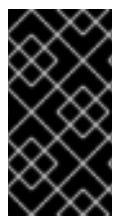
Deduplicated unified storage design enables for multiple file systems to collectively use the same deduplication domain through the LVM tools. Also, file systems can take advantage of LVM snapshot, copy-on-write, and shrink or grow features, all on top of VDO.

## Encryption

Device Mapper (DM) mechanisms such as DM Crypt are compatible with VDO. Encrypting VDO volumes helps ensure data security, and any file systems above VDO are still deduplicated.



RHEL\_466924\_0218



### IMPORTANT

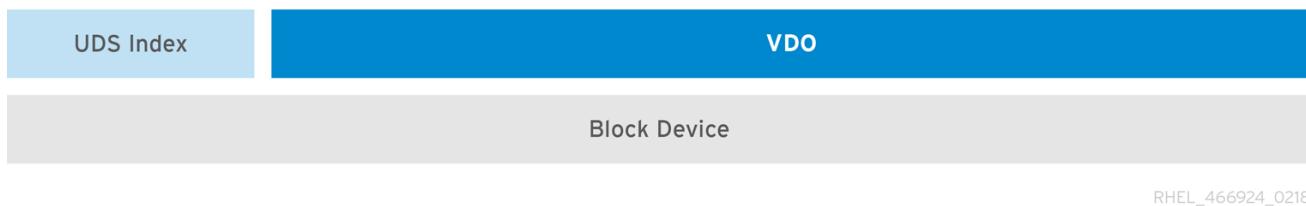
Applying the encryption layer above VDO results in little if any data deduplication. Encryption makes duplicate blocks different before VDO can deduplicate them.

Always place the encryption layer below VDO.

## 71.3. COMPONENTS OF A VDO VOLUME

VDO uses a block device as a backing store, which can include an aggregation of physical storage consisting of one or more disks, partitions, or even flat files. When a storage management tool creates a VDO volume, VDO reserves volume space for the UDS index and VDO volume. The UDS index and the VDO volume interact together to provide deduplicated block storage.

Figure 71.1. VDO disk organization



RHEL\_466924\_0218

The VDO solution consists of the following components:

### **kvdo**

A kernel module that loads into the Linux Device Mapper layer provides a deduplicated, compressed, and thinly provisioned block storage volume.

The **kvdo** module exposes a block device. You can access this block device directly for block storage or present it through a Linux file system, such as XFS or ext4.

When **kvdo** receives a request to read a logical block of data from a VDO volume, it maps the requested logical block to the underlying physical block and then reads and returns the requested data.

When **kvdo** receives a request to write a block of data to a VDO volume, it first checks whether the request is a DISCARD or TRIM request or whether the data is uniformly zero. If either of these conditions is true, **kvdo** updates its block map and acknowledges the request. Otherwise, VDO processes and optimizes the data.

### **uds**

A kernel module that communicates with the Universal Deduplication Service (UDS) index on the volume and analyzes data for duplicates. For each new piece of data, UDS quickly determines if that piece is identical to any previously stored piece of data. If the index finds a match, the storage system can then internally reference the existing item to avoid storing the same information more than once. The UDS index runs inside the kernel as the **uds** kernel module.

### Command line tools

For configuring and managing optimized storage.

## 71.4. THE PHYSICAL AND LOGICAL SIZE OF A VDO VOLUME

This section describes the physical size, available physical size, and logical size that VDO can utilize:

### Physical size

This is the same size as the underlying block device. VDO uses this storage for:

- User data, which might be deduplicated and compressed
- VDO metadata, such as the UDS index

### Available physical size

This is the portion of the physical size that VDO is able to use for user data

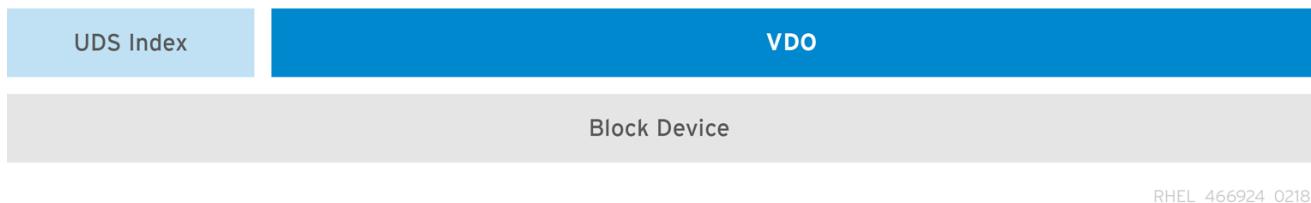
It is equivalent to the physical size minus the size of the metadata, minus the remainder after dividing the volume into slabs by the given slab size.

## Logical Size

This is the provisioned size that the VDO volume presents to applications. It is usually larger than the available physical size. If the **--vdoLogicalSize** option is not specified, then the provisioning of the logical volume is now provisioned to a **1:1** ratio. For example, if a VDO volume is put on top of a 20 GB block device, then 2.5 GB is reserved for the UDS index (if the default index size is used). The remaining 17.5 GB is provided for the VDO metadata and user data. As a result, the available storage to consume is not more than 17.5 GB, and can be less due to metadata that makes up the actual VDO volume.

VDO currently supports any logical size up to 254 times the size of the physical volume with an absolute maximum logical size of 4PB.

**Figure 71.2. VDO disk organization**



In this figure, the VDO deduplicated storage target sits completely on top of the block device, meaning the physical size of the VDO volume is the same size as the underlying block device.

## Additional resources

- For more information on how much storage VDO metadata requires on block devices of different sizes, see [Section 71.6.4, “Examples of VDO requirements by physical volume size”](#).

## 71.5. SLAB SIZE IN VDO

The physical storage of the VDO volume is divided into a number of slabs. Each slab is a contiguous region of the physical space. All of the slabs for a given volume have the same size, which can be any power of 2 multiple of 128 MB up to 32 GB.

The default slab size is 2 GB in order to facilitate evaluating VDO on smaller test systems. A single VDO volume can have up to 8192 slabs. Therefore, in the default configuration with 2 GB slabs, the maximum allowed physical storage is 16 TB. When using 32 GB slabs, the maximum allowed physical storage is 256 TB. VDO always reserves at least one entire slab for metadata, and therefore, the reserved slab cannot be used for storing user data.

Slab size has no effect on the performance of the VDO volume.

**Table 71.1. Recommended VDO slab sizes by physical volume size**

Physical volume size	Recommended slab size
10–99 GB	1 GB
100 GB – 1 TB	2 GB
2–256 TB	32 GB

You can control the slab size by providing the **--vdoSlabSize=megabytes** option to the **vdo create** command.

## 71.6. VDO REQUIREMENTS

VDO has certain requirements on its placement and your system resources.

### 71.6.1. Placement of VDO in the storage stack

You should place certain storage layers under VDO and others above VDO.

A VDO volume is a thinly provisioned block device. To prevent running out of physical space, place the volume on top of storage that you can expand at a later time. Examples of such expandable storage are LVM volumes or MD RAID arrays.

You can place thick-provisioned layers on top of VDO, but you cannot rely on the guarantees of thick provisioning in that case. Because the VDO layer is thin-provisioned, the effects of thin provisioning apply to all layers above it. If you do not monitor the VDO device, you might run out of physical space on thick-provisioned volumes above VDO.

Red Hat recommends the following configurations:

#### Place only under VDO

- DM Multipath
- DM Crypt
- Software RAID (LVM or MD RAID)

#### Place only above VDO

- LVM cache
- LVM snapshots
- LVM thin provisioning

The following configurations are **not supported**:

- VDO on top of VDO volumes: storage → VDO → LVM → VDO
- VDO on top of LVM snapshots
- VDO on top of LVM cache
- VDO on top of a loopback device
- VDO on top of LVM thin provisioning
- Encrypted volumes on top of VDO: storage → VDO → DM-Crypt
- Partitions on a VDO volume
- RAID (LVM RAID, MD RAID, or any other type) on top of a VDO volume

## Additional resources

- For more information on stacking VDO with LVM, see the [Stacking LVM volumes](#) article.

### 71.6.2. VDO memory requirements

Each VDO volume has two distinct memory requirements:

#### The VDO module

VDO requires 370 MB of RAM plus an additional 268 MB per each 1 TB of physical storage managed by the volume.

#### The Universal Deduplication Service (UDS) index

UDS requires a minimum of 250 MB of RAM, which is also the default amount that deduplication uses.

The memory required for the UDS index is determined by the index type and the required size of the deduplication window:

Index type	Deduplication window	Note
Dense	1 TB per 1 GB of RAM	A 1 GB dense index is generally sufficient for up to 4 TB of physical storage.
Sparse	10 TB per 1 GB of RAM	A 1 GB sparse index is generally sufficient for up to 40 TB of physical storage.

The UDS Sparse Indexing feature is the recommended mode for VDO. It relies on the temporal locality of data and attempts to retain only the most relevant index entries in memory. With the sparse index, UDS can maintain a deduplication window that is ten times larger than with dense, while using the same amount of memory.

Although the sparse index provides the greatest coverage, the dense index provides more deduplication advice. For most workloads, given the same amount of memory, the difference in deduplication rates between dense and sparse indexes is negligible.

## Additional resources

- For concrete examples of UDS index memory requirements, see [Section 71.6.4, “Examples of VDO requirements by physical volume size”](#).

### 71.6.3. VDO storage space requirements

You can configure a VDO volume to use up to 256 TB of physical storage. Only a certain part of the physical storage is usable to store data. This section provides the calculations to determine the usable size of a VDO-managed volume.

VDO requires storage for two types of VDO metadata and for the UDS index:

- The first type of VDO metadata uses approximately 1 MB for each 4 GB of *physical storage* plus an additional 1 MB per slab.
- The second type of VDO metadata consumes approximately 1.25 MB for each 1 GB of *logical storage*, rounded up to the nearest slab.

- The amount of storage required for the UDS index depends on the type of index and the amount of RAM allocated to the index. For each 1 GB of RAM, a dense UDS index uses 17 GB of storage, and a sparse UDS index will use 170 GB of storage.

## Additional resources

- For concrete examples of VDO storage requirements, see [Section 71.6.4, "Examples of VDO requirements by physical volume size"](#).

### 71.6.4. Examples of VDO requirements by physical volume size

The following tables provide approximate system requirements of VDO based on the size of the underlying physical volume. Each table lists requirements appropriate to the intended deployment, such as primary storage or backup storage.

The exact numbers depend on your configuration of the VDO volume.

#### Primary storage deployment

In the primary storage case, the UDS index is between 0.01% to 25% the size of the physical volume.

**Table 71.2. Storage and memory requirements for primary storage**

Physical volume size	RAM usage	Disk usage	Index type
10GB-1TB	250MB	2.5 GB	Dense
2-10TB	1GB	10GB	Dense
	250MB	22GB	Sparse
11-50TB	2GB	170GB	Sparse
51-100TB	3GB	255GB	Sparse
101-256TB	12GB	1020GB	Sparse

#### Backup storage deployment

In the backup storage case, the UDS index covers the size of the backup set but is not bigger than the physical volume. If you expect the backup set or the physical size to grow in the future, factor this into the index size.

**Table 71.3. Storage and memory requirements for backup storage**

Physical volume size	RAM usage	Disk usage	Index type
10GB-1TB	250MB	2.5 GB	Dense
2-10TB	2GB	170GB	Sparse
11-50TB	10GB	850GB	Sparse

Physical volume size	RAM usage	Disk usage	Index type
51-100TB	20GB	1700GB	Sparse
101-256TB	26GB	3400GB	Sparse

## 71.7. INSTALLING VDO

This procedure installs software necessary to create, mount, and manage VDO volumes.

### Procedure

- Install the **vdo** and **kmod-kvdo** packages:

```
yum install vdo kmod-kvdo
```

## 71.8. CREATING A VDO VOLUME

This procedure creates a VDO volume on a block device.

### Prerequisites

- Install the VDO software. See [Section 71.7, “Installing VDO”](#).
- Use expandable storage as the backing block device. For more information, see [Section 71.6.1, “Placement of VDO in the storage stack”](#).

### Procedure

In all the following steps, replace *vdo-name* with the identifier you want to use for your VDO volume; for example, **vdo1**. You must use a different name and device for each instance of VDO on the system.

- Find a persistent name for the block device where you want to create the VDO volume. For more information on persistent names, see [Chapter 63, Overview of persistent naming attributes](#).  
If you use a non-persistent device name, then VDO might fail to start properly in the future if the device name changes.
- Create the VDO volume:

```
vdo create \
--name=vdo-name \
--device=block-device \
--vdoLogicalSize=logical-size
```

- Replace *block-device* with the persistent name of the block device where you want to create the VDO volume. For example, **/dev/disk/by-id/scsi-3600508b1001c264ad2af21e903ad031f**.
- Replace *logical-size* with the amount of logical storage that the VDO volume should present:
  - For active VMs or container storage, use logical size that is **ten** times the physical size

of your block device. For example, if your block device is 1TB in size, use **10T** here.

- For object storage, use logical size that is **three** times the physical size of your block device. For example, if your block device is 1TB in size, use **3T** here.
- If the physical block device is larger than 16TiB, add the **--vdoSlabSize=32G** option to increase the slab size on the volume to 32GiB. Using the default slab size of 2GiB on block devices larger than 16TiB results in the **vdo create** command failing with the following error:

```
vdo: ERROR - vdoformat: formatVDO failed on '/dev/device': VDO Status: Exceeds maximum number of slabs supported
```

### Example 71.1. Creating VDO for container storage

For example, to create a VDO volume for container storage on a 1TB block device, you might use:

```
vdo create \
--name=vdo1 \
--device=/dev/disk/by-id/scsi-3600508b1001c264ad2af21e903ad031f \
--vdoLogicalSize=10T
```



### IMPORTANT

If a failure occurs when creating the VDO volume, remove the volume to clean up. See [Section 72.10.2, “Removing an unsuccessfully created VDO volume”](#) for details.

### 3. Create a file system on top of the VDO volume:

- For the XFS file system:

```
mkfs.xfs -K /dev/mapper/vdo-name
```

- For the ext4 file system:

```
mkfs.ext4 -E nodiscard /dev/mapper/vdo-name
```

### 4. Use the following command to wait for the system to register the new device node:

```
udevadm settle
```

## Next steps

- Mount the file system. See [Section 71.9, “Mounting a VDO volume”](#) for details.
- Enable the **discard** feature for the file system on your VDO device. See [Section 71.10, “Enabling periodic block discard”](#) for details.

## Additional resources

- The **vdo(8)** man page

## 71.9. MOUNTING A VDO VOLUME

This procedure mounts a file system on a VDO volume, either manually or persistently.

### Prerequisites

- A VDO volume has been created on your system. For instructions, see [Section 71.8, “Creating a VDO volume”](#).

### Procedure

- To mount the file system on the VDO volume manually, use:

```
mount /dev/mapper/vdo-name mount-point
```

- To configure the file system to mount automatically at boot, add a line to the **/etc/fstab** file:

- For the XFS file system:

```
/dev/mapper/vdo-name mount-point xfs defaults,_netdev,x-systemd.device-timeout=0,x-systemd.requires=vdo.service 0 0
```

- For the ext4 file system:

```
/dev/mapper/vdo-name mount-point ext4 defaults,_netdev,x-systemd.device-timeout=0,x-systemd.requires=vdo.service 0 0
```

### Additional resources

- The **vdo(8)** man page

## 71.10. ENABLING PERIODIC BLOCK DISCARD

This procedure enables a **systemd** timer that regularly discards unused blocks on all supported file systems.

### Procedure

- Enable and start the **systemd** timer:

```
systemctl enable --now fstrim.timer
```

## 71.11. MONITORING VDO

This procedure describes how to obtain usage and efficiency information from a VDO volume.

### Prerequisites

- Install the VDO software. See [Section 71.7, “Installing VDO”](#).

## Procedure

- Use the **vdostats** utility to get information about a VDO volume:

```
vdostats --human-readable

Device 1K-blocks Used Available Use% Space saving%
/dev/mapper/node1osd1 926.5G 21.0G 905.5G 2% 73%
/dev/mapper/node1osd2 926.5G 28.2G 898.3G 3% 64%
```

## Additional resources

- The **vdostats(8)** man page.

# CHAPTER 72. MAINTAINING VDO

After deploying a VDO volume, you can perform certain tasks to maintain or optimize it. Some of the following tasks are required for the correct functioning of VDO volumes.

## Prerequisites

- VDO is installed and deployed. See [Chapter 71, Deploying VDO](#).

## 72.1. MANAGING FREE SPACE ON VDO VOLUMES

VDO is a thinly provisioned block storage target. Because of that, you must actively monitor and manage space usage on VDO volumes.

### 72.1.1. The physical and logical size of a VDO volume

This section describes the physical size, available physical size, and logical size that VDO can utilize:

#### Physical size

This is the same size as the underlying block device. VDO uses this storage for:

- User data, which might be deduplicated and compressed
- VDO metadata, such as the UDS index

#### Available physical size

This is the portion of the physical size that VDO is able to use for user data

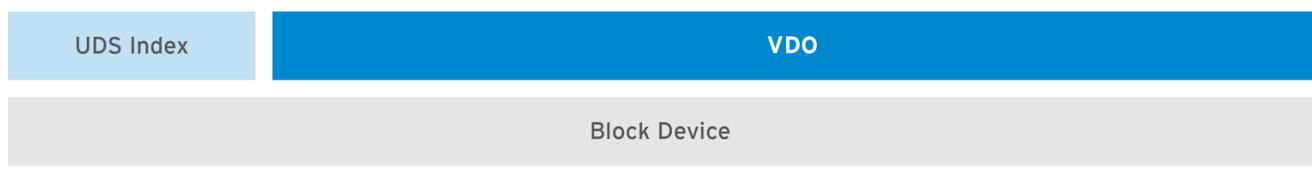
It is equivalent to the physical size minus the size of the metadata, minus the remainder after dividing the volume into slabs by the given slab size.

#### Logical Size

This is the provisioned size that the VDO volume presents to applications. It is usually larger than the available physical size. If the **--vdoLogicalSize** option is not specified, then the provisioning of the logical volume is now provisioned to a **1:1** ratio. For example, if a VDO volume is put on top of a 20 GB block device, then 2.5 GB is reserved for the UDS index (if the default index size is used). The remaining 17.5 GB is provided for the VDO metadata and user data. As a result, the available storage to consume is not more than 17.5 GB, and can be less due to metadata that makes up the actual VDO volume.

VDO currently supports any logical size up to 254 times the size of the physical volume with an absolute maximum logical size of 4PB.

Figure 72.1. VDO disk organization



RHEL\_466924\_0218

In this figure, the VDO deduplicated storage target sits completely on top of the block device, meaning the physical size of the VDO volume is the same size as the underlying block device.

## Additional resources

- For more information on how much storage VDO metadata requires on block devices of different sizes, see [Section 71.6.4, "Examples of VDO requirements by physical volume size"](#).

### 72.1.2. Thin provisioning in VDO

VDO is a thinly provisioned block storage target. The amount of physical space that a VDO volume uses might differ from the size of the volume that is presented to users of the storage. You can make use of this disparity to save on storage costs.

#### Out-of-space conditions

Take care to avoid unexpectedly running out of storage space, if the data written does not achieve the expected rate of optimization.

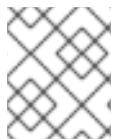
Whenever the number of logical blocks (virtual storage) exceeds the number of physical blocks (actual storage), it becomes possible for file systems and applications to unexpectedly run out of space. For that reason, storage systems using VDO must provide you with a way of monitoring the size of the free pool on the VDO volume.

You can determine the size of this free pool by using the **vdostats** utility. The default output of this utility lists information for all running VDO volumes in a format similar to the Linux **df** utility. For example:

Device	1K-blocks	Used	Available	Use%
/dev/mapper/vdo-name	211812352	105906176	105906176	50%

When the physical storage capacity of a VDO volume is almost full, VDO reports a warning in the system log, similar to the following:

```
Oct 2 17:13:39 system lvm[13863]: Monitoring VDO pool vdo-name.
Oct 2 17:27:39 system lvm[13863]: WARNING: VDO pool vdo-name is now 80.69% full.
Oct 2 17:28:19 system lvm[13863]: WARNING: VDO pool vdo-name is now 85.25% full.
Oct 2 17:29:39 system lvm[13863]: WARNING: VDO pool vdo-name is now 90.64% full.
Oct 2 17:30:29 system lvm[13863]: WARNING: VDO pool vdo-name is now 96.07% full.
```



#### NOTE

These warning messages appear only when the **lvm2-monitor** service is running. It is enabled by default.

#### How to prevent out-of-space conditions

If the size of free pool drops below a certain level, you can take action by:

- Deleting data. This reclaims space whenever the deleted data is not duplicated. Deleting data frees the space only after discards are issued.
- Adding physical storage



#### IMPORTANT

Monitor physical space on your VDO volumes to prevent out-of-space situations. Running out of physical blocks might result in losing recently written, unacknowledged data on the VDO volume.

## Thin provisioning and the TRIM and DISCARD commands

To benefit from the storage savings of thin provisioning, the physical storage layer needs to know when data is deleted. File systems that work with thinly provisioned storage send **TRIM** or **DISCARD** commands to inform the storage system when a logical block is no longer required.

Several methods of sending the **TRIM** or **DISCARD** commands are available:

- With the **discard** mount option, the file systems can send these commands whenever a block is deleted.
- You can send the commands in a controlled manner by using utilities such as **fstrim**. These utilities tell the file system to detect which logical blocks are unused and send the information to the storage system in the form of a **TRIM** or **DISCARD** command.

The need to use **TRIM** or **DISCARD** on unused blocks is not unique to VDO. Any thinly provisioned storage system has the same challenge.

### 72.1.3. Monitoring VDO

This procedure describes how to obtain usage and efficiency information from a VDO volume.

#### Prerequisites

- Install the VDO software. See [Section 71.7, “Installing VDO”](#).

#### Procedure

- Use the **vdostats** utility to get information about a VDO volume:

```
vdostats --human-readable

Device 1K-blocks Used Available Use% Space saving%
/dev/mapper/node1osd1 926.5G 21.0G 905.5G 2% 73%
/dev/mapper/node1osd2 926.5G 28.2G 898.3G 3% 64%
```

#### Additional resources

- The **vdostats(8)** man page.

### 72.1.4. Reclaiming space for VDO on file systems

This procedure reclaims storage space on a VDO volume that hosts a file system.

VDO cannot reclaim space unless file systems communicate that blocks are free using the **DISCARD**, **TRIM**, or **UNMAP** commands.

#### Procedure

- If the file system on your VDO volume supports discard operations, enable them. See [Discarding unused blocks](#).
- For file systems that do not use **DISCARD**, **TRIM**, or **UNMAP**, you can manually reclaim free space. Store a file consisting of binary zeros and then delete that file.

## 72.1.5. Reclaiming space for VDO without a file system

This procedure reclaims storage space on a VDO volume that is used as a block storage target without a file system.

### Procedure

- Use the **blkdiscard** utility.

For example, a single VDO volume can be carved up into multiple subvolumes by deploying LVM on top of it. Before deprovisioning a logical volume, use the **blkdiscard** utility to free the space previously used by that logical volume.

LVM supports the **REQ\_DISCARD** command and forwards the requests to VDO at the appropriate logical block addresses in order to free the space. If you use other volume managers, they also need to support **REQ\_DISCARD**, or equivalently, **UNMAP** for SCSI devices or **TRIM** for ATA devices.

### Additional resources

- The **blkdiscard(8)** man page

## 72.1.6. Reclaiming space for VDO on Fibre Channel or Ethernet network

This procedure reclaims storage space on VDO volumes (or portions of volumes) that are provisioned to hosts on a Fibre Channel storage fabric or an Ethernet network using SCSI target frameworks such as LIO or SCST.

### Procedure

- SCSI initiators can use the **UNMAP** command to free space on thinly provisioned storage targets, but the SCSI target framework needs to be configured to advertise support for this command. This is typically done by enabling thin provisioning on these volumes.

Verify support for **UNMAP** on Linux-based SCSI initiators by running the following command:

```
sg_vpd --page=0xb0 /dev/device
```

In the output, verify that the *Maximum unmap LBA count* value is greater than zero.

## 72.2. STARTING OR STOPPING VDO VOLUMES

You can start or stop a given VDO volume, or all VDO volumes, and their associated UDS indexes.

### 72.2.1. Started and activated VDO volumes

During the system boot, the **vdo systemd** unit automatically *starts* all VDO devices that are configured as *activated*.

The **vdo systemd** unit is installed and enabled by default when the **vdo** package is installed. This unit automatically runs the **vdo start --all** command at system startup to bring up all activated VDO volumes.

You can also create a VDO volume that does not start automatically by adding the **--activate=disabled** option to the **vdo create** command.

## The starting order

Some systems might place LVM volumes both above VDO volumes and below them. On these systems, it is necessary to start services in the right order:

1. The lower layer of LVM must start first. In most systems, starting this layer is configured automatically when the LVM package is installed.
2. The **vdo systemd** unit must start then.
3. Finally, additional scripts must run in order to start LVM volumes or other services on top of the running VDO volumes.

## How long it takes to stop a volume

Stopping a VDO volume takes time based on the speed of your storage device and the amount of data that the volume needs to write:

- The volume always writes around 1GiB for every 1GiB of the UDS index.
- With a sparse UDS index, the volume additionally writes the amount of data equal to the block map cache size plus up to 8MiB per slab.

### 72.2.2. Starting a VDO volume

This procedure starts a given VDO volume or all VDO volumes on your system.

#### Procedure

- To start a given VDO volume, use:

```
vdo start --name=my-vdo
```

- To start all VDO volumes, use:

```
vdo start --all
```

#### Additional resources

- The **vdo(8)** man page

### 72.2.3. Stopping a VDO volume

This procedure stops a given VDO volume or all VDO volumes on your system.

#### Procedure

1. Stop the volume.
  - To stop a given VDO volume, use:

```
vdo stop --name=my-vdo
```
  - To stop all VDO volumes, use:

```
vdo stop --all
```

2. Wait for the volume to finish writing data to the disk.

## Additional resources

- The **vdo(8)** man page

### 72.2.4. Related information

- If restarted after an unclean shutdown, VDO performs a rebuild to verify the consistency of its metadata and repairs it if necessary. See [Section 72.5, “Recovering a VDO volume after an unclean shutdown”](#) for more information on the rebuild process.

## 72.3. AUTOMATICALLY STARTING VDO VOLUMES AT SYSTEM BOOT

You can configure VDO volumes so that they start automatically at system boot. You can also disable the automatic start.

### 72.3.1. Started and activated VDO volumes

During the system boot, the **vdo systemd** unit automatically *starts* all VDO devices that are configured as *activated*.

The **vdo systemd** unit is installed and enabled by default when the **vdo** package is installed. This unit automatically runs the **vdo start --all** command at system startup to bring up all activated VDO volumes.

You can also create a VDO volume that does not start automatically by adding the **--activate=disabled** option to the **vdo create** command.

### The starting order

Some systems might place LVM volumes both above VDO volumes and below them. On these systems, it is necessary to start services in the right order:

1. The lower layer of LVM must start first. In most systems, starting this layer is configured automatically when the LVM package is installed.
2. The **vdo systemd** unit must start then.
3. Finally, additional scripts must run in order to start LVM volumes or other services on top of the running VDO volumes.

### How long it takes to stop a volume

Stopping a VDO volume takes time based on the speed of your storage device and the amount of data that the volume needs to write:

- The volume always writes around 1GiB for every 1GiB of the UDS index.
- With a sparse UDS index, the volume additionally writes the amount of data equal to the block map cache size plus up to 8MiB per slab.

### 72.3.2. Activating a VDO volume

This procedure activates a VDO volume to enable it to start automatically.

### Procedure

- To activate a specific volume:

```
vdo activate --name=my-vdo
```

- To activate all volumes:

```
vdo activate --all
```

### Additional resources

- The **vdo(8)** man page

## 72.3.3. Deactivating a VDO volume

This procedure deactivates a VDO volume to prevent it from starting automatically.

### Procedure

- To deactivate a specific volume:

```
vdo deactivate --name=my-vdo
```

- To deactivate all volumes:

```
vdo deactivate --all
```

### Additional resources

- The **vdo(8)** man page

## 72.4. SELECTING A VDO WRITE MODE

You can configure write mode for a VDO volume, based on what the underlying block device requires. By default, VDO selects write mode automatically.

### 72.4.1. VDO write modes

VDO supports the following write modes:

#### sync

When VDO is in **sync** mode, the layers above it assume that a write command writes data to persistent storage. As a result, it is not necessary for the file system or application, for example, to issue FLUSH or force unit access (FUA) requests to cause the data to become persistent at critical points.

VDO must be set to **sync** mode only when the underlying storage guarantees that data is written to persistent storage when the write command completes. That is, the storage must either have no volatile write cache, or have a write through cache.

## async

When VDO is in **async** mode, VDO does not guarantee that the data is written to persistent storage when a write command is acknowledged. The file system or application must issue FLUSH or FUA requests to ensure data persistence at critical points in each transaction.

VDO must be set to **async** mode if the underlying storage does not guarantee that data is written to persistent storage when the write command completes; that is, when the storage has a volatile write back cache.



### WARNING

When VDO is running in **async** mode, it is not compliant with Atomicity, Consistency, Isolation, Durability (ACID). When there is an application or a file system that assumes ACID compliance on top of the VDO volume, **async** mode might cause unexpected data loss.

## auto

The **auto** mode automatically selects **sync** or **async** based on the characteristics of each device. This is the default option.

### 72.4.2. The internal processing of VDO write modes

This section provides details on how the **sync** and **async** VDO write modes operate.

If the **kvdo** module is operating in synchronous mode:

1. It temporarily writes the data in the request to the allocated block and then acknowledges the request.
2. Once the acknowledgment is complete, an attempt is made to deduplicate the block by computing a MurmurHash-3 signature of the block data, which is sent to the VDO index.
3. If the VDO index contains an entry for a block with the same signature, **kvdo** reads the indicated block and does a byte-by-byte comparison of the two blocks to verify that they are identical.
4. If they are indeed identical, then **kvdo** updates its block map so that the logical block points to the corresponding physical block and releases the allocated physical block.
5. If the VDO index did not contain an entry for the signature of the block being written, or the indicated block does not actually contain the same data, **kvdo** updates its block map to make the temporary physical block permanent.

If **kvdo** is operating in asynchronous mode:

1. Instead of writing the data, it will immediately acknowledge the request.
2. It will then attempt to deduplicate the block in same manner as described above.

3. If the block turns out to be a duplicate, **kvdo** updates its block map and releases the allocated block. Otherwise, it writes the data in the request to the allocated block and updates the block map to make the physical block permanent.

### 72.4.3. Checking the write mode on a VDO volume

This procedure lists the active write mode on a selected VDO volume.

#### Procedure

- Use the following command to see the write mode used by a VDO volume:

```
vdo status --name=my-vdo
```

The output lists:

- The *configured write policy*, which is the option selected from **sync**, **async**, or **auto**
- The *write policy*, which is the particular write mode that VDO applied, that is either **sync** or **async**

### 72.4.4. Checking for a volatile cache

This procedure determines if a block device has a volatile cache or not. You can use the information to choose between the **sync** and **async** VDO write modes.

#### Procedure

1. Use either of the following methods to determine if a device has a writeback cache:

- Read the **/sys/block/*block-device*/device/scsi\_disk/*identifier*/cache\_type** sysfs file. For example:

```
$ cat '/sys/block/sda/device/scsi_disk/7:0:0:0/cache_type'
```

write back

```
$ cat '/sys/block/sdb/device/scsi_disk/1:2:0:0/cache_type'
```

None

- Alternatively, you can find whether the above mentioned devices have a write cache or not in the kernel boot log:

```
sd 7:0:0:0: [sda] Write cache: enabled, read cache: enabled, doesn't support DPO or
```

FUA

```
sd 1:2:0:0: [sdb] Write cache: disabled, read cache: disabled, supports DPO and FUA
```

2. In the previous examples:

- Device **sda** indicates that it *has* a writeback cache. Use **async** mode for it.
- Device **sdb** indicates that it *does not have* a writeback cache. Use **sync** mode for it.

You should configure VDO to use the **sync** write mode if the **cache\_type** value is **None** or **write through**.

### 72.4.5. Setting a VDO write mode

This procedure sets a write mode for a VDO volume, either for an existing one or when creating a new volume.



#### IMPORTANT

Using an incorrect write mode might result in data loss after a power failure, a system crash, or any unexpected loss of contact with the disk.

#### Prerequisites

- Determine which write mode is correct for your device. See [Section 72.4.4, “Checking for a volatile cache”](#).

#### Procedure

- You can set a write mode either on an existing VDO volume or when creating a new volume:
  - To modify an existing VDO volume, use:
 

```
vdo changeWritePolicy --writePolicy=sync/async/auto \
--name=vdo-name
```
  - To specify a write mode when creating a VDO volume, add the **--writePolicy=sync/async/auto** option to the **vdo create** command.

## 72.5. RECOVERING A VDO VOLUME AFTER AN UNCLEAN SHUTDOWN

You can recover a VDO volume after an unclean shutdown to enable it to continue operating. The task is mostly automated. Additionally, you can clean up after a VDO volume was unsuccessfully created because of a failure in the process.

### 72.5.1. VDO write modes

VDO supports the following write modes:

#### sync

When VDO is in **sync** mode, the layers above it assume that a write command writes data to persistent storage. As a result, it is not necessary for the file system or application, for example, to issue FLUSH or force unit access (FUA) requests to cause the data to become persistent at critical points.

VDO must be set to **sync** mode only when the underlying storage guarantees that data is written to persistent storage when the write command completes. That is, the storage must either have no volatile write cache, or have a write through cache.

#### async

When VDO is in **async** mode, VDO does not guarantee that the data is written to persistent storage when a write command is acknowledged. The file system or application must issue FLUSH or FUA requests to ensure data persistence at critical points in each transaction.

VDO must be set to **async** mode if the underlying storage does not guarantee that data is written to persistent storage when the write command completes; that is, when the storage has a volatile write back cache.



### WARNING

When VDO is running in **async** mode, it is not compliant with Atomicity, Consistency, Isolation, Durability (ACID). When there is an application or a file system that assumes ACID compliance on top of the VDO volume, **async** mode might cause unexpected data loss.

#### auto

The **auto** mode automatically selects **sync** or **async** based on the characteristics of each device. This is the default option.

#### 72.5.2. VDO volume recovery

When a VDO volume restarts after an unclean shutdown, VDO performs the following actions:

- Verifies the consistency of the metadata on the volume.
- Rebuilds a portion of the metadata to repair it if necessary.

Rebuilds are automatic and do not require user intervention.

VDO might rebuild different writes depending on the active write mode:

#### sync

If VDO was running on synchronous storage and write policy was set to **sync**, all data written to the volume are fully recovered.

#### async

If the write policy was **async**, some writes might not be recovered if they were not made durable. This is done by sending VDO a **FLUSH** command or a write I/O tagged with the FUA (force unit access) flag. You can accomplish this from user mode by invoking a data integrity operation like **fsync**, **fdatasync**, **sync**, or **umount**.

In either mode, some writes that were either unacknowledged or not followed by a flush might also be rebuilt.

#### Automatic and manual recovery

When a VDO volume enters **recovering** operating mode, VDO automatically rebuilds the unclean VDO volume after it comes back online. This is called *online recovery*.

If VDO cannot recover a VDO volume successfully, it places the volume in **read-only** operating mode that persists across volume restarts. You need to fix the problem manually by forcing a rebuild.

#### Additional resources

- For more information on automatic and manual recovery and VDO operating modes, see [Section 72.5.3, "VDO operating modes"](#).

### 72.5.3. VDO operating modes

This section describes the modes that indicate whether a VDO volume is operating normally or is recovering from an error.

You can display the current operating mode of a VDO volume using the **vdostats --verbose device** command. See the *Operating mode* attribute in the output.

#### normal

This is the default operating mode. VDO volumes are always in **normal** mode, unless either of the following states forces a different mode. A newly created VDO volume starts in **normal** mode.

#### recovering

When a VDO volume does not save all of its metadata before shutting down, it automatically enters **recovering** mode the next time that it starts up. The typical reasons for entering this mode are sudden power loss or a problem from the underlying storage device.

In **recovering** mode, VDO is fixing the references counts for each physical block of data on the device. Recovery usually does not take very long. The time depends on how large the VDO volume is, how fast the underlying storage device is, and how many other requests VDO is handling simultaneously. The VDO volume functions normally with the following exceptions:

- Initially, the amount of space available for write requests on the volume might be limited. As more of the metadata is recovered, more free space becomes available.
- Data written while the VDO volume is recovering might fail to deduplicate against data written before the crash if that data is in a portion of the volume that has not yet been recovered. VDO can compress data while recovering the volume. You can still read or overwrite compressed blocks.
- During an online recovery, certain statistics are unavailable: for example, *blocks in use* and *blocks free*. These statistics become available when the rebuild is complete.
- Response times for reads and writes might be slower than usual due to the ongoing recovery work

You can safely shut down the VDO volume in **recovering** mode. If the recovery does not finish before shutting down, the device enters **recovering** mode again the next time that it starts up.

The VDO volume automatically exits **recovering** mode and moves to **normal** mode when it has fixed all the reference counts. No administrator action is necessary. For details, see [Section 72.5.4, "Recovering a VDO volume online"](#).

#### read-only

When a VDO volume encounters a fatal internal error, it enters **read-only** mode. Events that might cause **read-only** mode include metadata corruption or the backing storage device becoming read-only. This mode is an error state.

In **read-only** mode, data reads work normally but data writes always fail. The VDO volume stays in **read-only** mode until an administrator fixes the problem.

You can safely shut down a VDO volume in **read-only** mode. The mode usually persists after the VDO volume is restarted. In rare cases, the VDO volume is not able to record the **read-only** state to the backing storage device. In these cases, VDO attempts to do a recovery instead.

Once a volume is in read-only mode, there is no guarantee that data on the volume has not been lost or corrupted. In such cases, Red Hat recommends copying the data out of the read-only volume and possibly restoring the volume from backup.

If the risk of data corruption is acceptable, it is possible to force an offline rebuild of the VDO volume metadata so the volume can be brought back online and made available. The integrity of the rebuilt data cannot be guaranteed. For details, see [Section 72.5.5, “Forcing an offline rebuild of a VDO volume metadata”](#).

#### 72.5.4. Recovering a VDO volume online

This procedure performs an online recovery on a VDO volume to recover metadata after an unclean shutdown.

##### Procedure

1. If the VDO volume is not already started, start it:

```
vdo start --name=my-vdo
```

No additional steps are necessary. The recovery runs in the background.

2. If you rely on volume statistics like *blocks in use* and *blocks free*, wait until they are available.

#### 72.5.5. Forcing an offline rebuild of a VDO volume metadata

This procedure performs a forced offline rebuild of a VDO volume metadata to recover after an unclean shutdown.



##### WARNING

This procedure might cause data loss on the volume.

##### Prerequisites

- The VDO volume is started.

##### Procedure

1. Check if the volume is in read-only mode. See the *operating mode* attribute in the command output:

```
vdo status --name=my-vdo
```

If the volume is not in read-only mode, it is not necessary to force an offline rebuild. Perform an online recovery as described in [Section 72.5.4, “Recovering a VDO volume online”](#).

2. Stop the volume if it is running:



```
vdo stop --name=my-vdo
```

3. Restart the volume with the **--forceRebuild** option:

```
vdo start --name=my-vdo --forceRebuild
```

### 72.5.6. Removing an unsuccessfully created VDO volume

This procedure cleans up a VDO volume in an intermediate state. A volume is left in an intermediate state if a failure occurs when creating the volume. This might happen when, for example:

- The system crashes
- Power fails
- The administrator interrupts a running **vdo create** command

#### Procedure

- To clean up, remove the unsuccessfully created volume with the **--force** option:

```
vdo remove --force --name=my-vdo
```

The **--force** option is required because the administrator might have caused a conflict by changing the system configuration since the volume was unsuccessfully created.

Without the **--force** option, the **vdo remove** command fails with the following message:

```
[...]
A previous operation failed.
Recovery from the failure either failed or was interrupted.
Add '--force' to 'remove' to perform the following cleanup.
Steps to clean up VDO my-vdo:
umount -f /dev/mapper/my-vdo
udevadm settle
dmsetup remove my-vdo
vdo: ERROR - VDO volume my-vdo previous operation (create) is incomplete
```

## 72.6. OPTIMIZING THE UDS INDEX

You can configure certain settings of the UDS index to optimize it on your system.

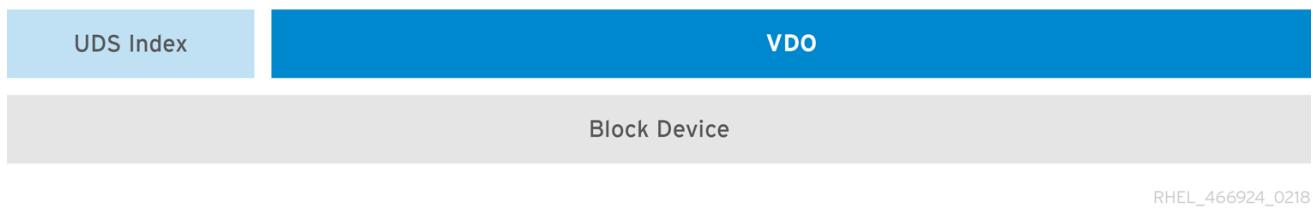


#### IMPORTANT

You cannot change the properties of the UDS index *after* creating the VDO volume.

### 72.6.1. Components of a VDO volume

VDO uses a block device as a backing store, which can include an aggregation of physical storage consisting of one or more disks, partitions, or even flat files. When a storage management tool creates a VDO volume, VDO reserves volume space for the UDS index and VDO volume. The UDS index and the VDO volume interact together to provide deduplicated block storage.

**Figure 72.2. VDO disk organization**

RHEL\_466924\_0218

The VDO solution consists of the following components:

### **kvdo**

A kernel module that loads into the Linux Device Mapper layer provides a deduplicated, compressed, and thinly provisioned block storage volume.

The **kvdo** module exposes a block device. You can access this block device directly for block storage or present it through a Linux file system, such as XFS or ext4.

When **kvdo** receives a request to read a logical block of data from a VDO volume, it maps the requested logical block to the underlying physical block and then reads and returns the requested data.

When **kvdo** receives a request to write a block of data to a VDO volume, it first checks whether the request is a DISCARD or TRIM request or whether the data is uniformly zero. If either of these conditions is true, **kvdo** updates its block map and acknowledges the request. Otherwise, VDO processes and optimizes the data.

### **uds**

A kernel module that communicates with the Universal Deduplication Service (UDS) index on the volume and analyzes data for duplicates. For each new piece of data, UDS quickly determines if that piece is identical to any previously stored piece of data. If the index finds a match, the storage system can then internally reference the existing item to avoid storing the same information more than once. The UDS index runs inside the kernel as the **uds** kernel module.

### Command line tools

For configuring and managing optimized storage.

## 72.6.2. The UDS index

VDO uses a high-performance deduplication index called UDS to detect duplicate blocks of data as they are being stored.

The UDS index provides the foundation of the VDO product. For each new piece of data, it quickly determines if that piece is identical to any previously stored piece of data. If the index finds match, the storage system can then internally reference the existing item to avoid storing the same information more than once.

The UDS index runs inside the kernel as the **uds** kernel module.

The *deduplication window* is the number of previously written blocks that the index remembers. The size of the deduplication window is configurable. For a given window size, the index requires a specific amount of RAM and a specific amount of disk space. The size of the window is usually determined by specifying the size of the index memory using the **--indexMem=***size* option. VDO then determines the amount of disk space to use automatically.

The UDS index consists of two parts:

- A compact representation is used in memory that contains at most one entry per unique block.
- An on-disk component that records the associated block names presented to the index as they occur, in order.

UDS uses an average of 4 bytes per entry in memory, including cache.

The on-disk component maintains a bounded history of data passed to UDS. UDS provides deduplication advice for data that falls within this deduplication window, containing the names of the most recently seen blocks. The deduplication window allows UDS to index data as efficiently as possible while limiting the amount of memory required to index large data repositories. Despite the bounded nature of the deduplication window, most datasets which have high levels of deduplication also exhibit a high degree of temporal locality – in other words, most deduplication occurs among sets of blocks that were written at about the same time. Furthermore, in general, data being written is more likely to duplicate data that was recently written than data that was written a long time ago. Therefore, for a given workload over a given time interval, deduplication rates will often be the same whether UDS indexes only the most recent data or all the data.

Because duplicate data tends to exhibit temporal locality, it is rarely necessary to index every block in the storage system. Were this not so, the cost of index memory would outstrip the savings of reduced storage costs from deduplication. Index size requirements are more closely related to the rate of data ingestion. For example, consider a storage system with 100 TB of total capacity but with an ingestion rate of 1 TB per week. With a deduplication window of 4 TB, UDS can detect most redundancy among the data written within the last month.

### 72.6.3. Recommended UDS index configuration

This section describes the recommended options to use with the UDS index, based on your intended use case.

In general, Red Hat recommends using a **sparse** UDS index for all production use cases. This is an extremely efficient indexing data structure, requiring approximately one-tenth of a byte of RAM per block in its deduplication window. On disk, it requires approximately 72 bytes of disk space per block. The minimum configuration of this index uses 256 MB of RAM and approximately 25 GB of space on disk.

To use this configuration, specify the **--sparseIndex=enabled --indexMem=0.25** options to the **vdo create** command. This configuration results in a deduplication window of 2.5 TB (meaning it will remember a history of 2.5 TB). For most use cases, a deduplication window of 2.5 TB is appropriate for deduplicating storage pools that are up to 10 TB in size.

The default configuration of the index, however, is to use a **dense** index. This index is considerably less efficient (by a factor of 10) in RAM, but it has much lower (also by a factor of 10) minimum required disk space, making it more convenient for evaluation in constrained environments.

In general, a deduplication window that is one quarter of the physical size of a VDO volume is a recommended configuration. However, this is not an actual requirement. Even small deduplication windows (compared to the amount of physical storage) can find significant amounts of duplicate data in many use cases. Larger windows may also be used, but it in most cases, there will be little additional benefit to doing so.

#### Additional resources

- Speak with your Red Hat Technical Account Manager representative for additional guidelines on tuning this important system parameter.

## 72.7. ENABLING OR DISABLING DEDUPLICATION IN VDO

In some instances, you might want to temporarily disable deduplication of data being written to a VDO volume while still retaining the ability to read to and write from the volume. Disabling deduplication prevents subsequent writes from being deduplicated, but the data that was already deduplicated remains so.

### 72.7.1. Deduplication in VDO

Deduplication is a technique for reducing the consumption of storage resources by eliminating multiple copies of duplicate blocks.

Instead of writing the same data more than once, VDO detects each duplicate block and records it as a reference to the original block. VDO maintains a mapping from logical block addresses, which are used by the storage layer above VDO, to physical block addresses, which are used by the storage layer under VDO.

After deduplication, multiple logical block addresses can be mapped to the same physical block address. These are called shared blocks. Block sharing is invisible to users of the storage, who read and write blocks as they would if VDO were not present.

When a shared block is overwritten, VDO allocates a new physical block for storing the new block data to ensure that other logical block addresses that are mapped to the shared physical block are not modified.

### 72.7.2. Enabling deduplication on a VDO volume

This procedure restarts the associated UDS index and informs the VDO volume that deduplication is active again.



#### NOTE

Deduplication is enabled by default.

#### Procedure

- To restart deduplication on a VDO volume, use the following command:

```
vdo enableDeduplication --name=my-vdo
```

### 72.7.3. Disabling deduplication on a VDO volume

This procedure stops the associated UDS index and informs the VDO volume that deduplication is no longer active.

#### Procedure

- To stop deduplication on a VDO volume, use the following command:

```
vdo disableDeduplication --name=my-vdo
```

- You can also disable deduplication when creating a new VDO volume by adding the **--deduplication=disabled** option to the **vdo create** command.

## 72.8. ENABLING OR DISABLING COMPRESSION IN VDO

VDO provides data compression. You can disable it to maximize performance or to speed processing of data that is unlikely to compress, or re-enable it to increase space savings.

### 72.8.1. Compression in VDO

In addition to block-level deduplication, VDO also provides inline block-level compression using the HIOPS Compression™ technology.

VDO volume compression is on by default.

While deduplication is the optimal solution for virtual machine environments and backup applications, compression works very well with structured and unstructured file formats that do not typically exhibit block-level redundancy, such as log files and databases.

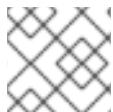
Compression operates on blocks that have not been identified as duplicates. When VDO sees unique data for the first time, it compresses the data. Subsequent copies of data that have already been stored are deduplicated without requiring an additional compression step.

The compression feature is based on a parallelized packaging algorithm that enables it to handle many compression operations at once. After first storing the block and responding to the requestor, a best-fit packing algorithm finds multiple blocks that, when compressed, can fit into a single physical block. After it is determined that a particular physical block is unlikely to hold additional compressed blocks, it is written to storage and the uncompressed blocks are freed and reused.

By performing the compression and packaging operations after having already responded to the requestor, using compression imposes a minimal latency penalty.

### 72.8.2. Enabling compression on a VDO volume

This procedure enables compression on a VDO volume to increase space savings.



#### NOTE

Compression is enabled by default.

#### Procedure

- To start it again, use the following command:

```
vdo enableCompression --name=my-vdo
```

### 72.8.3. Disabling compression on a VDO volume

This procedure stops compression on a VDO volume to maximize performance or to speed processing of data that is unlikely to compress.

#### Procedure

- To stop compression on an existing VDO volume, use the following command:

```
vdo disableCompression --name=my-vdo
```

- Alternatively, you can disable compression by adding the **--compression=disabled** option to the **vdo create** command when creating a new volume.

## 72.9. INCREASING THE SIZE OF A VDO VOLUME

You can increase the physical size of a VDO volume to utilize more underlying storage capacity, or the logical size to provide more capacity on the volume.

### 72.9.1. The physical and logical size of a VDO volume

This section describes the physical size, available physical size, and logical size that VDO can utilize:

#### Physical size

This is the same size as the underlying block device. VDO uses this storage for:

- User data, which might be deduplicated and compressed
- VDO metadata, such as the UDS index

#### Available physical size

This is the portion of the physical size that VDO is able to use for user data

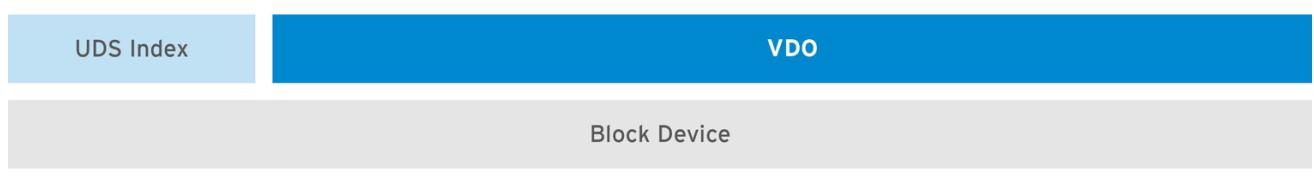
It is equivalent to the physical size minus the size of the metadata, minus the remainder after dividing the volume into slabs by the given slab size.

#### Logical Size

This is the provisioned size that the VDO volume presents to applications. It is usually larger than the available physical size. If the **--vdoLogicalSize** option is not specified, then the provisioning of the logical volume is now provisioned to a **1:1** ratio. For example, if a VDO volume is put on top of a 20 GB block device, then 2.5 GB is reserved for the UDS index (if the default index size is used). The remaining 17.5 GB is provided for the VDO metadata and user data. As a result, the available storage to consume is not more than 17.5 GB, and can be less due to metadata that makes up the actual VDO volume.

VDO currently supports any logical size up to 254 times the size of the physical volume with an absolute maximum logical size of 4PB.

Figure 72.3. VDO disk organization



RHEL\_466924\_0218

In this figure, the VDO deduplicated storage target sits completely on top of the block device, meaning the physical size of the VDO volume is the same size as the underlying block device.

#### Additional resources

- For more information on how much storage VDO metadata requires on block devices of different sizes, see [Section 71.6.4, "Examples of VDO requirements by physical volume size"](#).

## 72.9.2. Thin provisioning in VDO

VDO is a thinly provisioned block storage target. The amount of physical space that a VDO volume uses might differ from the size of the volume that is presented to users of the storage. You can make use of this disparity to save on storage costs.

### Out-of-space conditions

Take care to avoid unexpectedly running out of storage space, if the data written does not achieve the expected rate of optimization.

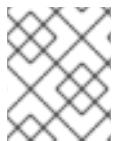
Whenever the number of logical blocks (virtual storage) exceeds the number of physical blocks (actual storage), it becomes possible for file systems and applications to unexpectedly run out of space. For that reason, storage systems using VDO must provide you with a way of monitoring the size of the free pool on the VDO volume.

You can determine the size of this free pool by using the **vdostats** utility. The default output of this utility lists information for all running VDO volumes in a format similar to the Linux **df** utility. For example:

Device	1K-blocks	Used	Available	Use%
/dev/mapper/vdo-name	211812352	105906176	105906176	50%

When the physical storage capacity of a VDO volume is almost full, VDO reports a warning in the system log, similar to the following:

```
Oct 2 17:13:39 system lvm[13863]: Monitoring VDO pool vdo-name.
Oct 2 17:27:39 system lvm[13863]: WARNING: VDO pool vdo-name is now 80.69% full.
Oct 2 17:28:19 system lvm[13863]: WARNING: VDO pool vdo-name is now 85.25% full.
Oct 2 17:29:39 system lvm[13863]: WARNING: VDO pool vdo-name is now 90.64% full.
Oct 2 17:30:29 system lvm[13863]: WARNING: VDO pool vdo-name is now 96.07% full.
```



### NOTE

These warning messages appear only when the **lvm2-monitor** service is running. It is enabled by default.

### How to prevent out-of-space conditions

If the size of free pool drops below a certain level, you can take action by:

- Deleting data. This reclaims space whenever the deleted data is not duplicated. Deleting data frees the space only after discards are issued.
- Adding physical storage



### IMPORTANT

Monitor physical space on your VDO volumes to prevent out-of-space situations. Running out of physical blocks might result in losing recently written, unacknowledged data on the VDO volume.

## Thin provisioning and the TRIM and DISCARD commands

To benefit from the storage savings of thin provisioning, the physical storage layer needs to know when data is deleted. File systems that work with thinly provisioned storage send **TRIM** or **DISCARD** commands to inform the storage system when a logical block is no longer required.

Several methods of sending the **TRIM** or **DISCARD** commands are available:

- With the **discard** mount option, the file systems can send these commands whenever a block is deleted.
- You can send the commands in a controlled manner by using utilities such as **fstrim**. These utilities tell the file system to detect which logical blocks are unused and send the information to the storage system in the form of a **TRIM** or **DISCARD** command.

The need to use **TRIM** or **DISCARD** on unused blocks is not unique to VDO. Any thinly provisioned storage system has the same challenge.

### 72.9.3. Increasing the logical size of a VDO volume

This procedure increases the logical size of a given VDO volume. It enables you to initially create VDO volumes that have a logical size small enough to be safe from running out of space. After some period of time, you can evaluate the actual rate of data reduction, and if sufficient, you can grow the logical size of the VDO volume to take advantage of the space savings.

It is not possible to decrease the logical size of a VDO volume.

#### Procedure

- To grow the logical size, use:

```
vdo growLogical --name=my-vdo \
--vdoLogicalSize=new-logical-size
```

When the logical size increases, VDO informs any devices or file systems on top of the volume of the new size.

### 72.9.4. Increasing the physical size of a VDO volume

This procedure increases the amount of physical storage available to a VDO volume.

It is not possible to shrink a VDO volume in this way.

#### Prerequisites

- The underlying block device has a larger capacity than the current physical size of the VDO volume.
- If it does not, you can attempt to increase the size of the device. The exact procedure depends on the type of the device. For example, to resize an MBR or GPT partition, see the [Resizing a partition](#) section in the *Managing storage devices* guide.

#### Procedure

- Add the new physical storage space to the VDO volume:

```
vdo growPhysical --name=my-vdo
```

## 72.10. REMOVING VDO VOLUMES

You can remove an existing VDO volume on your system.

### 72.10.1. Removing a working VDO volume

This procedure removes a VDO volume and its associated UDS index.

#### Procedure

1. Unmount the file systems and stop the applications that are using the storage on the VDO volume.
2. To remove the VDO volume from your system, use:

```
vdo remove --name=my-vdo
```

### 72.10.2. Removing an unsuccessfully created VDO volume

This procedure cleans up a VDO volume in an intermediate state. A volume is left in an intermediate state if a failure occurs when creating the volume. This might happen when, for example:

- The system crashes
- Power fails
- The administrator interrupts a running **vdo create** command

#### Procedure

- To clean up, remove the unsuccessfully created volume with the **--force** option:

```
vdo remove --force --name=my-vdo
```

The **--force** option is required because the administrator might have caused a conflict by changing the system configuration since the volume was unsuccessfully created.

Without the **--force** option, the **vdo remove** command fails with the following message:

[...]

A previous operation failed.

Recovery from the failure either failed or was interrupted.

Add '--force' to 'remove' to perform the following cleanup.

Steps to clean up VDO *my-vdo*:

umount -f /dev/mapper/*my-vdo*

udevadm settle

dmsetup remove *my-vdo*

vdo: ERROR - VDO volume *my-vdo* previous operation (create) is incomplete

## 72.11. RELATED INFORMATION

- You can use the **Ansible** tool to automate VDO deployment and administration. For details, see:
  - Ansible documentation: <https://docs.ansible.com/>
  - VDO Ansible module documentation:  
[https://docs.ansible.com/ansible/latest/modules/vdo\\_module.html](https://docs.ansible.com/ansible/latest/modules/vdo_module.html)

# CHAPTER 73. DISCARDING UNUSED BLOCKS

You can perform or schedule discard operations on block devices that support them.

## 73.1. BLOCK DISCARD OPERATIONS

Block discard operations discard blocks that are no longer in use by a mounted file system. They are useful on:

- Solid-state drives (SSDs)
- Thinly-provisioned storage

### Requirements

The block device underlying the file system must support physical discard operations.

Physical discard operations are supported if the value in the **/sys/block/device/queue/discard\_max\_bytes** file is not zero.

## 73.2. TYPES OF BLOCK DISCARD OPERATIONS

You can run discard operations using different methods:

### Batch discard

Are run explicitly by the user. They discard all unused blocks in the selected file systems.

### Online discard

Are specified at mount time. They run in real time without user intervention. Online discard operations discard only the blocks that are transitioning from used to free.

### Periodic discard

Are batch operations that are run regularly by a **systemd** service.

All types are supported by the XFS and ext4 file systems and by VDO.

### Recommendations

Red Hat recommends that you use batch or periodic discard.

Use online discard only if:

- the system's workload is such that batch discard is not feasible, or
- online discard operations are necessary to maintain performance.

## 73.3. PERFORMING BATCH BLOCK DISCARD

This procedure performs a batch block discard operation to discard unused blocks on a mounted file system.

### Prerequisites

- The file system is mounted.
- The block device underlying the file system supports physical discard operations.

## Procedure

- Use the **fstrim** utility:
  - To perform discard only on a selected file system, use:

```
fstrim mount-point
```
  - To perform discard on all mounted file systems, use:

```
fstrim --all
```

If you execute the **fstrim** command on:

- a device that does not support discard operations, or
- a logical device (LVM or MD) composed of multiple devices, where any one of the device does not support discard operations,

the following message displays:

```
fstrim /mnt/non_discard
fstrim: /mnt/non_discard: the discard operation is not supported
```

## Additional resources

- The **fstrim(8)** man page

## 73.4. ENABLING ONLINE BLOCK DISCARD

This procedure enables online block discard operations that automatically discard unused blocks on all supported file systems.

## Procedure

- Enable online discard at mount time:
  - When mounting a file system manually, add the **-o discard** mount option:

```
mount -o discard device mount-point
```
  - When mounting a file system persistently, add the **discard** option to the mount entry in the **/etc/fstab** file.

## Additional resources

- The **mount(8)** man page
- The **fstab(5)** man page

## 73.5. ENABLING ONLINE BLOCK DISCARD USING RHEL SYSTEM ROLES

This section describes how to enable online block discard using the **storage** role.

## Prerequisites

- An Ansible playbook including the **storage** role exists.

For information on how to apply such a playbook, see [Applying a role](#).

### 73.5.1. Example Ansible playbook to enable online block discard

This section provides an example Ansible playbook. This playbook applies the **storage** role to mount an XFS file system with online block discard enabled.

```

- hosts: all
 vars:
 storage_volumes:
 - name: barefs
 type: disk
 disks:
 - sdb
 fs_type: xfs
 mount_point: /mnt/data
 mount_options: discard
 roles:
 - rhel-system-roles.storage
```

## 73.6. ENABLING PERIODIC BLOCK DISCARD

This procedure enables a **systemd** timer that regularly discards unused blocks on all supported file systems.

### Procedure

- Enable and start the **systemd** timer:

```
systemctl enable --now fstrim.timer
```

# CHAPTER 74. USING THE WEB CONSOLE FOR MANAGING VIRTUAL DATA OPTIMIZER VOLUMES

This chapter describes the Virtual Data Optimizer (VDO) configuration using the RHEL 8 web console. After reading it, you will be able to:

- Create VDO volumes
- Format VDO volumes
- Extend VDO volumes

## Prerequisites

- The RHEL 8 web console is installed and accessible.  
For details, see [Installing the web console](#).

## 74.1. VDO VOLUMES IN THE WEB CONSOLE

Red Hat Enterprise Linux 8 supports Virtual Data Optimizer (VDO). VDO is a block virtualization technology that combines:

### Compression

For details, see [Enabling or disabling compression in VDO](#).

### Deduplication

For details, see [Enabling or disabling deduplication in VDO](#).

### Thin provisioning

For details, see [Thinly-provisioned logical volumes \(thin volumes\)](#).

Using these technologies, VDO:

- Saves storage space inline
- Compresses files
- Eliminates duplications
- Enables you to allocate more virtual space than how much the physical or logical storage provides
- Enables you to extend the virtual storage by growing

VDO can be created on top of many types of storage. In the RHEL 8 web console, you can configure VDO on top of:

- LVM



### NOTE

It is not possible to configure VDO on top of thinly-provisioned volumes.

- Physical volume

- Software RAID

For details about placement of VDO in the Storage Stack, see [System Requirements](#).

#### Additional resources

- For details about VDO, see [Deduplicating and compressing storage](#).

## 74.2. CREATING VDO VOLUMES IN THE WEB CONSOLE

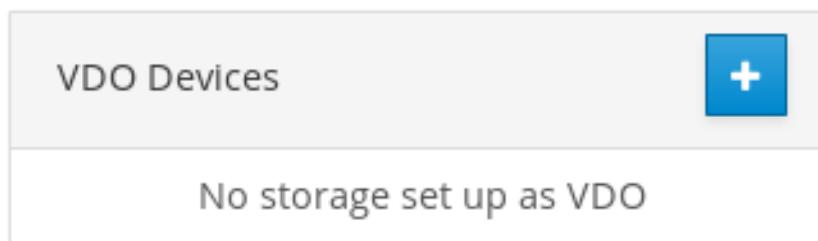
This section helps you to create a VDO volume in the RHEL web console.

#### Prerequisites

- Physical drives, LVMs, or RAID from which you want to create VDO.

#### Procedure

1. Log in to the RHEL 8 web console.  
For details, see [Logging in to the web console](#).
2. Click **Storage**.
3. Click the + icon in the **VDO Devices** box.



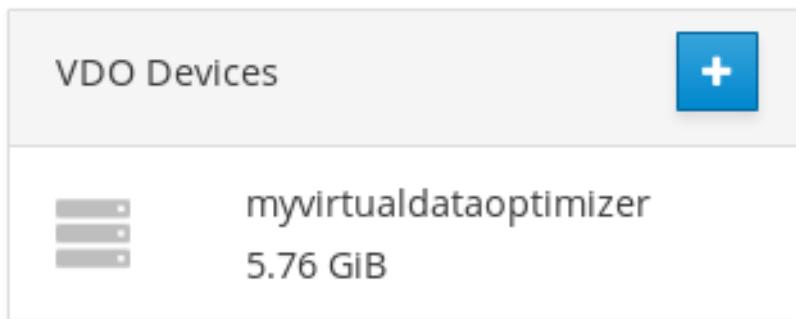
4. In the **Name** field, enter a name of a VDO volume without spaces.
5. Select the drive that you want to use.
6. In the **Logical Size** bar, set up the size of the VDO volume. You can extend it more than ten times, but consider for what purpose you are creating the VDO volume:
  - For active VMs or container storage, use logical size that is ten times the physical size of the volume.
  - For object storage, use logical size that is three times the physical size of the volume.
 For details, see [Deploying VDO](#).
7. In the **Index Memory** bar, allocate memory for the VDO volume.  
For details about VDO system requirements, see [System Requirements](#).
8. Select the **Compression** option. This option can efficiently reduce various file formats.  
For details, see [Enabling or disabling compression in VDO](#).
9. Select the **Deduplication** option.  
This option reduces the consumption of storage resources by eliminating multiple copies of duplicate blocks. For details, see [Enabling or disabling deduplication in VDO](#).

- [Optional] If you want to use the VDO volume with applications that need a 512 bytes block size, select **Use 512 Byte emulation**. This reduces the performance of the VDO volume, but should be very rarely needed. If in doubt, leave it off.
- Click **Create**.

**Create VDO Device**

Name	myvirtualdataoptimizer
Disk	<input checked="" type="radio"/> 5.72 GiB RAID Device 127 /dev/md/127
Logical Size	5.76 GiB
Index Memory	256 MiB
Options	<input checked="" type="checkbox"/> Compression <input checked="" type="checkbox"/> Deduplication <input type="checkbox"/> Use 512 Byte emulation
<input type="button" value="Cancel"/> <input type="button" value="Create"/>	

If the process of creating the VDO volume succeeds, you can see the new VDO volume in the **Storage** section and format it with a file system.



### 74.3. FORMATTING VDO VOLUMES IN THE WEB CONSOLE

VDO volumes act as physical drives. To use them, you need to format them with a file system.



#### WARNING

Formatting VDO will erase all data on the volume.

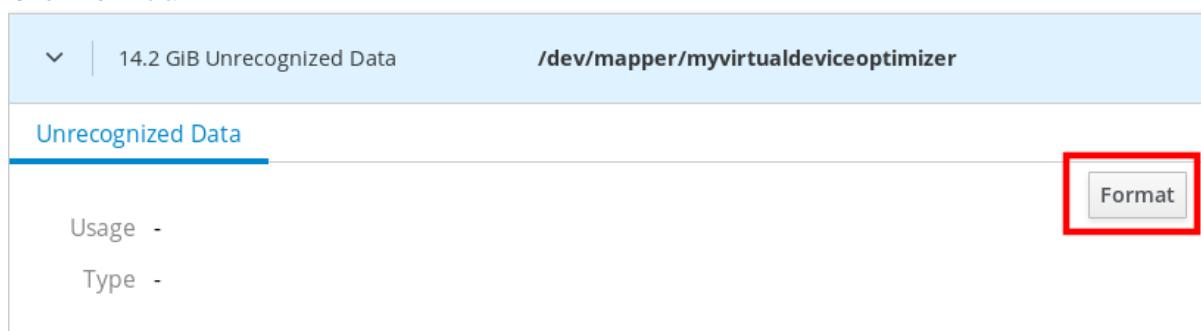
The following steps describe the procedure to format VDO volumes.

## Prerequisites

- A VDO volume is created. For details, see [Section 74.2, “Creating VDO volumes in the web console”](#).

## Procedure

1. Log in to the RHEL 8 web console.  
For details, see [Logging in to the web console](#).
2. Click **Storage**.
3. Click the VDO volume.
4. Click on the **Unrecognized Data** tab.
5. Click **Format**.



6. In the **Erase** drop down menu, select:

### Don't overwrite existing data

The RHEL web console rewrites only the disk header. The advantage of this option is the speed of formatting.

### Overwrite existing data with zeros

The RHEL web console rewrites the whole disk with zeros. This option is slower because the program has to go through the whole disk. Use this option if the disk includes any data and you need to rewrite them.

7. In the **Type** drop down menu, select a filesystem:

- The **XFS** file system supports large logical volumes, switching physical drives online without outage, and growing. Leave this file system selected if you do not have a different strong preference.  
XFS does not support shrinking volumes. Therefore, you will not be able to reduce volume formatted with XFS.
- The **ext4** file system supports logical volumes, switching physical drives online without outage, growing, and shrinking.

You can also select a version with the LUKS (Linux Unified Key Setup) encryption, which allows you to encrypt the volume with a passphrase.

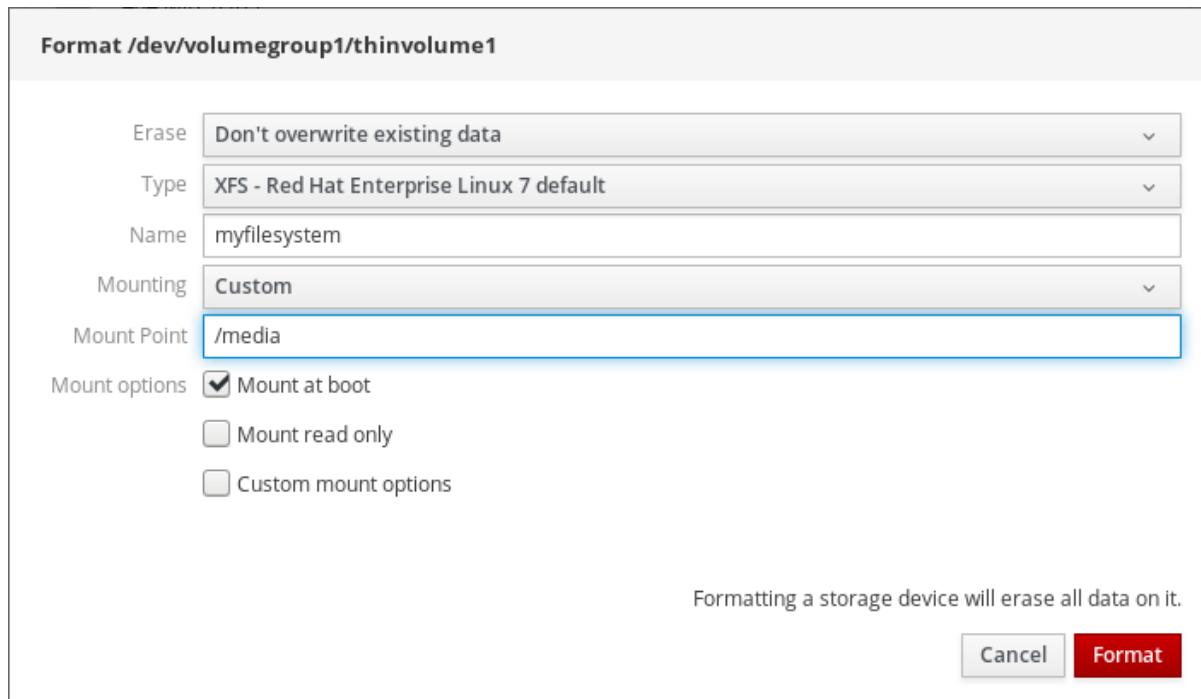
8. In the **Name** field, enter the logical volume name.

9. In the **Mounting** drop down menu, select **Custom**.

The **Default** option does not ensure that the file system will be mounted on the next boot.

10. In the **Mount Point** field, add the mount path.

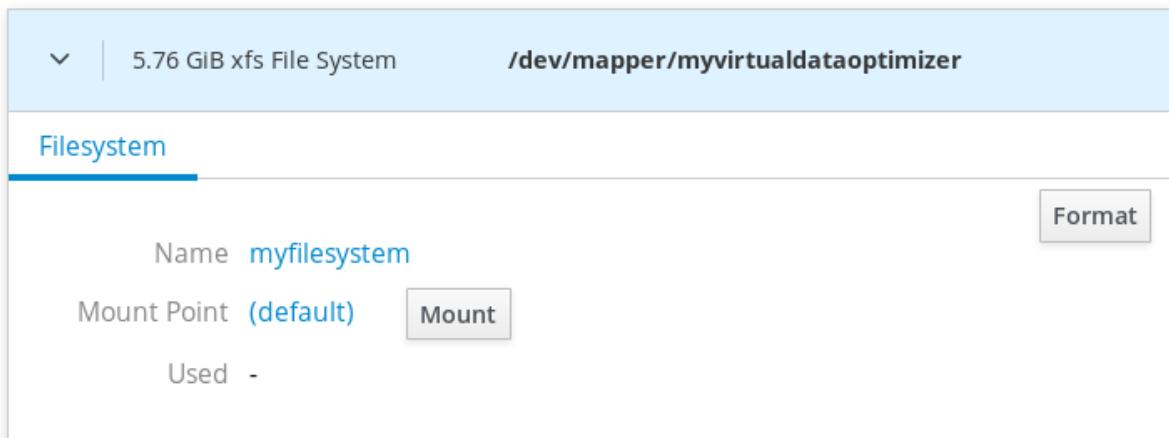
11. Select **Mount at boot**.



12. Click **Format**.

Formatting can take several minutes depending on the used formatting options and the volume size.

After a successful finish, you can see the details of the formatted VDO volume on the **Filesystem** tab.



13. To use the VDO volume, click **Mount**.

At this point, the system uses the mounted and formatted VDO volume.

## 74.4. EXTENDING VDO VOLUMES IN THE WEB CONSOLE

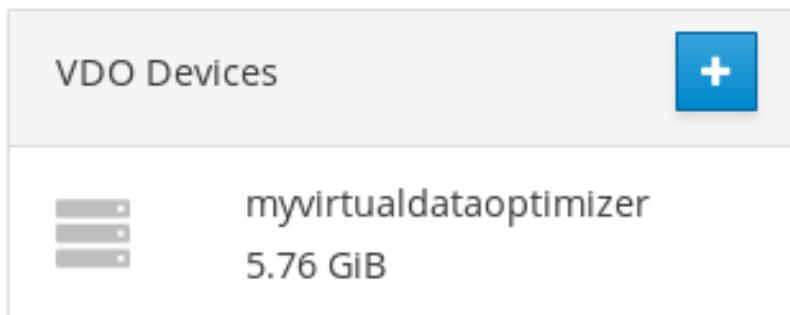
This section describes extending VDO volumes in the RHEL 8 web console.

### Prerequisites

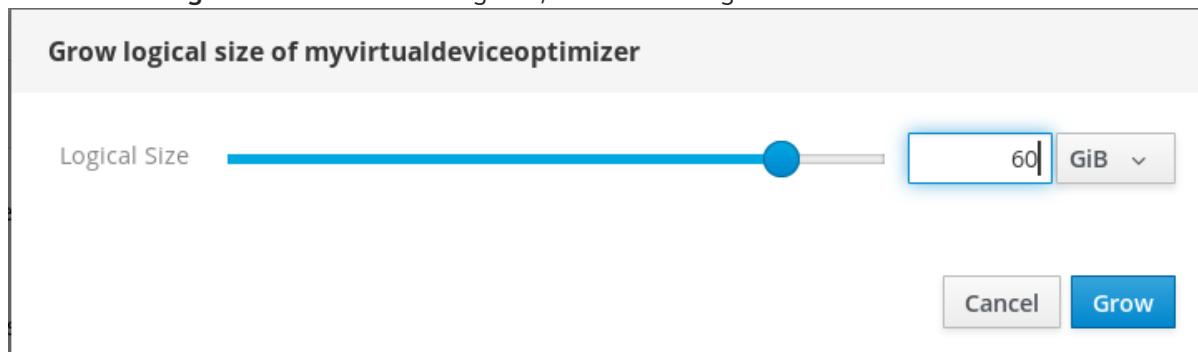
- The VDO volume created.

## Procedure

1. Log in to the RHEL 8 web console.  
For details, see [Logging in to the web console](#).
2. Click **Storage**.
3. Click your VDO volume in the **VDO Devices** box.



4. In the VDO volume details, click the **Grow** button.
5. In the **Grow logical size of VDO** dialog box, extend the logical size of the VDO volume.



Original size of the logical volume from the screenshot was 6 GB. As you can see, the RHEL web console enables you to grow the volume to more than ten times the size and it works correctly because of the compression and deduplication.

6. Click **Grow**.

If the process of growing VDO succeeds, you can see the new size in the VDO volume details.

VDO Device myvirtualdataoptimizer

Stop Delete

Device File `/dev/mapper/myvirtualdataoptimizer`

Backing Device `/dev/md/127`

Physical 1.11 MiB data + 3.72 GiB overhead used of 5.72 GiB (65%)

Logical 11.7 MiB used of 60 GiB (90% saved)   Grow

Index Memory 256 MiB

Compression ON  

Deduplication ON

## PART V. DESIGN OF LOG FILE

# CHAPTER 75. AUDITING THE SYSTEM

Audit does not provide additional security to your system; rather, it can be used to discover violations of security policies used on your system. These violations can further be prevented by additional security measures such as SELinux.

## 75.1. LINUX AUDIT

The Linux Audit system provides a way to track security-relevant information on your system. Based on pre-configured rules, Audit generates log entries to record as much information about the events that are happening on your system as possible. This information is crucial for mission-critical environments to determine the violator of the security policy and the actions they performed.

The following list summarizes some of the information that Audit is capable of recording in its log files:

- Date and time, type, and outcome of an event.
- Sensitivity labels of subjects and objects.
- Association of an event with the identity of the user who triggered the event.
- All modifications to Audit configuration and attempts to access Audit log files.
- All uses of authentication mechanisms, such as SSH, Kerberos, and others.
- Changes to any trusted database, such as **/etc/passwd**.
- Attempts to import or export information into or from the system.
- Include or exclude events based on user identity, subject and object labels, and other attributes.

The use of the Audit system is also a requirement for a number of security-related certifications. Audit is designed to meet or exceed the requirements of the following certifications or compliance guides:

- Controlled Access Protection Profile (CAPP)
- Labeled Security Protection Profile (LSPP)
- Rule Set Base Access Control (RSBAC)
- National Industrial Security Program Operating Manual (NISPOM)
- Federal Information Security Management Act (FISMA)
- Payment Card Industry – Data Security Standard (PCI-DSS)
- Security Technical Implementation Guides (STIG)

Audit has also been:

- Evaluated by National Information Assurance Partnership (NIAP) and Best Security Industries (BSI).
- Certified to LSPP/CAPP/RSBAC/EAL4+ on Red Hat Enterprise Linux 5.
- Certified to Operating System Protection Profile / Evaluation Assurance Level 4+ (OSPP/EAL4+) on Red Hat Enterprise Linux 6.

## Use Cases

### Watching file access

Audit can track whether a file or a directory has been accessed, modified, executed, or the file's attributes have been changed. This is useful, for example, to detect access to important files and have an Audit trail available in case one of these files is corrupted.

### Monitoring system calls

Audit can be configured to generate a log entry every time a particular system call is used. This can be used, for example, to track changes to the system time by monitoring the **settimeofday**, **clock\_adjtime**, and other time-related system calls.

### Recording commands run by a user

Audit can track whether a file has been executed, so rules can be defined to record every execution of a particular command. For example, a rule can be defined for every executable in the **/bin** directory. The resulting log entries can then be searched by user ID to generate an audit trail of executed commands per user.

### Recording execution of system pathnames

Aside from watching file access which translates a path to an inode at rule invocation, Audit can now watch the execution of a path even if it does not exist at rule invocation, or if the file is replaced after rule invocation. This allows rules to continue to work after upgrading a program executable or before it is even installed.

### Recording security events

The **pam\_faillock** authentication module is capable of recording failed login attempts. Audit can be set up to record failed login attempts as well and provides additional information about the user who attempted to log in.

### Searching for events

Audit provides the **ausearch** utility, which can be used to filter the log entries and provide a complete audit trail based on several conditions.

### Running summary reports

The **aureport** utility can be used to generate, among other things, daily reports of recorded events. A system administrator can then analyze these reports and investigate suspicious activity further.

### Monitoring network access

The **iptables** and **ebtables** utilities can be configured to trigger Audit events, allowing system administrators to monitor network access.



#### NOTE

System performance may be affected depending on the amount of information that is collected by Audit.

## 75.2. AUDIT SYSTEM ARCHITECTURE

The Audit system consists of two main parts: the user-space applications and utilities, and the kernel-side system call processing. The kernel component receives system calls from user-space applications and filters them through one of the following filters: **user**, **task**, **fstype**, or **exit**.

Once a system call passes the **exclude** filter, it is sent through one of the aforementioned filters, which, based on the Audit rule configuration, sends it to the Audit daemon for further processing.

The user-space Audit daemon collects the information from the kernel and creates entries in a log file. Other Audit user-space utilities interact with the Audit daemon, the kernel Audit component, or the Audit log files:

- **auditctl** – the Audit control utility interacts with the kernel Audit component to manage rules and to control many settings and parameters of the event generation process.
- The remaining Audit utilities take the contents of the Audit log files as input and generate output based on user's requirements. For example, the **aureport** utility generates a report of all recorded events.

In RHEL 8, the Audit dispatcher daemon (**audisp**) functionality is integrated in the Audit daemon (**auditd**). Configuration files of plugins for the interaction of real-time analytical programs with Audit events are located in the **/etc/audit/plugins.d** directory by default.

## 75.3. CONFIGURING AUDITD FOR A SECURE ENVIRONMENT

The default **auditd** configuration should be suitable for most environments. However, if your environment has to meet strict security policies, the following settings are suggested for the Audit daemon configuration in the **/etc/audit/auditd.conf** file:

### **log\_file**

The directory that holds the Audit log files (usually **/var/log/audit/**) should reside on a separate mount point. This prevents other processes from consuming space in this directory and provides accurate detection of the remaining space for the Audit daemon.

### **max\_log\_file**

Specifies the maximum size of a single Audit log file, must be set to make full use of the available space on the partition that holds the Audit log files.

### **max\_log\_file\_action**

Decides what action is taken once the limit set in **max\_log\_file** is reached, should be set to **keep\_logs** to prevent Audit log files from being overwritten.

### **space\_left**

Specifies the amount of free space left on the disk for which an action that is set in the **space\_left\_action** parameter is triggered. Must be set to a number that gives the administrator enough time to respond and free up disk space. The **space\_left** value depends on the rate at which the Audit log files are generated.

### **space\_left\_action**

It is recommended to set the **space\_left\_action** parameter to **email** or **exec** with an appropriate notification method.

### **admin\_space\_left**

Specifies the absolute minimum amount of free space for which an action that is set in the **admin\_space\_left\_action** parameter is triggered, must be set to a value that leaves enough space to log actions performed by the administrator.

### **admin\_space\_left\_action**

Should be set to **single** to put the system into single-user mode and allow the administrator to free up some disk space.

### **disk\_full\_action**

Specifies an action that is triggered when no free space is available on the partition that holds the Audit log files, must be set to **halt** or **single**. This ensures that the system is either shut down or operating in single-user mode when Audit can no longer log events.

**disk\_error\_action**

Specifies an action that is triggered in case an error is detected on the partition that holds the Audit log files, must be set to **syslog**, **single**, or **halt**, depending on your local security policies regarding the handling of hardware malfunctions.

**flush**

Should be set to **incremental\_async**. It works in combination with the **freq** parameter, which determines how many records can be sent to the disk before forcing a hard synchronization with the hard drive. The **freq** parameter should be set to **100**. These parameters assure that Audit event data is synchronized with the log files on the disk while keeping good performance for bursts of activity.

The remaining configuration options should be set according to your local security policy.

## 75.4. STARTING AND CONTROLLING AUDITD

Once **auditd** is configured, start the service to collect Audit information and store it in the log files. Use the following command as the root user to start **auditd**:

```
~]# service auditd start
```

To configure **auditd** to start at boot time:

```
~]# systemctl enable auditd
```

A number of other actions can be performed on **auditd** using the **service auditd action** command, where *action* can be one of the following:

**stop**

Stops **auditd**.

**restart**

Restarts **auditd**.

**reload or force-reload**

Reloads the configuration of **auditd** from the **/etc/audit/auditd.conf** file.

**rotate**

Rotates the log files in the **/var/log/audit/** directory.

**resume**

Resumes logging of Audit events after it has been previously suspended, for example, when there is not enough free space on the disk partition that holds the Audit log files.

**condrestart or try-restart**

Restarts **auditd** only if it is already running.

**status**

Displays the running status of **auditd**.

**NOTE**

The **service** command is the only way to correctly interact with the **auditd** daemon. You need to use the **service** command so that the **audit** value is properly recorded. You can use the **systemctl** command only for two actions: **enable** and **status**.

## 75.5. UNDERSTANDING AUDIT LOG FILES

By default, the Audit system stores log entries in the `/var/log/audit/audit.log` file; if log rotation is enabled, rotated `audit.log` files are stored in the same directory.

Add the following Audit rule to log every attempt to read or modify the `/etc/ssh/sshd_config` file:

```
auditctl -w /etc/ssh/sshd_config -p warx -k sshd_config
```

If the **auditd** daemon is running, for example, using the following command creates a new event in the Audit log file:

```
$ cat /etc/ssh/sshd_config
```

This event in the **audit.log** file looks as follows:

```
type=SYSCALL msg=audit(1364481363.243:24287): arch=c000003e syscall=2 success=no exit=-13
a0=7ffd19c5592 a1=0 a2=7ffd19c4b50 a3=a items=1 ppid=2686 pid=3538 auid=1000 uid=1000
gid=1000 euid=1000 suid=1000 egid=1000 sgid=1000 fsgid=1000 tty=pts0 ses=1
comm="cat" exe="/bin/cat" subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
key="sshd_config"
type=CWD msg=audit(1364481363.243:24287): cwd="/home/shadowman"
type=PATH msg=audit(1364481363.243:24287): item=0 name="/etc/ssh/sshd_config" inode=409248
dev=fd:00 mode=0100600 ouid=0 ogid=0 rdev=00:00 obj=system_u:object_r:etc_t:s0
nametype=NORMAL cap_fp=none cap_fi=none cap_fe=0 cap_fver=0
type=PROCTITLE msg=audit(1364481363.243:24287) :
proctitle=636174002F6574632F7373682F737368645F636F6E666967
```

The above event consists of four records, which share the same time stamp and serial number. Records always start with the **type=** keyword. Each record consists of several **name=value** pairs separated by a white space or a comma. A detailed analysis of the above event follows:

### First Record

#### **type=SYSCALL**

The **type** field contains the type of the record. In this example, the **SYSCALL** value specifies that this record was triggered by a system call to the kernel.

#### **msg=audit(1364481363.243:24287):**

The **msg** field records:

- a time stamp and a unique ID of the record in the form **audit(*time\_stamp*:*ID*)**. Multiple records can share the same time stamp and ID if they were generated as part of the same Audit event. The time stamp is using the Unix time format - seconds since 00:00:00 UTC on 1 January 1970.
- various event-specific **name=value** pairs provided by the kernel or user-space applications.

#### **arch=c000003e**

The **arch** field contains information about the CPU architecture of the system. The value, **c000003e**, is encoded in hexadecimal notation. When searching Audit records with the **ausearch** command, use the **-i** or **--interpret** option to automatically convert hexadecimal values into their human-readable equivalents. The **c000003e** value is interpreted as **x86\_64**.

**syscall=2**

The **syscall** field records the type of the system call that was sent to the kernel. The value, **2**, can be matched with its human-readable equivalent in the **/usr/include/asm/unistd\_64.h** file. In this case, **2** is the **open** system call. Note that the **ausyscall** utility allows you to convert system call numbers to their human-readable equivalents. Use the **ausyscall --dump** command to display a listing of all system calls along with their numbers. For more information, see the **ausyscall(8)** man page.

**success=no**

The **success** field records whether the system call recorded in that particular event succeeded or failed. In this case, the call did not succeed.

**exit=-13**

The **exit** field contains a value that specifies the exit code returned by the system call. This value varies for a different system call. You can interpret the value to its human-readable equivalent with the following command:

```
~]# ausearch --interpret --exit -13
```

Note that the previous example assumes that your Audit log contains an event that failed with exit code **-13**.

**a0=7ffd19c5592, a1=0, a2=7ffd19c5592, a3=a**

The **a0** to **a3** fields record the first four arguments, encoded in hexadecimal notation, of the system call in this event. These arguments depend on the system call that is used; they can be interpreted by the **ausearch** utility.

**items=1**

The **items** field contains the number of PATH auxiliary records that follow the syscall record.

**ppid=2686**

The **ppid** field records the Parent Process ID (PPID). In this case, **2686** was the PPID of the parent process such as **bash**.

**pid=3538**

The **pid** field records the Process ID (PID). In this case, **3538** was the PID of the **cat** process.

**auid=1000**

The **auid** field records the Audit user ID, that is the loginuid. This ID is assigned to a user upon login and is inherited by every process even when the user's identity changes, for example, by switching user accounts with the **su - john** command.

**uid=1000**

The **uid** field records the user ID of the user who started the analyzed process. The user ID can be interpreted into user names with the following command: **ausearch -i --uid *UID***.

**gid=1000**

The **gid** field records the group ID of the user who started the analyzed process.

**euid=1000**

The **euid** field records the effective user ID of the user who started the analyzed process.

**suid=1000**

The **suid** field records the set user ID of the user who started the analyzed process.

**fsuid=1000**

The **fsuid** field records the file system user ID of the user who started the analyzed process.

**egid=1000**

The **egid** field records the effective group ID of the user who started the analyzed process.

#### **sgid=1000**

The **sgid** field records the set group ID of the user who started the analyzed process.

#### **fsgid=1000**

The **fsgid** field records the file system group ID of the user who started the analyzed process.

#### **tty=pts0**

The **tty** field records the terminal from which the analyzed process was invoked.

#### **ses=1**

The **ses** field records the session ID of the session from which the analyzed process was invoked.

#### **comm="cat"**

The **comm** field records the command-line name of the command that was used to invoke the analyzed process. In this case, the **cat** command was used to trigger this Audit event.

#### **exe="/bin/cat"**

The **exe** field records the path to the executable that was used to invoke the analyzed process.

#### **subj=unconfined\_u:unconfined\_r:unconfined\_t:s0-s0:c0.c1023**

The **subj** field records the SELinux context with which the analyzed process was labeled at the time of execution.

#### **key="sshd\_config"**

The **key** field records the administrator-defined string associated with the rule that generated this event in the Audit log.

### Second Record

#### **type=CWD**

In the second record, the **type** field value is **CWD** – current working directory. This type is used to record the working directory from which the process that invoked the system call specified in the first record was executed.

The purpose of this record is to record the current process's location in case a relative path winds up being captured in the associated **PATH** record. This way the absolute path can be reconstructed.

#### **msg=audit(1364481363.243:24287)**

The **msg** field holds the same time stamp and ID value as the value in the first record. The time stamp is using the Unix time format – seconds since 00:00:00 UTC on 1 January 1970.

#### **cwd="/home/user\_name"**

The **cwd** field contains the path to the directory in which the system call was invoked.

### Third Record

#### **type=PATH**

In the third record, the **type** field value is **PATH**. An Audit event contains a **PATH**-type record for every path that is passed to the system call as an argument. In this Audit event, only one path (**/etc/ssh/sshd\_config**) was used as an argument.

#### **msg=audit(1364481363.243:24287):**

The **msg** field holds the same time stamp and ID value as the value in the first and second record.

#### **item=0**

The **item** field indicates which item, of the total number of items referenced in the **SYSCALL** type record, the current record is. This number is zero-based; a value of **0** means it is the first item.

#### **name="/etc/ssh/sshd\_config"**

The **name** field records the path of the file or directory that was passed to the system call as an argument. In this case, it was the **/etc/ssh/sshd\_config** file.

#### **inode=409248**

The **inode** field contains the inode number associated with the file or directory recorded in this event. The following command displays the file or directory that is associated with the **409248** inode number:

```
~]# find / -inum 409248 -print
/etc/ssh/sshd_config
```

#### **dev=fd:00**

The **dev** field specifies the minor and major ID of the device that contains the file or directory recorded in this event. In this case, the value represents the **/dev/fd/0** device.

#### **mode=0100600**

The **mode** field records the file or directory permissions, encoded in numerical notation as returned by the **stat** command in the **st\_mode** field. See the **stat(2)** man page for more information. In this case, **0100600** can be interpreted as **-rw-----**, meaning that only the root user has read and write permissions to the **/etc/ssh/sshd\_config** file.

#### **ouid=0**

The **ouid** field records the object owner's user ID.

#### **ogid=0**

The **ogid** field records the object owner's group ID.

#### **rdev=00:00**

The **rdev** field contains a recorded device identifier for special files only. In this case, it is not used as the recorded file is a regular file.

#### **obj=system\_u:object\_r:etc\_t:s0**

The **obj** field records the SELinux context with which the recorded file or directory was labeled at the time of execution.

#### **nametype=NORMAL**

The **nametype** field records the intent of each path record's operation in the context of a given syscall.

#### **cap\_fp=none**

The **cap\_fp** field records data related to the setting of a permitted file system-based capability of the file or directory object.

#### **cap\_hi=none**

The **cap\_hi** field records data related to the setting of an inherited file system-based capability of the file or directory object.

#### **cap\_fe=0**

The **cap\_fe** field records the setting of the effective bit of the file system-based capability of the file or directory object.

#### **cap\_fver=0**

The **cap\_fver** field records the version of the file system-based capability of the file or directory object.

## Fourth Record

### type=PROCTITLE

The **type** field contains the type of the record. In this example, the **PROCTITLE** value specifies that this record gives the full command-line that triggered this Audit event, triggered by a system call to the kernel.

**proctitle=636174002F6574632F7373682F737368645F636F6E666967**

The **proctitle** field records the full command-line of the command that was used to invoke the analyzed process. The field is encoded in hexadecimal notation to not allow the user to influence the Audit log parser. The text decodes to the command that triggered this Audit event. When searching Audit records with the **ausearch** command, use the **-i** or **--interpret** option to automatically convert hexadecimal values into their human-readable equivalents. The **636174002F6574632F7373682F737368645F636F6E666967** value is interpreted as **cat /etc/ssh/sshd\_config**.

## 75.6. USING AUDITCTL FOR DEFINING AND EXECUTING AUDIT RULES

The Audit system operates on a set of rules that define what is captured in the log files. Audit rules can be set either on the command line using the **auditctl** utility or in the **/etc/audit/rules.d** directory.

The **auditctl** command enables you to control the basic functionality of the Audit system and to define rules that decide which Audit events are logged.

### File-system rules examples

1. To define a rule that logs all write access to, and every attribute change of, the **/etc/passwd** file:

```
auditctl -w /etc/passwd -p wa -k passwd_changes
```

2. To define a rule that logs all write access to, and every attribute change of, all the files in the **/etc/selinux/** directory:

```
auditctl -w /etc/selinux/ -p wa -k selinux_changes
```

### System-call rules examples

1. To define a rule that creates a log entry every time the **adjtimex** or **settimeofday** system calls are used by a program, and the system uses the 64-bit architecture:

```
auditctl -a always,exit -F arch=b64 -S adjtimex -S settimeofday -k time_change
```

2. To define a rule that creates a log entry every time a file is deleted or renamed by a system user whose ID is 1000 or larger:

```
auditctl -a always,exit -S unlink -S unlinkat -S rename -S renameat -F auid>=1000 -F auid!=4294967295 -k delete
```

Note that the **-F auid!=4294967295** option is used to exclude users whose login UID is not set.

### Executable-file rules

To define a rule that logs all execution of the **/bin/id** program, execute the following command:

```
auditctl -a always,exit -F exe=/bin/id -F arch=b64 -S execve -k execution_bin_id
```

## Additional resources

See the **auditctl(8)** man page for more information, performance tips, and additional examples of use.

## 75.7. DEFINING PERSISTENT AUDIT RULES

To define Audit rules that are persistent across reboots, you must either directly include them in the **/etc/audit/rules.d/audit.rules** file or use the **augenrules** program that reads rules located in the **/etc/audit/rules.d/** directory.

Note that the **/etc/audit/audit.rules** file is generated whenever the **auditd** service starts. Files in **/etc/audit/rules.d/** use the same **auditctl** command-line syntax to specify the rules. Empty lines and text following a hash sign (#) are ignored.

The **auditctl** command can also be used to read rules from a specified file using the -R option, for example:

```
auditctl -R /usr/share/doc/audit/rules/30-stig.rules
```

## 75.8. USING PRE-CONFIGURED RULES FILES

In the **/usr/share/doc/audit/rules/** directory, the **audit** package provides a set of pre-configured rules files according to various certification standards:

### 30-nispom.rules

Audit rule configuration that meets the requirements specified in the Information System Security chapter of the National Industrial Security Program Operating Manual.

### 30-ospp-v42.rules

Audit rule configuration that meets the requirements defined in the OSPP (Protection Profile for General Purpose Operating Systems) profile version 4.2.

### 30-pci-dss-v31.rules

Audit rule configuration that meets the requirements set by Payment Card Industry Data Security Standard (PCI DSS) v3.1.

### 30-stig.rules

Audit rule configuration that meets the requirements set by Security Technical Implementation Guides (STIG).

To use these configuration files, copy them to the **/etc/audit/rules.d/** directory and use the **augenrules --load** command, for example:

```
cp /usr/share/doc/audit/rules/10-base-config.rules /usr/share/doc/audit/rules/30-stig.rules
/usr/share/doc/audit/rules/31-privileged.rules /usr/share/doc/audit/rules/99-finalize.rules
/etc/audit/rules.d/
augenrules --load
```

## Additional resources

The Audit rules have a numbering scheme that allows them to be ordered. To learn more about the naming scheme, see the **/usr/share/doc/audit/rules/README-rules** file.

See the **audit.rules(7)** man page for more information, troubleshooting, and additional examples of use.

## 75.9. USING AUGENRULES TO DEFINE PERSISTENT RULES

The **augenrules** script reads rules located in the **/etc/audit/rules.d/** directory and compiles them into an **audit.rules** file. This script processes all files that end with **.rules** in a specific order based on their natural sort order. The files in this directory are organized into groups with the following meanings:

- 10 – Kernel and auditctl configuration
- 20 – Rules that could match general rules but you want a different match
- 30 – Main rules
- 40 – Optional rules
- 50 – Server-specific rules
- 70 – System local rules
- 90 – Finalize (immutable)

The rules are not meant to be used all at once. They are pieces of a policy that should be thought out and individual files copied to **/etc/audit/rules.d/**. For example, to set a system up in the STIG configuration, copy rules **10-base-config**, **30-stig**, **31-privileged**, and **99-finalize**.

Once you have the rules in the **/etc/audit/rules.d/** directory, load them by running the **augenrules** script with the **--load** directive:

```
augenrules --load
/sbin/augenrules: No change
No rules
enabled 1
failure 1
pid 742
rate_limit 0
[trimmed for clarity]
```

### Additional resources

For more information on the Audit rules and the **augenrules** script, see the **audit.rules(8)** and **augenrules(8)** man pages.

## 75.10. RELATED INFORMATION

For more information about the Audit system, see the following sources.

### Online Sources

- The [RHEL Audit System Reference](#) Knowledgebase article.
- The [Auditd execution options in a container](#) Knowledgebase article.
- The [Linux Audit Documentation Project page](#).

## Installed Documentation

Documentation provided by the **audit** package can be found in the **/usr/share/doc/audit** directory.

### Manual Pages

- **audispd.conf(5)**
- **auditd.conf(5)**
- **ausearch-expression(5)**
- **audit.rules(7)**
- **audispd(8)**
- **auditctl(8)**
- **auditd(8)**
- **aulast(8)**
- **aulastlog(8)**
- **aureport(8)**
- **ausearch(8)**
- **ausyscall(8)**
- **autrace(8)**
- **auvirt(8)**

## PART VI. DESIGN OF KERNEL

# CHAPTER 76. THE LINUX KERNEL RPM

The following sections describe the Linux kernel RPM package provided and maintained by Red Hat.

## 76.1. WHAT AN RPM IS

An RPM package is a file containing other files and their metadata (information about the files that are needed by the system).

Specifically, an RPM package consists of the **cpio** archive.

The **cpio** archive contains:

- Files
- RPM header (package metadata)

The **rpm** package manager uses this metadata to determine dependencies, where to install files, and other information.

### Types of RPM packages

There are two types of RPM packages. Both types share the file format and tooling, but have different contents and serve different purposes:

- Source RPM (SRPM)  
An SRPM contains source code and a SPEC file, which describes how to build the source code into a binary RPM. Optionally, the patches to source code are included as well.
- Binary RPM  
A binary RPM contains the binaries built from the sources and patches.

## 76.2. THE LINUX KERNEL RPM PACKAGE OVERVIEW

The **kernel** RPM is a meta package that does not contain any files, but rather ensures that the following sub-packages are properly installed:

- **kernel-core** - contains a minimal number of kernel modules needed for core functionality. This sub-package alone could be used in virtualized and cloud environments to provide a Red Hat Enterprise Linux 8 kernel with a quick boot time and a small disk size footprint.
- **kernel-modules** - contains further kernel modules.
- **kernel-modules-extra** - contains kernel modules for rare hardware.

The small set of **kernel** sub-packages above aims to provide a reduced maintenance surface to system administrators especially in virtualized and cloud environments.

The other common kernel packages are for example:

- **kernel-debug** – Contains a kernel with numerous debugging options enabled for kernel diagnosis, at the expense of reduced performance.
- **kernel-tools** – Contains tools for manipulating the Linux kernel and supporting documentation.
- **kernel-devel** – Contains the kernel headers and makefiles sufficient to build modules against the **kernel** package.

- **kernel-abi-whitelists** – Contains information pertaining to the Red Hat Enterprise Linux kernel ABI, including a list of kernel symbols that are needed by external Linux kernel modules and a **yum** plug-in to aid enforcement.
- **kernel-headers** – Includes the C header files that specify the interface between the Linux kernel and user-space libraries and programs. The header files define structures and constants that are needed for building most standard programs.

## 76.3. DISPLAYING CONTENTS OF THE KERNEL PACKAGE

The following procedure describes how to view the contents of the kernel package and its sub-packages without installing them using the **rpm** command.

### Prerequisites

- Obtained **kernel**, **kernel-core**, **kernel-modules**, **kernel-modules-extra** RPM packages for your CPU architecture

### Procedure

- List modules for **kernel**:

```
$ rpm -qlp <kernel_rpm>
(contains no files)
...
```

- List modules for **kernel-core**:

```
$ rpm -qlp <kernel-core_rpm>
...
/lib/modules/4.18.0-80.el8.x86_64/kernel/fs/udf/udf.ko.xz
/lib/modules/4.18.0-80.el8.x86_64/kernel/fs/xfs
/lib/modules/4.18.0-80.el8.x86_64/kernel/fs/xfs/xfs.ko.xz
/lib/modules/4.18.0-80.el8.x86_64/kernel/kernel
/lib/modules/4.18.0-80.el8.x86_64/kernel/kernel/trace
/lib/modules/4.18.0-80.el8.x86_64/kernel/kernel/trace/ring_buffer_benchmark.ko.xz
/lib/modules/4.18.0-80.el8.x86_64/kernel/lib
/lib/modules/4.18.0-80.el8.x86_64/kernel/lib/cordic.ko.xz
...
```

- List modules for **kernel-modules**:

```
$ rpm -qlp <kernel-modules_rpm>
...
/lib/modules/4.18.0-80.el8.x86_64/kernel/drivers/infiniband/hw/mlx4/mlx4_ib.ko.xz
/lib/modules/4.18.0-80.el8.x86_64/kernel/drivers/infiniband/hw/mlx5/mlx5_ib.ko.xz
/lib/modules/4.18.0-80.el8.x86_64/kernel/drivers/infiniband/hw/qedr/qedr.ko.xz
/lib/modules/4.18.0-80.el8.x86_64/kernel/drivers/infiniband/hw/usnic/usnic_verbs.ko.xz
/lib/modules/4.18.0-
80.el8.x86_64/kernel/drivers/infiniband/hw/vmw_pvrdma/vmw_pvrdma.ko.xz
...
```

- List modules for **kernel-modules-extra**:

```
$ rpm -qlp <kernel-modules-extra_rpm>
...
/lib/modules/4.18.0-80.el8.x86_64/extra/net/sched/sch_cbq.ko.xz
/lib/modules/4.18.0-80.el8.x86_64/extra/net/sched/sch_choke.ko.xz
/lib/modules/4.18.0-80.el8.x86_64/extra/net/sched/sch_drr.ko.xz
/lib/modules/4.18.0-80.el8.x86_64/extra/net/sched/sch_dsmark.ko.xz
/lib/modules/4.18.0-80.el8.x86_64/extra/net/sched/sch_gred.ko.xz
...
```

## Additional resources

- For information on how to use the **rpm** command on already installed **kernel** RPM, including its sub-packages, see the **rpm(8)** manual page.
- Introduction to *RPM packages*

# CHAPTER 77. UPDATING KERNEL WITH YUM

The following sections bring information about the Linux kernel provided and maintained by Red Hat (Red Hat kernel), and how to keep the Red Hat kernel updated. As a consequence, the operating system will have all the latest bug fixes, performance enhancements, and patches ensuring compatibility with new hardware.

## 77.1. WHAT IS THE KERNEL

The kernel is a core part of a Linux operating system, which manages the system resources, and provides interface between hardware and software applications. The Red Hat kernel is a custom-built kernel based on the upstream Linux mainline kernel that Red Hat engineers further develop and harden with a focus on stability and compatibility with the latest technologies and hardware.

Before Red Hat releases a new kernel version, the kernel needs to pass a set of rigorous quality assurance tests.

The Red Hat kernels are packaged in the RPM format so that they are easy to upgrade and verify by the **yum** package manager.



### WARNING

Kernels that have not been compiled by Red Hat are **not** supported by Red Hat.

## 77.2. WHAT IS YUM

This section refers to description of the **yum** *package manager*.

### Additional resources

- For more information on **yum** see the relevant sections of [Configuring basic system settings](#).

## 77.3. UPDATING THE KERNEL

The following procedure describes how to update the kernel using the **yum** package manager.

### Procedure

1. To update the kernel, use the following:

```
yum update kernel
```

This command updates the kernel along with all dependencies to the latest available version.

2. Reboot your system for the changes to take effect.



## NOTE

When upgrading from Red Hat Enterprise Linux 7 to Red Hat Enterprise Linux 8, follow relevant sections of the [Upgrading from RHEL 7 to RHEL 8](#) document.

## 77.4. INSTALLING THE KERNEL

The following procedure describes how to install new kernels using the **yum** package manager.

### Procedure

- To install a specific kernel version, use the following:

```
yum install kernel-{version}
```

### Additional resources

- For a list of available kernels, refer to [Red Hat Code Browser](#).
- For a list of release dates of specific kernel versions, see [this article](#).

# CHAPTER 78. CONFIGURING KERNEL COMMAND-LINE PARAMETERS

Kernel command-line parameters are a way to change the behavior of certain aspects of the Red Hat Enterprise Linux kernel at boot time. As a system administrator, you have full control over what options get set at boot. Certain kernel behaviors are only able to be set at boot time, so understanding how to make these changes is a key administration skill.



## IMPORTANT

Opting to change the behavior of the system by modifying kernel command-line parameters may have negative effects on your system. You should therefore test changes prior to deploying them in production. For further guidance, contact Red Hat Support.

## 78.1. UNDERSTANDING KERNEL COMMAND-LINE PARAMETERS

Kernel command-line parameters are used for boot time configuration of:

- The Red Hat Enterprise Linux kernel
- The initial RAM disk
- The user space features

Kernel boot time parameters are often used to overwrite default values and for setting specific hardware settings.

By default, the kernel command-line parameters for systems using the GRUB2 bootloader are defined in the **kernelopts** variable of the **/boot/grub2/grubenv** file for all kernel boot entries.



## NOTE

For IBM Z, the kernel command-line parameters are stored in the boot entry configuration file because the zipl bootloader does not support environment variables. Thus, the **kernelopts** environment variable cannot be used.

### Additional resources

- For more information about what kernel command-line parameters you can modify, see **kernel-command-line(7)**, **bootparam(7)** and **dracut.cmdline(7)** manual pages.
- For more information about the **kernelopts** variable, see the knowledge base article, [How to install and boot custom kernels in Red Hat Enterprise Linux 8](#).

## 78.2. WHAT GRUBBY IS

**grubby** is a utility for manipulating bootloader-specific configuration files.

You can use **grubby** also for changing the default boot entry, and for adding/removing arguments from a GRUB2 menu entry.

For more details see the **grubby(8)** manual page.

## 78.3. WHAT BOOT ENTRIES ARE

A boot entry is a collection of options which are stored in a configuration file and tied to a particular kernel version. In practice, you have at least as many boot entries as your system has installed kernels. The boot entry configuration file is located in the **/boot/loader/entries** directory and can look like this:

```
6f9cc9cb7d7845d49698c9537337cedc-4.18.0-5.el8.x86_64.conf
```

The file name above consists of a machine ID stored in the **/etc/machine-id** file, and a kernel version.

The boot entry configuration file contains information about the kernel version, the initial ramdisk image, and the **kernelopts** environment variable, which contains the kernel command-line parameters. The contents of a boot entry config can be seen below:

```
title Red Hat Enterprise Linux (4.18.0-74.el8.x86_64) 8.0 (Ootpa)
version 4.18.0-74.el8.x86_64
linux /vmlinuz-4.18.0-74.el8.x86_64
initrd /initramfs-4.18.0-74.el8.x86_64.img $tuned_initrd
options $kernelopts $tuned_params
id rhel-20190227183418-4.18.0-74.el8.x86_64
grub_users $grub_users
grub_arg --unrestricted
grub_class kernel
```

The **kernelopts** environment variable is defined in the **/boot/grub2/grubenv** file.

### Additional resources

For more information about **kernelopts** variable, see knowledge base article [How to install and boot custom kernels in Red Hat Enterprise Linux 8](#).

## 78.4. SETTING KERNEL COMMAND-LINE PARAMETERS

To adjust the behavior of your system from the early stages of the booting process, you need to set certain kernel command-line parameters.

This section explains how to change kernel command-line parameters on various CPU architectures.

### 78.4.1. Changing kernel command-line parameters for all boot entries

This procedure describes how to change kernel command-line parameters for all boot entries on your system.

#### Prerequisites

- Verify that the **grubby** and **zipl** utilities are installed on your system.

#### Procedure

- To add a parameter:

```
grubby --update-kernel=ALL --args="<NEW_PARAMETER>"
```

For systems that use the GRUB2 bootloader, the command updates the **/boot/grub2/grubenv** file by adding a new kernel parameter to the **kernelopts** variable in that file.

On IBM Z that use the zIPL bootloader, the command adds a new kernel parameter to each **/boot/loader/entries/<ENTRY>.conf** file.

- On IBM Z, execute the **zipl** command with no options to update the boot menu.
- To remove a parameter:
 

```
grubpy --update-kernel=ALL --remove-args=<PARAMETER_TO_REMOVE>
```

  - On IBM Z, execute the **zipl** command with no options to update the boot menu.

## Additional resources

- For more information about kernel command-line parameters, see [Section 78.1, “Understanding kernel command-line parameters”](#).
- For information on the **grubby** utility, see the **grubby(8)** manual page.
- For further examples on how to use **grubby**, see the [grubby tool](#).
- For information about the **zipl** utility, see the **zipl(8)** manual page.

### 78.4.2. Changing kernel command-line parameters for a single boot entry

This procedure describes how to change kernel command-line parameters for a single boot entry on your system.

#### Prerequisites

- Verify that the **grubby** and **zipl** utilities are installed on your system.

#### Procedure

- To add a parameter:
 

```
grubpy --update-kernel=/boot/vmlinuz-$(uname -r) --args=<NEW_PARAMETER>
```

  - On IBM Z, execute the **zipl** command with no options to update the boot menu.
- To remove a parameter use the following:
 

```
grubpy --update-kernel=/boot/vmlinuz-$(uname -r) --remove-args=<PARAMETER_TO_REMOVE>
```

  - On IBM Z, execute the **zipl** command with no options to update the boot menu.



#### NOTE

On systems that use the **grub.cfg** file, there is, by default, the **options** parameter for each kernel boot entry, which is set to the **kernelopts** variable. This variable is defined in the **/boot/grub2/grubenv** configuration file.



## IMPORTANT

On GRUB 2 systems:

- + \* If the kernel command-line parameters are modified for all boot entries, the **grubby** utility updates the **kernelopts** variable in the **/boot/grub2/grubenv** file.
- + \* If kernel command-line parameters are modified for a single boot entry, the **kernelopts** variable is expanded, the kernel parameters are modified, and the resulting value is stored in the respective boot entry's **/boot/loader/entries/<RELEVANT\_KERNEL\_BOOT\_ENTRY.conf** file.
- + \* On zIPL systems, **grubby** modifies and stores the kernel command-line parameters of an individual kernel boot entry in the **/boot/loader/entries/<ENTRY>.conf** file.

## Additional resources

- For more information about kernel command-line parameters, see [Section 78.1, “Understanding kernel command-line parameters”](#).
- For information on the **grubby** utility, see the **grubby(8)** manual page.
- For further examples on how to use **grubby**, see the [\*grubby tool\*](#).
- For information about the **zipl** utility, see the **zipl(8)** manual page.

# CHAPTER 79. CONFIGURING KERNEL PARAMETERS AT RUNTIME

As a system administrator, you can modify many facets of the Red Hat Enterprise Linux kernel's behavior at runtime. This section describes how to configure kernel parameters at runtime by using the **sysctl** command and by modifying the configuration files in the **/etc/sysctl.d/** and **/proc/sys/** directories.

## 79.1. WHAT ARE KERNEL PARAMETERS

Kernel parameters are tunable values which you can adjust while the system is running. There is no requirement to reboot or recompile the kernel for changes to take effect.

It is possible to address the kernel parameters through:

- + \* The **sysctl** command
- + \* The virtual file system mounted at the **/proc/sys/** directory
- + \* The configuration files in the **/etc/sysctl.d/** directory

Tunables are divided into classes by the kernel subsystem. Red Hat Enterprise Linux has the following tunable classes:

**Table 79.1. Table of sysctl classes**

Tunable class	Subsystem
abi	Execution domains and personalities
crypto	Cryptographic interfaces
debug	Kernel debugging interfaces
dev	Device-specific information
fs	Global and specific file system tunables
kernel	Global kernel tunables
net	Network tunables
sunrpc	Sun Remote Procedure Call (NFS)
user	User Namespace limits
vm	Tuning and management of memory, buffers, and cache

### Additional resources

- For more information about **sysctl**, see **sysctl(8)** manual pages.
- For more information about **/etc/sysctl.d/** see, **sysctl.d(5)** manual pages.

## 79.2. SETTING KERNEL PARAMETERS AT RUNTIME



### IMPORTANT

Configuring kernel parameters on a production system requires careful planning. Unplanned changes may render the kernel unstable, requiring a system reboot. Verify that you are using valid options before changing any kernel values.

### 79.2.1. Configuring kernel parameters temporarily with **sysctl**

The following procedure describes how to use the **sysctl** command to temporarily set kernel parameters at runtime. The command is also useful for listing and filtering tunables.

#### Prerequisites

- [Kernel parameters introduction](#)
- Root permissions

#### Procedure

1. To list all parameters and their values, use the following:

```
sysctl -a
```

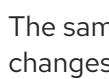


### NOTE

The **# sysctl -a** command displays kernel parameters, which can be adjusted at runtime and at boot time.

2. To configure a parameter temporarily, use the command as in the following example:

```
sysctl <TUNABLE_CLASS>.<PARAMETER>=<TARGET_VALUE>
```



### NOTE

The changes return back to default after your system reboots.

#### Additional resources

- For more information about **sysctl**, see the **sysctl(8)** manual page.
- To permanently modify kernel parameters, either use the **sysctl** command to write the values to the **/etc/sysctl.conf** file or make manual changes to the configuration files in the **/etc/sysctl.d/** directory.

## 79.2.2. Configuring kernel parameters permanently with `sysctl`

The following procedure describes how to use the `sysctl` command to permanently set kernel parameters.

### Prerequisites

- [Kernel parameters introduction](#)
- Root permissions

### Procedure

1. To list all parameters, use the following:

```
sysctl -a
```

The command displays all kernel parameters that can be configured at runtime.

2. To configure a parameter permanently:

```
sysctl -w <TUNABLE_CLASS>.<PARAMETER>=<TARGET_VALUE> >>
/etc/sysctl.conf
```

The sample command changes the tunable value and writes it to the `/etc/sysctl.conf` file, which overrides the default values of kernel parameters. The changes take effect immediately and persistently, without a need for restart.



### NOTE

To permanently modify kernel parameters you can also make manual changes to the configuration files in the `/etc/sysctl.d/` directory.

### Additional resources

- For more information about `sysctl`, see the [sysctl\(8\)](#) and [sysctl.conf\(5\)](#) manual pages.
- For more information about using the configuration files in the `/etc/sysctl.d/` directory to make permanent changes to kernel parameters, see [Using configuration files in /etc/sysctl.d/ to adjust kernel parameters](#) section.

## 79.2.3. Using configuration files in `/etc/sysctl.d/` to adjust kernel parameters

The following procedure describes how to manually modify configuration files in the `/etc/sysctl.d/` directory to permanently set kernel parameters.

### Prerequisites

- [Kernel parameters introduction](#)
- Root permissions

### Procedure

1. Create a new configuration file in **/etc/sysctl.d/**:

```
vim /etc/sysctl.d/<some_file.conf>
```

2. Include kernel parameters, one per line, as follows:

```
<TUNABLE_CLASS>.<PARAMETER>=<TARGET_VALUE>
<TUNABLE_CLASS>.<PARAMETER>=<TARGET_VALUE>
```

3. Save the configuration file.
4. Reboot the machine for the changes to take effect.

- Alternatively, to apply changes without rebooting, execute:

```
sysctl -p /etc/sysctl.d/<some_file.conf>
```

The command enables you to read values from the configuration file, which you created earlier.

## Additional resources

- For more information about **sysctl**, see the **sysctl(8)** manual page.
- For more information about **/etc/sysctl.d/**, see the **sysctl.d(5)** manual page.

### 79.2.4. Configuring kernel parameters temporarily through **/proc/sys/**

The following procedure describes how to set kernel parameters temporarily through the files in the virtual file system **/proc/sys/** directory.

#### Prerequisites

- [Kernel parameters introduction](#)
- Root permissions

#### Procedure

1. Identify a kernel parameter you want to configure:

```
ls -l /proc/sys/<TUNABLE_CLASS>/
```

The writable files returned by the command can be used to configure the kernel. The files with read-only permissions provide feedback on the current settings.

2. Assign a target value to the kernel parameter:

```
echo <TARGET_VALUE> > /proc/sys/<TUNABLE_CLASS>/<PARAMETER>
```

The command makes configuration changes that will disappear once the system is restarted.

3. Optionally, verify the value of the newly set kernel parameter:

```
cat /proc/sys/<TUNABLE_CLASS>/<PARAMETER>
```

## Additional resources

- To permanently modify kernel parameters, either use the [sysctl](#) command or make manual changes to the configuration files in the [/etc/sysctl.d](#) directory.

## 79.3. KEEPING KERNEL PANIC PARAMETERS DISABLED IN VIRTUALIZED ENVIRONMENTS

When configuring a virtualized environment in Red Hat Enterprise Linux 8 (RHEL 8), you should not enable the **softlockup\_panic** and **nmi\_watchdog** kernel parameters, as the virtualized environment may trigger a spurious soft lockup that should not require a system panic.

The following sections explain the reasons behind this advice by summarizing:

- What causes a soft lockup.
- Describing the kernel parameters that control a system's behavior on a soft lockup.
- Explaining how soft lockups may be triggered in a virtualized environment.

### 79.3.1. What is a soft lockup

A soft lockup is a situation usually caused by a bug, when a task is executing in kernel space on a CPU without rescheduling. The task also does not allow any other task to execute on that particular CPU. As a result, a warning is displayed to a user through the system console. This problem is also referred to as the soft lockup firing.

## Additional resources

- For a technical reason behind a soft lockup, example log messages, and other details, see the following [Knowledge Article](#).

### 79.3.2. Parameters controlling kernel panic

The following kernel parameters can be set to control a system's behavior when a soft lockup is detected.

#### softlockup\_panic

Controls whether or not the kernel will panic when a soft lockup is detected.

Type	Value	Effect
Integer	0	kernel does not panic on soft lockup
Integer	1	kernel panics on soft lockup

By default, on RHEL8 this value is 0.

In order to panic, the system needs to detect a hard lockup first. The detection is controlled by the **nmi\_watchdog** parameter.

### nmi\_watchdog

Controls whether lockup detection mechanisms (**watchdogs**) are active or not. This parameter is of integer type.

Value	Effect
0	disables lockup detector
1	enables lockup detector

The hard lockup detector monitors each CPU for its ability to respond to interrupts.

### watchdog\_thresh

Controls frequency of watchdog **hrtimer**, NMI events, and soft/hard lockup thresholds.

Default threshold	Soft lockup threshold
10 seconds	$2 * \text{watchdog\_thresh}$

Setting this parameter to zero disables lockup detection altogether.

### Additional resources

- For further information about **nmi\_watchdog** and **softlockup\_panic**, see the [Softlockup detector and hardlockup detector](#) document.
- For more details about **watchdog\_thresh**, see the [Kernel sysctl](#) document.

#### 79.3.3. Spurious soft lockups in virtualized environments

The soft lockup firing on physical hosts, as described in [Section 79.3.1, “What is a soft lockup”](#), usually represents a kernel or hardware bug. The same phenomenon happening on guest operating systems in virtualized environments may represent a false warning.

Heavy work-load on a host or high contention over some specific resource such as memory, usually causes a spurious soft lockup firing. This is because the host may schedule out the guest CPU for a period longer than 20 seconds. Then when the guest CPU is again scheduled to run on the host, it experiences a *time jump* which triggers due timers. The timers include also watchdog **hrtimer**, which can consequently report a soft lockup on the guest CPU.

Because a soft lockup in a virtualized environment may be spurious, you should not enable the kernel parameters that would cause a system panic when a soft lockup is reported on a guest CPU.



#### IMPORTANT

To understand soft lockups in guests, it is essential to know that the host schedules the guest as a task, and the guest then schedules its own tasks.

## Additional resources

- For soft lockup definition and technicalities behind its functioning, see [Section 79.3.1, "What is a soft lockup"](#).
- To learn about components of RHEL 8 virtualized environments and their interaction, see [RHEL 8 virtual machine components and their interaction](#).

## 79.4. ADJUSTING KERNEL PARAMETERS FOR DATABASE SERVERS

There are different sets of kernel parameters which can affect performance of specific database applications. The following sections explain what kernel parameters to configure to secure efficient operation of database servers and databases.

### 79.4.1. Introduction to database servers

A database server is a hardware device which has a certain amount of main memory, and a database (DB) application installed. This DB application provides services as a means of writing the cached data from the main memory, which is usually small and expensive, to DB files (database). These services are provided to multiple clients on a network. There can be as many DB servers as a machine's main memory and storage allows.

Red Hat Enterprise Linux 8 provides the following database applications:

- MariaDB 10.3
- MySQL 8.0
- PostgreSQL 10
- PostgreSQL 9.6
- PostgreSQL 12 – available since RHEL 8.1.1

### 79.4.2. Parameters affecting performance of database applications

The following kernel parameters affect performance of database applications.

#### fs.aio-max-nr

Defines the maximum number of asynchronous I/O operations the system can handle on the server.



#### NOTE

Raising the **fs.aio-max-nr** parameter produces no additional changes beyond increasing the aio limit.

#### fs.file-max

Defines the maximum number of file handles (temporary file names or IDs assigned to open files) the system supports at any instance.

The kernel dynamically allocates file handles whenever a file handle is requested by an application. The kernel however does not free these file handles when they are released by the application. The kernel recycles these file handles instead. This means that over time the total number of allocated file handles will increase even though the number of currently used file handles may be low.

**kernel.shmall**

Defines the total number of shared memory pages that can be used system-wide. To use the entire main memory, the value of the **kernel.shmall** parameter should be  $\leq$  total main memory size.

**kernel.shmmax**

Defines the maximum size in bytes of a single shared memory segment that a Linux process can allocate in its virtual address space.

**kernel.shmmni**

Defines the maximum number of shared memory segments the database server is able to handle.

**net.ipv4.ip\_local\_port\_range**

Defines the port range the system can use for programs which want to connect to a database server without a specific port number.

**net.core.rmem\_default**

Defines the default receive socket memory through Transmission Control Protocol (TCP).

**net.core.rmem\_max**

Defines the maximum receive socket memory through Transmission Control Protocol (TCP).

**net.core.wmem\_default**

Defines the default send socket memory through Transmission Control Protocol (TCP).

**net.core.wmem\_max**

Defines the maximum send socket memory through Transmission Control Protocol (TCP).

**vm.dirty\_bytes / vm.dirty\_ratio**

Defines a threshold in bytes / in percentage of dirty-able memory at which a process generating dirty data is started in the **write()** function.

**NOTE**

Either **vm.dirty\_bytes** or **vm.dirty\_ratio** can be specified at a time.

**vm.dirty\_background\_bytes / vm.dirty\_background\_ratio**

Defines a threshold in bytes / in percentage of dirty-able memory at which the kernel tries to actively write dirty data to hard-disk.

**NOTE**

Either **vm.dirty\_background\_bytes** or **vm.dirty\_background\_ratio** can be specified at a time.

**vm.dirty\_writeback\_centisecs**

Defines a time interval between periodic wake-ups of the kernel threads responsible for writing dirty data to hard-disk.

This kernel parameters measures in 100th's of a second.

**vm.dirty\_expire\_centisecs**

Defines the time after which dirty data is old enough to be written to hard-disk.

This kernel parameters measures in 100th's of a second.

**Additional resources**

- For explanation of dirty data writebacks, how they work, and what kernel parameters relate to them, see the [\*Dirty pagecache writeback and vm.dirty parameters\*](#) document.

# CHAPTER 80. INSTALLING AND CONFIGURING KDUMP

## 80.1. WHAT IS KDUMP

**kdump** is a service providing a crash dumping mechanism. The service enables you to save the contents of the system's memory for later analysis. **kdump** uses the **kexec** system call to boot into the second kernel (a *capture kernel*) without rebooting; and then captures the contents of the crashed kernel's memory (a *crash dump* or a *vmcore*) and saves it. The second kernel resides in a reserved part of the system memory.

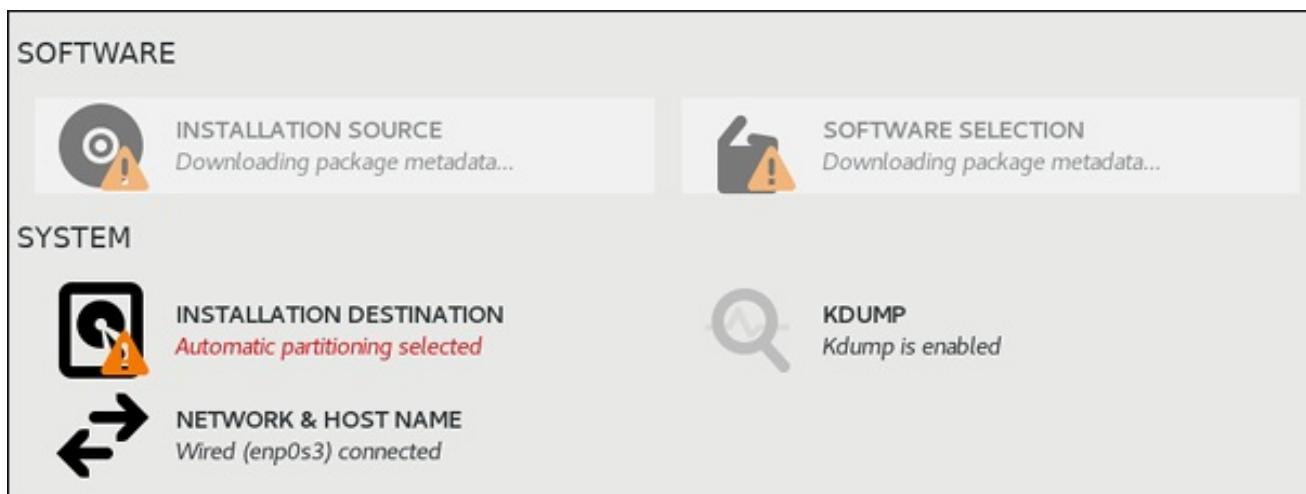


### IMPORTANT

A kernel crash dump can be the only information available in the event of a system failure (a critical bug). Therefore, ensuring that **kdump** is operational is important in mission-critical environments. Red Hat advise that system administrators regularly update and test **kexec-tools** in your normal kernel update cycle. This is especially important when new kernel features are implemented.

## 80.2. INSTALLING KDUMP

In many cases, the **kdump** service is installed and activated by default on the new Red Hat Enterprise Linux installations. The **Anaconda** installer provides a screen for **kdump** configuration when performing an interactive installation using the graphical or text interface. The installer screen is titled **Kdump** and is available from the main **Installation Summary** screen, and only allows limited configuration - you can only select whether **kdump** is enabled and how much memory is reserved.



Some installation options, such as custom Kickstart installations, in some cases do not install or enable **kdump** by default. If this is the case on your system, follow the procedure below to install **kdump**.

### Prerequisites

- An active Red Hat Enterprise Linux subscription
- A repository containing the **kexec-tools** package for your system CPU architecture
- Fulfilled **kdump** requirements

### Procedure

1. Execute the following command to check whether **kdump** is installed on your system:

```
$ rpm -q kexec-tools
```

Output if the package is installed:

```
kexec-tools-2.0.17-11.el8.x86_64
```

Output if the package is not installed:

```
package kexec-tools is not installed
```

2. Install **kdump** and other necessary packages by:

```
yum install kexec-tools
```



### IMPORTANT

Starting with Red Hat Enterprise Linux 7.4 (kernel-3.10.0-693.el7) the **Intel IOMMU** driver is supported with **kdump**. For prior versions, Red Hat Enterprise Linux 7.3 (kernel-3.10.0-514[.XYZ].el7) and earlier, it is advised that **Intel IOMMU** support is disabled, otherwise kdump kernel is likely to become unresponsive.

## Additional resources

- Information about memory requirements for **kdump** is available in [Section 80.5.1, “Memory requirements for kdump”](#).

## 80.3. CONFIGURING KDUMP ON THE COMMAND LINE

### 80.3.1. Configuring kdump memory usage

The memory reserved for the **kdump** feature is always reserved during the system boot. The amount of memory is specified in the system’s Grand Unified Bootloader (GRUB) 2 configuration. The procedure below describes how to configure the memory reserved for **kdump** through the command line.

#### Prerequisites

- Fulfilled [kdump requirements](#)

#### Procedure

1. Edit the **/etc/default/grub** file using the root permissions.
2. Set the **crashkernel**= option to the required value.  
For example, to reserve 128 MB of memory, use the following:

```
crashkernel=128M
```

Alternatively, you can set the amount of reserved memory to a variable depending on the total amount of installed memory. The syntax for memory reservation into a variable is **crashkernel=<range1>:<size1>,<range2>:<size2>**. For example:

```
crashkernel=512M-2G:64M,2G-:128M
```

The above example reserves 64 MB of memory if the total amount of system memory is 512 MB or higher and lower than 2 GB. If the total amount of memory is more than 2 GB, 128 MB is reserved for **kdump** instead.

- Offset the reserved memory.

Some systems require to reserve memory with a certain fixed offset since **crashkernel** reservation is very early, and it wants to reserve some area for special usage. If the offset is set, the reserved memory begins there. To offset the reserved memory, use the following syntax:

```
crashkernel=128M@16M
```

The example above means that **kdump** reserves 128 MB of memory starting at 16 MB (physical address 0x01000000). If the offset parameter is set to 0 or omitted entirely, **kdump** offsets the reserved memory automatically. This syntax can also be used when setting a variable memory reservation as described above; in this case, the offset is always specified last (for example, **crashkernel=512M-2G:64M,2G-:128M@16M**).

3. Use the following command to update the GRUB2 configuration file:

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```



#### NOTE

The alternative way to configure memory for **kdump** is to append the **crashkernel=<SOME\_VALUE>** parameter to the **kerneloops** variable with the **grub2-editenv** which will update all of your boot entries. Or you can use the **grubby** utility to update kernel command line parameters of just one entry.

#### Additional resources

- The **crashkernel** option can be defined in multiple ways. The **auto** value enables automatic configuration of reserved memory based on the total amount of memory in the system, following the guidelines described in [Memory requirements for kdump](#).
- For more information on boot entries, **kerneloops**, and how to work with **grub2-editenv** and **grubby** see [Configuring kernel command line parameters](#).

### 80.3.2. Configuring the kdump target

When a kernel crash is captured, the core dump can be either stored as a file in a local file system, written directly to a device, or sent over a network using the **NFS** (Network File System) or **SSH** (Secure Shell) protocol. Only one of these options can be set at a time, and the default behavior is to store the **vmcore** file in the **/var/crash/** directory of the local file system.

#### Prerequisites

- Fulfilled **kdump** requirements

#### Procedure

To store the **vmcore** file in **/var/crash/** directory of the local file system:

- Edit the **/etc/kdump.conf** file and specify the path:

path /var/crash

The option **path /var/crash** represents the path to the file system in which **kdump** saves the **vmcore** file. When you specify a dump target in the **/etc/kdump.conf** file, then the **path** is relative to the specified dump target.

If you do not specify a dump target in the **/etc/kdump.conf** file, then the **path** represents the absolute path from the root directory. Depending on what is mounted in the current system, the dump target and the adjusted dump path are taken automatically.



### WARNING

**kdump** saves the **vmcore** file in **/var/crash/var/crash** directory, when the dump target is mounted at **/var/crash** and the option **path** is also set as **/var/crash** in the **/etc/kdump.conf** file. For example, in the following instance, the **ext4** file system is already mounted at **/var/crash** and the **path** are set as **/var/crash**:

```
grep -v ^# etc/kdump.conf | grep -v ^$
ext4 /dev/mapper/vg00-varcrashvol
path /var/crash
core_collector makedumpfile -c --message-level 1 -d 31
```

This results in the **/var/crash/var/crash** path. To solve this problem, use the option **path /** instead of **path /var/crash**

To change the local directory in which the core dump is to be saved, as **root**, edit the **/etc/kdump.conf** configuration file as described below.

1. Remove the hash sign ("#") from the beginning of the **#path /var/crash** line.
2. Replace the value with the intended directory path. For example:

path /usr/local/cores



### IMPORTANT

In Red Hat Enterprise Linux 8, the directory defined as the kdump target using the **path** directive must exist when the **kdump** systemd service is started – otherwise the service fails. This behavior is different from earlier releases of Red Hat Enterprise Linux, where the directory was being created automatically if it did not exist when starting the service.

To write the file to a different partition, as **root**, edit the **/etc/kdump.conf** configuration file as described below.

1. Remove the hash sign ("#") from the beginning of the **#ext4** line, depending on your choice.
  - device name (the **#ext4 /dev/vg/lv\_kdump** line)

- file system label (the **#ext4 LABEL=/boot** line)
  - UUID (the **#ext4 UUID=03138356-5e61-4ab3-b58e-27507ac41937** line)
2. Change the file system type as well as the device name, label or UUID to the desired values. For example:

```
ext4 UUID=03138356-5e61-4ab3-b58e-27507ac41937
```



### IMPORTANT

It is recommended to specify storage devices using a **LABEL=** or **UUID=**. Disk device names such as **/dev/sda3** are not guaranteed to be consistent across reboot.



### IMPORTANT

When dumping to Direct Access Storage Device (DASD) on IBM Z hardware, it is essential that the dump devices are correctly specified in **/etc/dasd.conf** before proceeding.

To write the dump directly to a device:

1. Remove the hash sign ("#") from the beginning of the **#raw /dev/vg/lv\_kdump** line.
2. Replace the value with the intended device name. For example:

```
raw /dev/sdb1
```

To store the dump to a remote machine using the **NFS** protocol:

1. Remove the hash sign ("#") from the beginning of the **#nfs my.server.com:/export/tmp** line.
2. Replace the value with a valid hostname and directory path. For example:

```
nfs penguin.example.com:/export/cores
```

To store the dump to a remote machine using the **SSH** protocol:

1. Remove the hash sign ("#") from the beginning of the **#ssh user@my.server.com** line.
2. Replace the value with a valid username and hostname.
3. Include your **SSH** key in the configuration.
  - Remove the hash sign from the beginning of the **#sshkey /root/.ssh/kdump\_id\_rsa** line.
  - Change the value to the location of a key valid on the server you are trying to dump to. For example:

```
ssh john@penguin.example.com
sshkey /root/.ssh/mykey
```

## Additional resources

- For a complete list of currently supported and unsupported targets sorted by type, see [Section 80.5.3, “Supported kdump targets”](#).
- For information on how to configure an SSH server and set up a key-based authentication, see [Configuring basic system settings](#) in Red Hat Enterprise Linux.

### 80.3.3. Configuring the core collector

**kdump** uses a program specified as **core collector** to capture the vmcore. Currently, the only fully supported **core collector** is the **makedumpfile** utility. It has several configurable options, which affect the collection process. For example the extent of collected data, or whether the resulting vmcore should be compressed.

To enable and configure the **core collector**, follow the procedure below.

#### Prerequisites

- Fulfilled **kdump** requirements

#### Procedure

- As **root**, edit the **/etc/kdump.conf** configuration file and remove the hash sign ("#") from the beginning of the **#core\_collector makedumpfile -l --message-level 1 -d 31**.
- Add the **-c** parameter. For example:

```
core_collector makedumpfile -c
```

The command above enables the dump file compression.

- Add the **-d value** parameter. For example:

```
core_collector makedumpfile -d 17 -c
```

The command above removes both zero and free pages from the dump. The **value** represents a bitmask, where each bit is associated with a certain type of memory pages and determines whether that type of pages will be collected. For description of respective bits see [Section 80.5.4, “Supported kdump filtering levels”](#).

#### Additional resources

- See the **makedumpfile(8)** man page for a complete list of available options.

### 80.3.4. Configuring the kdump default failure responses

By default, when **kdump** fails to create a vmcore dump file at the target location specified in [Section 80.3.2, “Configuring the kdump target”](#), the system reboots, and the dump is lost in the process. To change this behavior, follow the procedure below.

#### Prerequisites

- Fulfilled **kdump** requirements

## Procedure

1. As **root**, remove the hash sign ("#") from the beginning of the **#default shell** line in the **/etc/kdump.conf** configuration file.
2. Replace the value with a desired action as described in [Section 80.5.5, “Supported default failure responses”](#). For example:

```
default poweroff
```

### 80.3.5. Enabling and disabling the kdump service

To start the **kdump** service at boot time, follow the procedure below.

## Prerequisites

- Fulfilled [kdump requirements](#).
- All [configuration](#) is set up according to your needs.

## Procedure

1. To enable the **kdump** service, use the following command:

```
systemctl enable kdump.service
```

This enables the service for **multi-user.target**.

2. To start the service in the current session, use the following command:

```
systemctl start kdump.service
```

3. To stop the **kdump** service, type the following command:

```
systemctl stop kdump.service
```

4. To disable the **kdump** service, execute the following command:

```
systemctl disable kdump.service
```



### WARNING

It is recommended to set **kptr\_restrict=1** as default. When **kptr\_restrict** is set to (1) as default, the **kdumpctl** service loads the crash kernel even if Kernel Address Space Layout (KASLR) is enabled or not enabled.

## Troubleshooting step

When **kptr\_restrict** is not set to (1), and if KASLR is enabled, the contents of **/proc/kcore** file are generated as all zeros. Consequently, the **kdumpctl** service fails to access the **/proc/kcore** and load the crash kernel.

To work around this problem, the **kexec-kdump-howto.txt** file displays a warning message, which specifies to keep the recommended setting as **kptr\_restrict=1**.

To ensure that **kdumpctl** service loads the crash kernel, verify that:

- Kernel **kptr\_restrict=1** in the **sysctl.conf** file.

## Additional resources

- For more information on **systemd** and configuring services in general, see [Configuring basic system settings](#) in Red Hat Enterprise Linux.

## 80.4. CONFIGURING KDUMP IN THE WEB CONSOLE

The following sections provide an overview of how to setup and test the **kdump** configuration through the Red Hat Enterprise Linux web console. The web console is part of a default installation of Red Hat Enterprise Linux 8 and enables or disables the **kdump** service at boot time. Further, the web console conveniently enables you to configure the reserved memory for **kdump**; or to select the **vmcore** saving location in an uncompressed or compressed format.

### Prerequisites

- See [Red Hat Enterprise Linux web console](#) for further details.

### 80.4.1. Configuring kdump memory usage and target location in web console

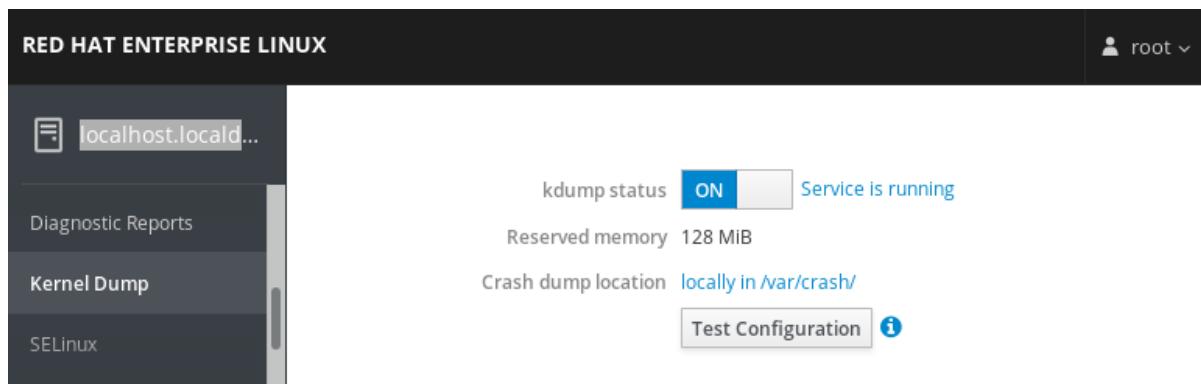
The procedure below shows you how to use the **Kernel Dump** tab in the Red Hat Enterprise Linux web console interface to configure the amount of memory that is reserved for the kdump kernel. The procedure also describes how to specify the target location of the **vmcore** dump file and how to test your configuration.

### Prerequisites

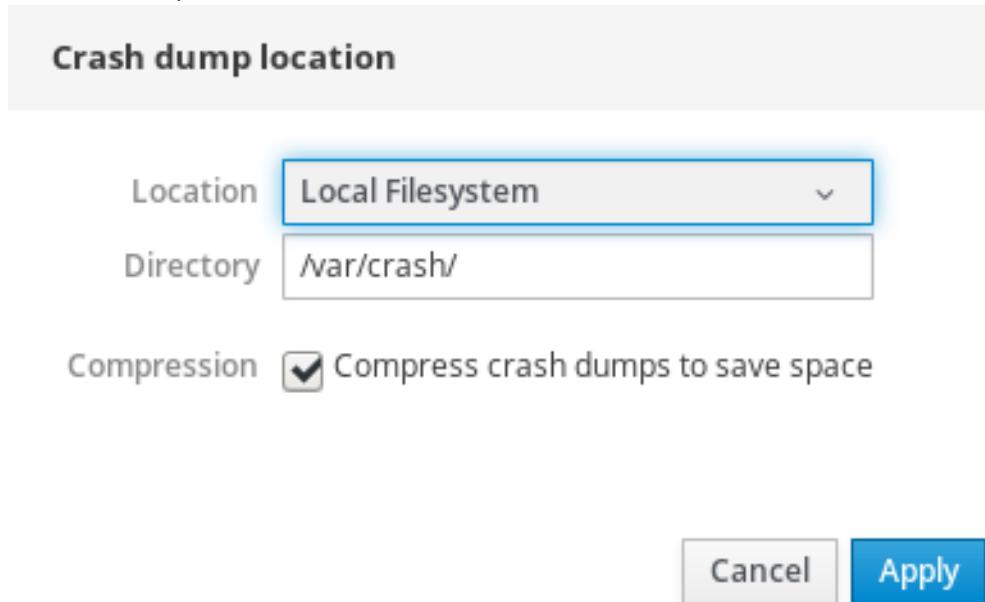
- Introduction to operating the [web console](#)

### Procedure

- Open the **Kernel Dump** tab and start the **kdump** service.
- Configure the **kdump** memory usage through the [command line](#).
- Click the link next to the **Crash dump location** option.



4. Select the **Local Filesystem** option from the drop-down and specify the directory you want to save the dump in.



- Alternatively, select the **Remote over SSH** option from the drop-down to send the vmcore to a remote machine using the SSH protocol. Fill the **Server**, **ssh key**, and **Directory** fields with the remote machine address, ssh key location, and a target directory.
- Another choice is to select the **Remote over NFS** option from the drop-down and fill the **Mount** field to send the vmcore to a remote machine using the NFS protocol.



#### NOTE

Tick the **Compression** check box to reduce the size of the vmcore file.

5. Test your configuration by crashing the kernel.

kdump status ON Service is running

Reserved memory 128 MiB

Crash dump location [locally in /var/crash/](#)

[Test Configuration](#) 



### WARNING

This step disrupts execution of the kernel and results in a system crash and loss of data.

## Additional resources

- For a complete list of currently supported targets for **kdump**, see [Supported kdump targets](#).
- For information on how to configure an SSH server and set up a key-based authentication, see [Using secure communications between two systems with OpenSSH](#).

## 80.5. SUPPORTED KDUMP CONFIGURATIONS AND TARGETS

### 80.5.1. Memory requirements for kdump

In order for **kdump** to be able to capture a kernel crash dump and save it for further analysis, a part of the system memory has to be permanently reserved for the capture kernel. When reserved, this part of the system memory is not available to the main kernel.

The memory requirements vary based on certain system parameters. One of the major factors is the system's hardware architecture. To find out the exact machine architecture (such as Intel 64 and AMD64, also known as x86\_64) and print it to standard output, use the following command:

```
$ uname -m
```

The table below contains a list of minimum memory requirements to automatically reserve a memory size for **kdump**. The size changes according to the system's architecture and total available physical memory.

**Table 80.1. Minimum Amount of Reserved Memory Required for kdump**

Architecture	Available Memory	Minimum Reserved Memory
AMD64 and Intel 64 ( <b>x86_64</b> )	1 GB to 64 GB	160 MB of RAM.

Architecture	Available Memory	Minimum Reserved Memory
	64 GB to 1 TB	256 MB of RAM.
	1 TB and more	512 MB of RAM.
64-bit ARM architecture ( <b>arm64</b> )	2 GB and more	512 MB of RAM.
IBM Power Systems ( <b>ppc64le</b> )	2 GB to 4 GB	384 MB of RAM.
	4 GB to 16 GB	512 MB of RAM.
	16 GB to 64 GB	1 GB of RAM.
	64 GB to 128 GB	2 GB of RAM.
	128 GB and more	4 GB of RAM.
IBM Z ( <b>s390x</b> )	4 GB to 64 GB	160 MB of RAM.
	64 GB to 1 TB	256 MB of RAM.
	1 TB and more	512 MB of RAM.

On many systems, **kdump** is able to estimate the amount of required memory and reserve it automatically. This behavior is enabled by default, but only works on systems that have more than a [certain amount of total available memory](#), which varies based on the system architecture.



## IMPORTANT

The automatic configuration of reserved memory based on the total amount of memory in the system is a best effort estimation. The actual required memory may vary due to other factors such as I/O devices. Using not enough of memory might cause that a debug kernel is not able to boot as a capture kernel in case of a kernel panic. To avoid this problem, sufficiently increase the crash kernel memory.

## Additional resources

- For information on how to change memory settings on the command line, see [Section 80.3.1, "Configuring kdump memory usage"](#).
- For instructions on how to set up the amount of reserved memory through the web console, see [Section 80.4.1, "Configuring kdump memory usage and target location in web console"](#).
- For more information about various Red Hat Enterprise Linux technology capabilities and limits, see the [technology capabilities and limits tables](#).

### 80.5.2. Minimum threshold for automatic memory reservation

On some systems, it is possible to allocate memory for **kdump** automatically, either by using the **crashkernel=auto** parameter in the boot loader configuration file, or by enabling this option in the graphical configuration utility. For this automatic reservation to work, however, a certain amount of total memory needs to be available in the system. The amount differs based on the system's architecture.

The table below lists the thresholds for automatic memory allocation. If the system has less memory than specified in the table, the memory needs to be [reserved manually](#).

**Table 80.2. Minimum Amount of Memory Required for Automatic Memory Reservation**

Architecture	Required Memory
AMD64 and Intel 64 ( <b>x86_64</b> )	2 GB
IBM Power Systems ( <b>ppc64le</b> )	2 GB
IBM Z ( <b>s390x</b> )	4 GB

### Additional resources

- For information on how to manually change these settings on the command line, see [Section 80.3.1, “Configuring kdump memory usage”](#).
- For instructions on how to manually change the amount of reserved memory through the web console, see [Section 80.4.1, “Configuring kdump memory usage and target location in web console”](#).

### 80.5.3. Supported kdump targets

When a kernel crash is captured, the `vmcore` dump file can be either written directly to a device, stored as a file on a local file system, or sent over a network. The table below contains a complete list of dump targets that are currently supported or explicitly unsupported by **kdump**.

**Table 80.3. Supported kdump Targets**

Type	Supported Targets	Unsupported Targets
Raw device	All locally attached raw disks and partitions.	
Local file system	<b>ext2</b> , <b>ext3</b> , <b>ext4</b> , and <b>xfs</b> file systems on directly attached disk drives, hardware RAID logical drives, LVM devices, and <b>mdraid</b> arrays.	Any local file system not explicitly listed as supported in this table, including the <b>auto</b> type (automatic file system detection).
Remote directory	Remote directories accessed using the <b>NFS</b> or <b>SSH</b> protocol over <b>IPv4</b> .	Remote directories on the <b>rootfs</b> file system accessed using the <b>NFS</b> protocol.

Type	Supported Targets	Unsupported Targets
Remote directories accessed using the <b>iSCSI</b> protocol over both hardware and software initiators.	Remote directories accessed using the <b>iSCSI</b> protocol on <b>be2iscsi</b> hardware.	Multipath-based storages.
		Remote directories accessed over <b>IPv6</b> .
		Remote directories accessed using the <b>SMB</b> or <b>CIFS</b> protocol.
		Remote directories accessed using the <b>FCoE</b> ( <i>Fibre Channel over Ethernet</i> ) protocol.
		Remote directories accessed using wireless network interfaces.



## IMPORTANT

Utilizing firmware assisted dump (**fadump**) to capture a vmcore and store it to a remote machine using SSH or NFS protocol causes renaming of the network interface to **kdump- <interface-name>**. The renaming happens if the **<interface-name>** is generic, for example **\*eth#, net#**, and so on. This problem occurs because the vmcore capture scripts in the initial RAM disk (**initrd**) add the **kdump-** prefix to the network interface name to secure persistent naming. Since the same **initrd** is used also for a regular boot, the interface name is changed for the production kernel too.

## Additional resources

- For information on how to configure the target type on the command line, see [Section 80.3.2, “Configuring the kdump target”](#).
- For information on how to configure the target through the web console, see [Section 80.4.1, “Configuring kdump memory usage and target location in web console”](#).

### 80.5.4. Supported kdump filtering levels

To reduce the size of the dump file, **kdump** uses the **makedumpfile** core collector to compress the data and optionally to omit unwanted information. The table below contains a complete list of filtering levels that are currently supported by the **makedumpfile** utility.

Table 80.4. Supported Filtering Levels

Option	Description
1	Zero pages

Option	Description
<b>2</b>	Cache pages
<b>4</b>	Cache private
<b>8</b>	User pages
<b>16</b>	Free pages



#### NOTE

The **makedumpfile** command supports removal of transparent huge pages and hugetlbfs pages. Consider both these types of hugepages User Pages and remove them using the **-8** level.

#### Additional resources

- For instructions on how to configure the core collector on the command line, see [Section 80.3.3, “Configuring the core collector”](#).

#### 80.5.5. Supported default failure responses

By default, when **kdump** fails to create a core dump, the operating system reboots. You can, however, configure **kdump** to perform a different operation in case it fails to save the core dump to the primary target. The table below lists all default actions that are currently supported.

**Table 80.5. Supported Default Actions**

Option	Description
<b>dump_to_rootfs</b>	Attempt to save the core dump to the root file system. This option is especially useful in combination with a network target: if the network target is unreachable, this option configures kdump to save the core dump locally. The system is rebooted afterwards.
<b>reboot</b>	Reboot the system, losing the core dump in the process.
<b>halt</b>	Halt the system, losing the core dump in the process.
<b>poweroff</b>	Power off the system, losing the core dump in the process.
<b>shell</b>	Run a shell session from within the initramfs, allowing the user to record the core dump manually.

## Additional resources

- For detailed information on how to set up the default failure responses on the command line, see [Section 80.3.4, “Configuring the kdump default failure responses”](#).

### 80.5.6. Estimating kdump size

When planning and building your kdump environment, it is necessary to know how much space is required for the dump file before one is produced.

The **makedumpfile --mem-usage** command provides a useful report about excludable pages, and can be used to determine which dump level you want to assign. Run this command when the system is under representative load, otherwise **makedumpfile --mem-usage** returns a smaller value than is expected in your production environment.

TYPE	PAGES	EXCLUDABLE	DESCRIPTION
ZERO	501635	yes	Pages filled with zero
CACHE	51657	yes	Cache pages
CACHE_PRIVATE	5442	yes	Cache pages + private
USER	16301	yes	User process pages
FREE	77738211	yes	Free pages
KERN_DATA	1333192	no	Dumpable kernel data



#### IMPORTANT

The **makedumpfile --mem-usage** command reports in **pages**. This means that you have to calculate the size of memory in use against the kernel page size. By default the Red Hat Enterprise Linux kernel uses 4 KB sized pages for AMD64 and Intel 64 architectures, and 64 KB sized pages for IBM POWER architectures.

## 80.6. TESTING THE KDUMP CONFIGURATION

The following procedure describes how to test that the kernel dump process works and is valid before the machine enters production.



#### WARNING

The commands below cause the kernel to crash. Use caution when following these steps, and never carelessly use them on active production system.

#### Procedure

1. Reboot the system with **kdump** enabled.
2. Make sure that **kdump** is running:

```
~]# systemctl is-active kdump
active
```

3. Force the Linux kernel to crash:

```
echo 1 > /proc/sys/kernel/sysrq
echo c > /proc/sysrq-trigger
```



### WARNING

The command above crashes the kernel and a reboot is required.

Once booted again, the **address-YYYY-MM-DD-HH:MM:SS/vmcore** file is created at the location you have **specified** in **/etc/kdump.conf** (by default to **/var/crash/**).



### NOTE

In addition to confirming the validity of the configuration, it is possible to use this action to record how long it takes for a crash dump to complete, while a representative load was running.

## 80.7. USING KEXEC TO REBOOT THE KERNEL

The **kexec** system call enables loading and booting into another kernel from the currently running kernel, thus performing a function of a boot loader from within the kernel.

The **kexec** utility loads the kernel and the **initramfs** image for the **kexec** system call to boot into another kernel.

The following procedure describes how to manually invoke the **kexec** system call when using the **kexec** utility to reboot into another kernel.

### Procedure

1. Execute the **kexec** utility:

```
kexec -l /boot/vmlinuz-3.10.0-1040.el7.x86_64 --initrd=/boot/initramfs-3.10.0-1040.el7.x86_64.img --reuse-cmdline
```

The command manually loads the kernel and the initramfs image for the **kexec** system call.

2. Reboot the system:

```
reboot
```

The command detects the kernel, shuts down all services and then calls the **kexec** system call to reboot into the kernel you provided in the previous step.



### WARNING

When you use the **kexec -e** command to reboot the kernel, the system does not go through the standard shutdown sequence before starting the next kernel, which may cause data loss or an unresponsive system.

## 80.8. BLACKLISTING KERNEL DRIVERS FOR KDUMP

Blacklisting kernel drivers for kdump is a mechanism to prevent the intended kernel drivers from loading. Blacklisting kernel drivers prevents the **oom killer** or other crash kernel failures.

To blacklist the kernel drivers, you may update the **KDUMP\_COMMANDLINE\_APPEND=** variable in the **/etc/sysconfig/kdump** file and specify one of the following blacklisting option:

- **rd.driver.blacklist=<modules>**
- **modprobe.blacklist=<modules>**

When you blacklist drivers in **/etc/sysconfig/kdump** file, it prevents the **kdump initramfs** from loading the blacklisted modules.

The following procedure describes how to blacklist a kernel driver to prevent crash kernel failures.

### Procedure

1. Select the kernel module that you intend to blacklist:

```
$ lsmod

Module Size Used by
fuse 126976 3
xt_CHECKSUM 16384 1
ipt_MASQUERADE 16384 1
uinput 20480 1
xt_conntrack 16384 1
```

The **lsmod** command displays a list of modules that are loaded to the currently running kernel.

2. Update the **KDUMP\_COMMANDLINE\_APPEND=** line in the **/etc/sysconfig/kdump** file as follows:

```
KDUMP_COMMANDLINE_APPEND="rd.driver.blacklist=hv_vmbus,hv_storvsc,hv_utils,hv_netv
sc,hid-hyperv"
```

3. You can also update the **KDUMP\_COMMANDLINE\_APPEND=** line in the **/etc/sysconfig/kdump** file as follows:

```
KDUMP_COMMANDLINE_APPEND="modprobe.blacklist=emcp modprobe.blacklist=bnx2fc
modprobe.blacklist=libfcoe modprobe.blacklist=fcoe"
```

4. Restart the **kdump** service:

```
$ systemctl restart kdump
```

## Additional resources

- For more information concerning the **oom killer**, see the following [Knowledge Article](#).
- The **dracut.cmdline** manpage for modules blacklist options.

## 80.9. RUNNING KDUMP ON SYSTEMS WITH ENCRYPTED DISK

When running an encrypted partition created by the Logical Volume Manager (LVM) tool, systems require a certain amount of available memory. If the system has less than the required amount of available memory, the **cryptsetup** utility fails to mount the partition. As a result, capturing the **vmcore** file to a local **kdump** target location (with LVM and enabled encryption), fails in the second kernel (capture kernel).

This procedure describes the running **kdump** mechanism by increasing the **crashkernel=** value, using a remote **kdump** target, or using a key derivation function (KDF).

### Procedure

Run the **kdump** mechanism using one of the following procedures:

- To run the **kdump** define one of the following:
  - Configure a remote **kdump** target.
  - Define the dump to an unencrypted partition.
  - Specify an increased **crashkernel=** value to the required level.
- Add an extra key slot by using a key derivation function (KDF):
  1. **cryptsetup luksAddKey --pbkdf pbkdf2 /dev/vda2**
  2. **cryptsetup config --key-slot 1 --priority prefer /dev/vda2**
  3. **cryptsetup luksDump /dev/vda2**

Using the default KDF of the encrypted partition may consume a lot of memory. You must manually provide the password in the second kernel (capture), even if you encounter an Out of Memory (OOM) error message.



### WARNING

Adding an extra key slot can have a negative effect on security, as multiple keys can decrypt an encrypted volume. This may cause a potential risk to the volume.

## 80.10. ANALYZING A CORE DUMP

To determine the cause of the system crash, you can use the **crash** utility, which provides an interactive prompt very similar to the GNU Debugger (GDB). This utility allows you to interactively analyze a core dump created by **kdump**, **netdump**, **diskdump** or **xendump** as well as a running Linux system. Alternatively, you have the option to use the [Kdump Helper](#) or [Kernel Oops Analyzer](#).

### 80.10.1. Installing the crash utility

The following procedure describes how to install the **crash** analyzing tool.

#### Procedure

1. Enable the relevant **baseos** and **appstream** repositories:

```
subscription-manager repos --enable baseos repository
subscription-manager repos --enable appstream repository
```

2. Install the **crash** package:

```
yum install crash
```

3. Install the **kernel-debuginfo** package:

```
yum install kernel-debuginfo
```

The package corresponds to your running kernel and provides the data necessary for the dump analysis.

#### Additional resources

- For more information about how to work with repositories using the **subscription-manager** utility, see [Configuring basic system settings](#).

### 80.10.2. Running and exiting the crash utility

The following procedure describes how to start the crash utility for analyzing the cause of the system crash.

#### Prerequisites

- Identify the currently running kernel (for example **4.18.0-5.el8.x86\_64**).

#### Procedure

1. To start the **crash** utility, two necessary parameters need to be passed to the command:

- The debug-info (a decompressed vmlinuz image), for example **/usr/lib/debug/lib/modules/4.18.0-5.el8.x86\_64/vmlinuz** provided through a specific **kernel-debuginfo** package.
  - The actual vmcore file, for example **/var/crash/127.0.0.1-2018-10-06-14:05:33/vmcore**
- The resulting **crash** command then looks like this:

```
crash /usr/lib/debug/lib/modules/4.18.0-5.el8.x86_64/vmlinux /var/crash/127.0.0.1-2018-10-06-14:05:33/vmcore
```

Use the same <kernel> version that was captured by **kdump**.

### Example 80.1. Running the crash utility

The following example shows analyzing a core dump created on October 6 2018 at 14:05 PM, using the 4.18.0-5.el8.x86\_64 kernel.

```
...
WARNING: kernel relocated [202MB]: patching 90160 gdb minimal_symbol values

KERNEL: /usr/lib/debug/lib/modules/4.18.0-5.el8.x86_64/vmlinux
DUMPFILE: /var/crash/127.0.0.1-2018-10-06-14:05:33/vmcore [PARTIAL DUMP]
CPUS: 2
DATE: Sat Oct 6 14:05:16 2018
UPTIME: 01:03:57
LOAD AVERAGE: 0.00, 0.00, 0.00
TASKS: 586
NODENAME: localhost.localdomain
RELEASE: 4.18.0-5.el8.x86_64
VERSION: #1 SMP Wed Aug 29 11:51:55 UTC 2018
MACHINE: x86_64 (2904 Mhz)
MEMORY: 2.9 GB
PANIC: "sysrq: SysRq : Trigger a crash"
PID: 10635
COMMAND: "bash"
TASK: ffff8d6c84271800 [THREAD_INFO: ffff8d6c84271800]
CPU: 1
STATE: TASK_RUNNING (SYSRQ)

crash>
```

2. To exit the interactive prompt and terminate **crash**, type **exit** or **q**.

### Example 80.2. Exiting the crash utility

```
crash> exit
~]#
```



#### NOTE

The **crash** command can also be used as a powerful tool for debugging a live system. However use it with caution so as not to break your system.

### 80.10.3. Displaying various indicators in the crash utility

The following procedures describe how to use the crash utility and display various indicators, such as a kernel message buffer, a backtrace, a process status, virtual memory information and open files.

#### Displaying the message buffer

- To display the kernel message buffer, type the **log** command at the interactive prompt as displayed in the example below:

```
crash> log
... several lines omitted ...
EIP: 0060:[<c068124f>] EFLAGS: 00010096 CPU: 2
EIP is at sysrq_handle_crash+0xf/0x20
EAX: 00000063 EBX: 00000063 ECX: c09e1c8c EDX: 00000000
ESI: c0a09ca0 EDI: 00000286 EBP: 00000000 ESP: ef4dbf24
DS: 007b ES: 007b FS: 00d8 GS: 00e0 SS: 0068
Process bash (pid: 5591, ti=ef4da000 task=f196d560 task.ti=ef4da000)
Stack:
c068146b c0960891 c0968653 00000003 00000000 00000002 efade5c0 c06814d0
<0> ffffffb c068150f b7776000 f2600c40 c0569ec4 ef4dbf9c 00000002 b7776000
<0> efade5c0 00000002 b7776000 c0569e60 c051de50 ef4dbf9c f196d560 ef4dbfb4
Call Trace:
[<c068146b>] ? __handle_sysrq+0xfb/0x160
[<c06814d0>] ? write_sysrq_trigger+0x0/0x50
[<c068150f>] ? write_sysrq_trigger+0x3f/0x50
[<c0569ec4>] ? proc_reg_write+0x64/0xa0
[<c0569e60>] ? proc_reg_write+0x0/0xa0
[<c051de50>] ? vfs_write+0xa0/0x190
[<c051e8d1>] ? sys_write+0x41/0x70
[<c0409adc>] ? syscall_call+0x7/0xb
Code: a0 c0 01 0f b6 41 03 19 d2 f7 d2 83 e2 03 83 e0 cf c1 e2 04 09 d0 88 41 03 f3 c3 90 c7 05
c8 1b 9e c0 01 00 00 00 0f ae f8 89 f6 <c6> 05 00 00 00 00 01 c3 89 f6 8d bc 27 00 00 00 00 8d 50
d0 83
EIP: [<c068124f>] sysrq_handle_crash+0xf/0x20 SS:ESP 0068:ef4dbf24
CR2: 0000000000000000
```

Type **help log** for more information on the command usage.



### NOTE

The kernel message buffer includes the most essential information about the system crash and, as such, it is always dumped first in to the **vmcore-dmesg.txt** file. This is useful when an attempt to get the full **vmcore** file failed, for example because of lack of space on the target location. By default, **vmcore-dmesg.txt** is located in the **/var/crash/** directory.

## Displaying a backtrace

- To display the kernel stack trace, use the **bt** command.

```
crash> bt
PID: 5591 TASK: f196d560 CPU: 2 COMMAND: "bash"
#0 [ef4dbdcc] crash_kexec at c0494922
#1 [ef4dbe20] oops_end at c080e402
#2 [ef4dbe34] no_context at c043089d
#3 [ef4dbe58] bad_area at c0430b26
#4 [ef4dbe6c] do_page_fault at c080fb9b
#5 [ef4dbe4] error_code (via page_fault) at c080d809
EAX: 00000063 EBX: 00000063 ECX: c09e1c8c EDX: 00000000 EBP: 00000000
DS: 007b ESI: c0a09ca0 ES: 007b EDI: 00000286 GS: 00e0
```

```

CS: 0060 EIP: c068124f ERR: ffffffff EFLAGS: 00010096
#6 [ef4dbf18] sysrq_handle_crash at c068124f
#7 [ef4dbf24] __handle_sysrq at c0681469
#8 [ef4dbf48] write_sysrq_trigger at c068150a
#9 [ef4dbf54] proc_reg_write at c0569ec2
#10 [ef4dbf74] vfs_write at c051de4e
#11 [ef4dbf94] sys_write at c051e8cc
#12 [ef4dbfb0] system_call at c0409ad5
EAX: fffffda EBX: 00000001 ECX: b7776000 EDX: 00000002
DS: 007b ESI: 00000002 ES: 007b EDI: b7776000
SS: 007b ESP: bfcb2088 EBP: bfcb20b4 GS: 0033
CS: 0073 EIP: 00edc416 ERR: 00000004 EFLAGS: 00000246

```

Type **bt <pid>** to display the backtrace of a specific process or type **help bt** for more information on **bt** usage.

### Displaying a process status

- To display the status of processes in the system, use the **ps** command.

```

crash> ps
 PID PPID CPU TASK ST %MEM VSZ RSS COMM
> 0 0 0 c09dc560 RU 0.0 0 0 [swapper]
> 0 0 1 f7072030 RU 0.0 0 0 [swapper]
0 0 2 f70a3a90 RU 0.0 0 0 [swapper]
> 0 0 3 f70ac560 RU 0.0 0 0 [swapper]
1 0 1 f705ba90 IN 0.0 2828 1424 init
... several lines omitted ...
5566 1 1 f2592560 IN 0.0 12876 784 auditd
5567 1 2 ef427560 IN 0.0 12876 784 auditd
5587 5132 0 f196d030 IN 0.0 11064 3184 sshd
> 5591 5587 2 f196d560 RU 0.0 5084 1648 bash

```

Use **ps <pid>** to display the status of a single specific process. Use **help ps** for more information on **ps** usage.

### Displaying virtual memory information

- To display basic virtual memory information, type the **vm** command at the interactive prompt.

```

crash> vm
PID: 5591 TASK: f196d560 CPU: 2 COMMAND: "bash"
MM PGD RSS TOTAL_V
f19b5900 ef9c6000 1648k 5084k
VMA START END FLAGS FILE
f1bb0310 242000 260000 8000875 /lib/ld-2.12.so
f26af0b8 260000 261000 8100871 /lib/ld-2.12.so
efbc275c 261000 262000 8100873 /lib/ld-2.12.so
efbc2a18 268000 3ed000 8000075 /lib/libc-2.12.so
efbc23d8 3ed000 3ee000 8000070 /lib/libc-2.12.so
efbc2888 3ee000 3f0000 8100071 /lib/libc-2.12.so
efbc2cd4 3f0000 3f1000 8100073 /lib/libc-2.12.so
efbc243c 3f1000 3f4000 100073
efbc28ec 3f6000 3f9000 8000075 /lib/libdl-2.12.so

```

```

efbc2568 3f9000 3fa000 8100071 /lib/libdl-2.12.so
efbc2f2c 3fa000 3fb000 8100073 /lib/libdl-2.12.so
f26af888 7e6000 7fc000 8000075 /lib/libtinfo.so.5.7
f26aff2c 7fc000 7ff000 8100073 /lib/libtinfo.so.5.7
efbc211c d83000 d8f000 8000075 /lib/libnss_files-2.12.so
efbc2504 d8f000 d90000 8100071 /lib/libnss_files-2.12.so
efbc2950 d90000 d91000 8100073 /lib/libnss_files-2.12.so
f26afe00 edc000 edd000 4040075
f1bb0a18 8047000 8118000 8001875 /bin/bash
f1bb01e4 8118000 811d000 8101873 /bin/bash
f1bb0c70 811d000 8122000 100073
f26afae0 9fd9000 9ffa000 100073
... several lines omitted ...

```

Use **vm <pid>** to display information on a single specific process, or use **help vm** for more information on **vm** usage.

### Displaying open files

- To display information about open files, use the **files** command.

```

crash> files
PID: 5591 TASK: f196d560 CPU: 2 COMMAND: "bash"
ROOT: / CWD: /root
FD FILE DENTRY INODE TYPE PATH
0 f734f640 eedc2c6c eecd6048 CHR /pts/0
1 efade5c0 eee14090 f00431d4 REG /proc/sysrq-trigger
2 f734f640 eedc2c6c eecd6048 CHR /pts/0
10 f734f640 eedc2c6c eecd6048 CHR /pts/0
255 f734f640 eedc2c6c eecd6048 CHR /pts/0

```

Use **files <pid>** to display files opened by only one selected process, or use **help files** for more information on **files** usage.

### 80.10.4. Using Kernel Oops Analyzer

The Kernel Oops Analyzer is a tool that analyzes the crash dump by comparing the oops messages with known issues in the knowledge base.

#### Prerequisites

- Secure an oops message to feed the Kernel Oops Analyzer by following instructions in [Red Hat Labs](#).

#### Procedure

1. Follow the [Kernel Oops Analyzer](#) link to access the tool.
2. Browse for the oops message by hitting the **Browse** button.

## Data Input

### File Input

### Text Input

Choose and upload the [kernel oops log](#) generated from a vmcore.

[Browse...](#)

No file selected.

Maximum file size for uploaded kernel oops log is 10 MB.

**DETECT**

3. Click the **DETECT** button to compare the oops message based on information from [makedumpfile](#) against known solutions.

## 80.11. USING EARLY KDUMP TO CAPTURE BOOT TIME CRASHES

As a system administrator, you can utilize the **early kdump** support of the **kdump** service to capture a vmcore file of the crashing kernel during the early stages of the booting process. This section describes what **early kdump** is, how to configure it, and how to check the status of this mechanism.

### 80.11.1. What is early kdump

Kernel crashes during the booting phase occur when the **kdump** service is not yet started, and cannot facilitate capturing and saving the contents of the crashed kernel's memory. Therefore, the vital information for troubleshooting is lost.

To address this problem, RHEL 8 introduced the **early kdump** feature as a part of the **kdump** service.

#### Additional resources

- For more information about **early kdump** and its use, see the [/usr/share/doc/kexec-tools/early-kdump-howto.txt](#) file and [What is early kdump support and how do I configure it?](#) solution.
- For more information about the **kdump** service, see the [Section 80.1, "What is kdump"](#).

### 80.11.2. Enabling early kdump

This section describes how to enable the **early kdump** feature to eliminate the risk of losing information about the early boot kernel crashes.

#### Prerequisites

- An active Red Hat Enterprise Linux subscription
- A repository containing the **kexec-tools** package for your system CPU architecture
- Fulfilled [kdump requirements](#)

#### Procedure

1. Verify that the **kdump** service is enabled and active:

```
systemctl is-enabled kdump.service && systemctl is-active kdump.service
enabled
active
```

If **kdump** is not enabled and running see, [Section 80.3.5, “Enabling and disabling the kdump service”](#).

2. Rebuild the **initramfs** image of the booting kernel with the **early kdump** functionality:

```
dracut -f --add earlykdump
```

3. Add the **rd.earlykdump** kernel command line parameter:

```
grubby --update-kernel=/boot/vmlinuz-$(uname -r) --args="rd.earlykdump"
```

4. Reboot

5. Optionally, verify that **rd.earlykdump** was successfully added and **early kdump** feature was enabled:

```
cat /proc/cmdline
BOOT_IMAGE=(hd0,msdos1)/vmlinuz-4.18.0-187.el8.x86_64 root=/dev/mapper/rhel-root ro
crashkernel=auto resume=/dev/mapper/rhel-swap rd.lvm.lv=rhel/root rd.lvm.lv=rhel/swap
rhgb quiet rd.earlykdump

journalctl -x | grep early-kdump
Mar 20 15:44:41 redhat dracut-cmdline[304]: early-kdump is enabled.
Mar 20 15:44:42 redhat dracut-cmdline[304]: kexec: loaded early-kdump kernel
```

## Additional resources

- For more information on enabling **early kdump**, see the [/usr/share/doc/kexec-tools/early-kdump-howto.txt](#) file and [What is early kdump support and how do I configure it?](#) solution.

## 80.12. RELATED INFORMATION

The following section provides further information related to capturing crash information.

- **kdump.conf(5)** – a manual page for the **/etc/kdump.conf** configuration file containing the full documentation of available options.
- **zipl.conf(5)** – a manual page for the **/etc/zipl.conf** configuration file.
- **zipl(8)** – a manual page for the **zipl** boot loader utility for IBM System z.
- **makedumpfile(8)** – a manual page for the **makedumpfile** core collector.
- **kexec(8)** – a manual page for **kexec**.
- **crash(8)** – a manual page for the **crash** utility.

- **/usr/share/doc/kexec-tools/kexec-kdump-howto.txt** – an overview of the **kdump** and **kexec** installation and usage.
- For more information about the **kexec** and **kdump** configuration see the [Red Hat Knowledgebase article](#).
- For more information about the supported **kdump** targets see the [Red Hat Knowledgebase article](#).

include::assemblies/assembly\_applying-patches-with-kernel-live-patching.adoc :leveloffset: +1

## PART VII. SETTING LIMITS FOR APPLICATIONS

As a system administrator, use the control groups kernel functionality to set limits, prioritize or isolate the hardware resources of processes so that applications on your system are stable and do not run out of memory.

# CHAPTER 81. UNDERSTANDING CONTROL GROUPS

*Control groups* is a Linux kernel feature that enables you to organize processes into hierarchically ordered groups – **cgroups**. The hierarchy (control groups tree) is defined by providing structure to **cgroups** virtual file system, mounted by default on the **/sys/fs/cgroup/** directory. It is done manually by creating and removing sub-directories in **/sys/fs/cgroup/**. Alternatively, by using the **systemd** system and service manager.

The resource controllers (a kernel component) then modify the behavior of processes in **cgroups** by limiting, prioritizing or allocating system resources, (such as CPU time, memory, network bandwidth, or various combinations) of those processes.

The added value of **cgroups** is process aggregation which enables division of hardware resources among applications and users. Thereby an increase in overall efficiency, stability and security of users' environment can be achieved.

## Control groups version 1

*Control groups version 1 (cgroups-v1)* provide a per-resource controller hierarchy. It means that each resource, such as CPU, memory, I/O, and so on, has its own control group hierarchy. It is possible to combine different control group hierarchies in a way that one controller can coordinate with another one in managing their respective resources. However, the two controllers may belong to different process hierarchies, which does not permit their proper coordination.

The **cgroups-v1** controllers were developed across a large time span and as a result, the behavior and naming of their control files is not uniform.

## Control groups version 2

The problems with controller coordination, which stemmed from hierarchy flexibility, led to the development of *control groups version 2*.

*Control groups version 2 (cgroups-v2)* provides a single control group hierarchy against which all resource controllers are mounted.

The control file behavior and naming is consistent among different controllers.



### WARNING

RHEL 8 provides **cgroups-v2** as a technology preview with a limited number of resource controllers. For more information about the relevant resource controllers, see the [cgroups-v2 release note](#).

This sub-section was based on a Devconf.cz 2019 presentation.<sup>[3]</sup>

## Additional resources

- For more information about resource controllers, see [Chapter 82, What kernel resource controllers are](#) section and **cgroups(7)** manual pages.
- For more information about **cgroups** hierarchies and **cgroups** versions, refer to **cgroups(7)** manual pages.

- For more information about **systemd** and **cgroups** cooperation, see [Role of systemd in control groups](#) section.

---

[3] Linux Control Group v2 - An Introduction, Devconf.cz 2019 presentation by Waiman Long

## CHAPTER 82. WHAT KERNEL RESOURCE CONTROLLERS ARE

The functionality of control groups is enabled by kernel resource controllers. RHEL 8 supports various controllers for *control groups version 1* (**cgroups-v1**) and *control groups version 2* (**cgroups-v2**).

A resource controller, also called a control group subsystem, is a kernel subsystem that represents a single resource, such as CPU time, memory, network bandwidth or disk I/O. The Linux kernel provides a range of resource controllers that are mounted automatically by the **systemd** system and service manager. Find a list of currently mounted resource controllers in the **/proc/cgroups** file.

The following controllers are available for **cgroups-v1**:

- **blkio** - can set limits on input/output access to and from block devices.
- **cpu** - can adjust the parameters of the Completely Fair Scheduler (CFS) scheduler for control group's tasks. It is mounted together with the **cpuacct** controller on the same mount.
- **cpuacct** - creates automatic reports on CPU resources used by tasks in a control group. It is mounted together with the **cpu** controller on the same mount.
- **cpuset** - can be used to restrict control group tasks to run only on a specified subset of CPUs and to direct the tasks to use memory only on specified memory nodes.
- **devices** - can control access to devices for tasks in a control group.
- **freezer** - can be used to suspend or resume tasks in a control group.
- **memory** - can be used to set limits on memory use by tasks in a control group and generates automatic reports on memory resources used by those tasks.
- **net\_cls** - tags network packets with a class identifier (**classid**) that enables the Linux traffic controller (the **tc** command) to identify packets that originate from a particular control group task. A subsystem of **net\_cls**, the **net\_filter** (iptables), can also use this tag to perform actions on such packets. The **net\_filter** tags network sockets with a firewall identifier (**fwid**) that allows the Linux firewall (through **iptables** command) to identify packets originating from a particular control group task.
- **net\_prio** - sets the priority of network traffic.
- **pids** - can set limits for a number of processes and their children in a control group.
- **perf\_event** - can group tasks for monitoring by the **perf** performance monitoring and reporting utility.
- **rdma** - can set limits on Remote Direct Memory Access/InfiniBand specific resources in a control group.
- **hugetlb** - can be used to limit the usage of large size virtual memory pages by tasks in a control group.

The following controllers are available for **cgroups-v2**:

- **io** - A follow-up to **blkio** of **cgroups-v1**.
- **memory** - A follow-up to **memory** of **cgroups-v1**.
- **pids** - Same as **pids** in **cgroups-v1**.

- **rdma** - Same as **rdma** in **cgroups-v1**.
- **cpu** - A follow-up to **cpu** and **cpuacct** of **cgroups-v1**.
- **cpuset** - Supports only the core functionality (**cpus{,.effective}**, **mems{,.effective}**) with a new partition feature.
- **perf\_event** - Support is inherent, no explicit control file. You can specify a **v2 cgroup** as a parameter to the **perf** command that will profile all the tasks within that **cgroup**.



## IMPORTANT

A resource controller can be used either in a **cgroups-v1** hierarchy or a **cgroups-v2** hierarchy, not simultaneously in both.

## Additional resources

- For more information about resource controllers in general, refer to the **cgroups(7)** manual page.
- For detailed descriptions of specific resource controllers, see the documentation in the **/usr/share/doc/kernel-doc-<kernel\_version>/Documentation/cgroups-v1/** directory.
- For more information about **cgroups-v2**, refer to the **cgroups(7)** manual page.

# CHAPTER 83. USING CONTROL GROUPS THROUGH A VIRTUAL FILE SYSTEM

You can use *control groups* (**cgroups**) to set limits, prioritize, or control access to hardware resources for groups of processes. This allows you to granularly control resource usage of applications to utilize them more efficiently. The following sections provide an overview of tasks related to management of **cgroups** for both version 1 and version 2 using a virtual file system.

## 83.1. SETTING CPU LIMITS TO APPLICATIONS USING CGROUPS-V1

Sometimes an application consumes a lot of CPU time, which may negatively impact the overall health of your environment. Use the **/sys/fs** virtual file system to configure CPU limits to an application using *control groups version 1* (**cgroups-v1**).

### Prerequisites

- An application whose CPU consumption you want to restrict.
- Verify that the **cgroups-v1** controllers were mounted:

```
mount -l | grep cgroup
tmpfs on /sys/fs/cgroup type tmpfs (ro,nosuid,nodev,noexec,seclabel,mode=755)
cgroup on /sys/fs/cgroup/systemd type cgroup
(rw,nosuid,nodev,noexec,relatime,seclabel,xattr,release_agent=/usr/lib/systemd/systemd-
cgroups-agent,name=systemd)
cgroup on /sys/fs/cgroup/cpu,cpuacct type cgroup
(rw,nosuid,nodev,noexec,relatime,seclabel,cpu,cpuacct)
cgroup on /sys/fs/cgroup/perf_event type cgroup
(rw,nosuid,nodev,noexec,relatime,seclabel,perf_event)
cgroup on /sys/fs/cgroup/pids type cgroup (rw,nosuid,nodev,noexec,relatime,seclabel,pids)
...
...
```

### Procedure

1. Identify the process ID (PID) of the application you want to restrict in CPU consumption:

```
top
top - 11:34:09 up 11 min, 1 user, load average: 0.51, 0.27, 0.22
Tasks: 267 total, 3 running, 264 sleeping, 0 stopped, 0 zombie
%Cpu(s): 49.0 us, 3.3 sy, 0.0 ni, 47.5 id, 0.0 wa, 0.2 hi, 0.0 si, 0.0 st
MiB Mem : 1826.8 total, 303.4 free, 1046.8 used, 476.5 buff/cache
MiB Swap: 1536.0 total, 1396.0 free, 140.0 used, 616.4 avail Mem

 PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
 6955 root 20 0 228440 1752 1472 R 99.3 0.1 0:32.71 sha1sum
 5760 jdoe 20 0 3603868 205188 64196 S 3.7 11.0 0:17.19 gnome-shell
 6448 jdoe 20 0 743648 30640 19488 S 0.7 1.6 0:02.73 gnome-terminal-
 1 root 20 0 245300 6568 4116 S 0.3 0.4 0:01.87 systemd
 505 root 20 0 0 0 0 I 0.3 0.0 0:00.75 kworker/u4:4-events_unbound
...
...
```

The example output of the **top** program reveals that **PID 6955** (illustrative application **sha1sum**) consumes a lot of CPU resources.

2. Create a sub-directory in the **cpu** resource controller directory:

```
mkdir /sys/fs/cgroup/cpu/Example/
```

The directory above represents a control group, where you can place specific processes and apply certain CPU limits to the processes. At the same time, some **cgroups-v1** interface files and **cpu** controller-specific files will be created in the directory.

3. Optionally, inspect the newly created control group:

```
ll /sys/fs/cgroup/cpu/Example/
-rw-r--r--. 1 root root 0 Mar 11 11:42 cgroup.clone_children
-rw-r--r--. 1 root root 0 Mar 11 11:42 cgroup.procs
-r--r--r--. 1 root root 0 Mar 11 11:42 cpuacct.stat
-rw-r--r--. 1 root root 0 Mar 11 11:42 cpuacct.usage
-r--r--r--. 1 root root 0 Mar 11 11:42 cpuacct.usage_all
-r--r--r--. 1 root root 0 Mar 11 11:42 cpuacct.usage_percpu
-r--r--r--. 1 root root 0 Mar 11 11:42 cpuacct.usage_percpu_sys
-r--r--r--. 1 root root 0 Mar 11 11:42 cpuacct.usage_percpu_user
-r--r--r--. 1 root root 0 Mar 11 11:42 cpuacct.usage_sys
-r--r--r--. 1 root root 0 Mar 11 11:42 cpuacct.usage_user
-rw-r--r--. 1 root root 0 Mar 11 11:42 cpu.cfs_period_us
-rw-r--r--. 1 root root 0 Mar 11 11:42 cpu.cfs_quota_us
-rw-r--r--. 1 root root 0 Mar 11 11:42 cpu.rt_period_us
-rw-r--r--. 1 root root 0 Mar 11 11:42 cpu.rt_runtime_us
-rw-r--r--. 1 root root 0 Mar 11 11:42 cpu.shares
-r--r--r--. 1 root root 0 Mar 11 11:42 cpu.stat
-rw-r--r--. 1 root root 0 Mar 11 11:42 notify_on_release
-rw-r--r--. 1 root root 0 Mar 11 11:42 tasks
```

The example output shows files, such as **cpuacct.usage**, **cpu.cfs.\_period\_us**, that represent specific configurations and/or limits, which can be set for processes in the **Example** control group. Notice that the respective file names are prefixed with the name of the control group controller to which they belong.

By default, the newly created control group inherits access to the system's entire CPU resources without a limit.

4. Configure CPU limits for the control group:

```
echo "1000000" > /sys/fs/cgroup/cpu/Example/cpu.cfs_period_us
echo "200000" > /sys/fs/cgroup/cpu/Example/cpu.cfs_quota_us
```

The **cpu.cfs\_period\_us** file represents a period of time in microseconds ( $\mu$ s, represented here as "us") for how frequently a control group's access to CPU resources should be reallocated. The upper limit is 1 second and the lower limit is 1000 microseconds.

The **cpu.cfs\_quota\_us** file represents the total amount of time in microseconds for which all processes collectively in a control group can run during one period (as defined by **cpu.cfs\_period\_us**). As soon as processes in a control group, during a single period, use up all the time specified by the quota, they are throttled for the remainder of the period and not allowed to run until the next period. The lower limit is 1000 microseconds.

The example commands above set the CPU time limits so that all processes collectively in the **Example** control group will be able to run only for 0.2 seconds (defined by **cpu.cfs\_quota\_us**) out of every 1 second (defined by **cpu.cfs\_period\_us**).

5. Optionally, verify the limits:

```
cat /sys/fs/cgroup/cpu/Example/cpu.cfs_period_us
/sys/fs/cgroup/cpu/Example/cpu.cfs_quota_us
1000000
200000
```

6. Add the application's PID to the **Example** control group:

```
echo "6955" > /sys/fs/cgroup/cpu/Example/cgroup.procs
or
echo "6955" > /sys/fs/cgroup/cpu/Example/tasks
```

The previous command ensures that a desired application becomes a member of the **Example** control group and hence does not exceed the CPU limits configured for the **Example** control group. The PID should represent an existing process in the system. The **PID 6955** here was assigned to process **sha1sum /dev/zero &**, used to illustrate the use-case of the **cpu** controller.

7. Verify that the application runs in the specified control group:

```
cat /proc/6955/cgroup
12:cpuset:/
11:hugetlb:/
10:net_cls,net_prio:/
9:memory:/user.slice/user-1000.slice/user@1000.service
8:devices:/user.slice
7:blkio:/
6:freezer:/
5:rdma:/
4:pids:/user.slice/user-1000.slice/user@1000.service
3:perf_event:/
2:cpu,cpuacct:/Example
1:name=systemd:/user.slice/user-1000.slice/user@1000.service/gnome-terminal-
server.service
```

The example output above shows that the process of the desired application runs in the **Example** control group, which applies CPU limits to the application's process.

8. Identify the current CPU consumption of your throttled application:

```
top
top - 12:28:42 up 1:06, 1 user, load average: 1.02, 1.02, 1.00
Tasks: 266 total, 6 running, 260 sleeping, 0 stopped, 0 zombie
%Cpu(s): 11.0 us, 1.2 sy, 0.0 ni, 87.5 id, 0.0 wa, 0.2 hi, 0.0 si, 0.2 st
MiB Mem : 1826.8 total, 287.1 free, 1054.4 used, 485.3 buff/cache
MiB Swap: 1536.0 total, 1396.7 free, 139.2 used. 608.3 avail Mem

 PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
 6955 root 20 0 228440 1752 1472 R 20.6 0.1 47:11.43 sha1sum
 5760 jdoe 20 0 3604956 208832 65316 R 2.3 11.2 0:43.50 gnome-shell
 6448 jdoe 20 0 743836 31736 19488 S 0.7 1.7 0:08.25 gnome-terminal-
```

```

505 root 20 0 0 0 0 1 0.3 0.0 0:03.39 kworker/u4:4-events_unbound
4217 root 20 0 74192 1612 1320 S 0.3 0.1 0:01.19 spice-vdagentd
...

```

Notice that the CPU consumption of the **PID 6955** has decreased from 99% to 20%.

## Additional resources

- For information about the control groups concept, see [Chapter 81, Understanding control groups](#).
- For more information about resource controllers, see the [Chapter 82, What kernel resource controllers are](#) section and the **cgroups(7)** manual page.
- For more information about the **/sys/fs/** virtual filesystem, see the **sysfs(5)** manual page.

## 83.2. SETTING CPU LIMITS TO APPLICATIONS USING CGROUPS-V2

Sometimes an application uses a lot of CPU time, which may negatively impact the overall health of your environment. Use *control groups version 2 (cgroups-v2)* to configure CPU limits to the application, and restrict that its consumption.

### Prerequisites

- An application whose CPU consumption you want to restrict.
- [Chapter 81, Understanding control groups](#)

### Procedure

1. Prevent **cgroups-v1** from automatically mounting during the system boot:

```
grub2 --update-kernel=/boot/vmlinuz-$(uname -r) --args="cgroup_no_v1=all"
```

The command adds a kernel command-line parameter to the current boot entry. The **cgroup\_no\_v1=all** parameter prevents **cgroups-v1** from being automatically mounted.

Alternatively, use the **systemd.unified\_cgroup\_hierarchy=1** kernel command-line parameter to mount **cgroups-v2** during the system boot by default.



### NOTE

RHEL 8 supports both **cgroups-v1** and **cgroups-v2**. However, **cgroups-v1** is enabled and mounted by default during the booting process.

2. Reboot the system for the changes to take effect.
3. Optionally, verify the **cgroups-v1** functionality has been disabled:

```
mount -l | grep cgroup
tmpfs on /sys/fs/cgroup type tmpfs (ro,nosuid,nodev,noexec,seclabel,mode=755)
cgroup on /sys/fs/cgroup/systemd type cgroup
(rw,nosuid,nodev,noexec,relatime,seclabel,xattr,release_agent=/usr/lib/systemd/systemd-cgroups-agent,name=systemd)
```

If **cgroups-v1** have been successfully disabled, the output does not show any "type cgroup" references, except for those which belong to **systemd**.

4. Mount **cgroups-v2** anywhere in the filesystem:

```
mount -t cgroup2 none <MOUNT_POINT>
```

5. Optionally, verify the **cgroups-v2** functionality has been mounted:

```
mount -l | grep cgroup2
none on /cgroups-v2 type cgroup2 (rw,relatime,seclabel)
```

The example output shows that **cgroups-v2** has been mounted to the **/cgroups-v2** directory.

6. Optionally, inspect the contents of the **/cgroups-v2** directory:

```
ll /cgroups-v2/
-r--r--. 1 root root 0 Mar 13 11:57 cgroup.controllers
-rw-r--. 1 root root 0 Mar 13 11:57 cgroup.max.depth
-rw-r--. 1 root root 0 Mar 13 11:57 cgroup.max.descendants
-rw-r--. 1 root root 0 Mar 13 11:57 cgroup.procs
-r--r--. 1 root root 0 Mar 13 11:57 cgroup.stat
-rw-r--. 1 root root 0 Mar 13 11:58 cgroup.subtree_control
-rw-r--. 1 root root 0 Mar 13 11:57 cgroup.threads
-rw-r--. 1 root root 0 Mar 13 11:57 cpu.pressure
-r--r--. 1 root root 0 Mar 13 11:57 cpuset.cpus.effective
-r--r--. 1 root root 0 Mar 13 11:57 cpuset.mems.effective
-rw-r--. 1 root root 0 Mar 13 11:57 io.pressure
-rw-r--. 1 root root 0 Mar 13 11:57 memory.pressure
```

The **/cgroups-v2** directory, also called the root control group, contains some interface files (starting with **cgroup**) and some controller-specific files such as **cpuset.cpus.effective**.

7. Identify the process IDs (PIDs) of applications you want to restrict in CPU consumption:

```
top
top - 15:39:52 up 3:45, 1 user, load average: 0.79, 0.20, 0.07
Tasks: 265 total, 3 running, 262 sleeping, 0 stopped, 0 zombie
%Cpu(s): 74.3 us, 6.1 sy, 0.0 ni, 19.4 id, 0.0 wa, 0.2 hi, 0.0 si, 0.0 st
MiB Mem : 1826.8 total, 243.8 free, 1102.1 used, 480.9 buff/cache
MiB Swap: 1536.0 total, 1526.2 free, 9.8 used. 565.6 avail Mem

 PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
 5473 root 20 0 228440 1740 1456 R 99.7 0.1 0:12.11 sha1sum
 5439 root 20 0 222616 3420 3052 R 60.5 0.2 0:27.08 cpu_load_generator
 2170 jdoe 20 0 3600716 209960 67548 S 0.3 11.2 1:18.50 gnome-shell
 3051 root 20 0 274424 3976 3092 R 0.3 0.2 1:01.25 top
 1 root 20 0 245448 10256 5448 S 0.0 0.5 0:02.52 systemd
 ...
...
```

The example output of the **top** program reveals that **PID 5473** and **5439** (illustrative application **sha1sum** and **cpu\_load\_generator**) consume a lot of resources, namely CPU. Both are example applications used to demonstrate managing the **cgroups-v2** functionality.

8. Enable CPU-related controllers:

```
echo "+cpu" > /cgroups-v2/cgroup.subtree_control
echo "+cpuset" > /cgroups-v2/cgroup.subtree_control
```

The previous commands enable the **cpu** and **cpuset** controllers for the immediate sub-control groups of the **/cgroups-v2/** root control group.

9. Create a sub-directory in the previously created **/cgroups-v2/** directory:

```
mkdir /cgroups-v2/Example/
```

The **/cgroups-v2/Example/** directory represents a sub-control group, where you can place specific processes and apply various CPU limits to the processes. Also, the previous step enabled the **cpu** and **cpuset** controllers for this sub-control group.

At the time of creation of **/cgroups-v2/Example/**, some **cgroups-v2** interface files and **cpu** and **cpuset** controller-specific files will be created in the directory.

10. Optionally, inspect the newly created control group:

```
ll /cgroups-v2/Example/
-r--r--. 1 root root 0 Mar 13 14:48 cgroup.controllers
-r--r--. 1 root root 0 Mar 13 14:48 cgroup.events
-rw-r--. 1 root root 0 Mar 13 14:48 cgroup.freeze
-rw-r--. 1 root root 0 Mar 13 14:48 cgroup.max.depth
-rw-r--. 1 root root 0 Mar 13 14:48 cgroup.max.descendants
-rw-r--. 1 root root 0 Mar 13 14:48 cgroup.procs
-r--r--. 1 root root 0 Mar 13 14:48 cgroup.stat
-rw-r--. 1 root root 0 Mar 13 14:48 cgroup.subtree_control
-rw-r--. 1 root root 0 Mar 13 14:48 cgroup.threads
-rw-r--. 1 root root 0 Mar 13 14:48 cgroup.type
-rw-r--. 1 root root 0 Mar 13 14:48 cpu.max
-rw-r--. 1 root root 0 Mar 13 14:48 cpu.pressure
-rw-r--. 1 root root 0 Mar 13 14:48 cpuset.cpus
-r--r--. 1 root root 0 Mar 13 14:48 cpuset.cpus.effective
-rw-r--. 1 root root 0 Mar 13 14:48 cpuset.cpus.partition
-rw-r--. 1 root root 0 Mar 13 14:48 cpuset.mems
-r--r--. 1 root root 0 Mar 13 14:48 cpuset.mems.effective
-r--r--. 1 root root 0 Mar 13 14:48 cpu.stat
-rw-r--. 1 root root 0 Mar 13 14:48 cpu.weight
-rw-r--. 1 root root 0 Mar 13 14:48 cpu.weight.nice
-rw-r--. 1 root root 0 Mar 13 14:48 io.pressure
-rw-r--. 1 root root 0 Mar 13 14:48 memory.pressure
```

The example output shows files such as **cpuset.cpus** and **cpu.max**. The files are specific to the **cpuset** and **cpu** controllers that you enabled for the root's (**/cgroups-v2/**) direct child control groups using the **/cgroups-v2/cgroup.subtree\_control** file. Also, there are general **cgroup** control interface files such as **cgroup.procs** or **cgroup.controllers**, which are common to all control groups, regardless of enabled controllers.

By default, the newly created sub-control group inherited access to the system's entire CPU resources without a limit.

11. Ensure the processes that you want to limit compete for CPU time on the same CPU:

```
echo "1" > /cgroups-v2/Example/cpuset.cpus
```

The previous command secures processes that you placed in the **Example** sub-control group, compete on the same CPU. This setting is important for the **cpu** controller to activate.



## IMPORTANT

The **cpu** controller is only activated if the relevant sub-control group has at least 2 processes, which compete for time on a single CPU.

12. Configure CPU limits of the control group:

```
echo "200000 1000000" > /cgroups-v2/Example/cpu.max
```

The first value is the allowed time quota in microseconds for which all processes collectively in a sub-control group can run during one period (specified by the second value). During a single period, when processes in a control group collectively exhaust all the time specified by this quota, they are throttled for the remainder of the period and not allowed to run until the next period.

The example command sets the CPU time limits so that all processes collectively in the **Example** sub-control group are able to run on the CPU only for 0.2 seconds out of every 1 second.

13. Optionally, verify the limits:

```
cat /cgroups-v2/Example/cpu.max
200000 1000000
```

14. Add the applications' PIDs to the **Example** sub-control group:

```
echo "5473" > /cgroups-v2/Example/cgroup.procs
echo "5439" > /cgroups-v2/Example/cgroup.procs
```

The example commands ensure that desired applications become members of the **Example** sub-control group and hence do not exceed the CPU limits configured for the **Example** sub-control group.

15. Verify that the applications run in the specified control group:

```
cat /proc/5473/cgroup /proc/5439/cgroup
1:name=systemd:/user.slice/user-1000.slice/user@1000.service/gnome-terminal-
server.service
0:/Example
1:name=systemd:/user.slice/user-1000.slice/user@1000.service/gnome-terminal-
server.service
0:/Example
```

The example output above shows that the processes of the desired applications run in the **Example** sub-control group.

16. Inspect the current CPU consumption of your throttled applications:

```
top
top - 15:56:27 up 4:02, 1 user, load average: 0.03, 0.41, 0.55
Tasks: 265 total, 4 running, 261 sleeping, 0 stopped, 0 zombie
```

```
%Cpu(s): 9.6 us, 0.8 sy, 0.0 ni, 89.4 id, 0.0 wa, 0.2 hi, 0.0 si, 0.0 st
MiB Mem : 1826.8 total, 243.4 free, 1102.1 used, 481.3 buff/cache
MiB Swap: 1536.0 total, 1526.2 free, 9.8 used. 565.5 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
5439	root	20	0	222616	3420	3052	R	10.0	0.2	6:15.83	cpu_load_generator
5473	root	20	0	228440	1740	1456	R	10.0	0.1	9:20.65	sha1sum
2753	jdoe	20	0	743928	35328	20608	S	0.7	1.9	0:20.36	gnome-terminal
2170	jdoe	20	0	3599688	208820	67552	S	0.3	11.2	1:33.06	gnome-shell
5934	root	20	0	274428	5064	4176	R	0.3	0.3	0:00.04	top
...											

Notice that the CPU consumption for the **PID 5439** and **PID 5473** has decreased to 10%. The **Example** sub-control group limits its processes to 20% of the CPU time collectively. Since there are 2 processes in the control group, each can utilize 10% of the CPU time.

## Additional resources

- For information about the control groups concept, see [Chapter 81, Understanding control groups](#).
- For more information about resource controllers, see the [Chapter 82, What kernel resource controllers are](#) section and the **cgroups(7)** manual page.
- For more information about the **/sys/fs** virtual filesystem, see the **sysfs(5)** manual page.

# CHAPTER 84. ROLE OF SYSTEMD IN CONTROL GROUPS

## VERSION 1

Red Hat Enterprise Linux 8 moves the resource management settings from the process level to the application level by binding the system of **cgroup** hierarchies with the **systemd** unit tree. Therefore, you can manage the system resources with the **systemctl** command, or by modifying the **systemd** unit files.

By default, the **systemd** system and service manager makes use of the **slice**, the **scope** and the **service** units to organize and structure processes in the control groups. The **systemctl** command enables you to further modify this structure by creating custom **slices**. Also, **systemd** automatically mounts hierarchies for important kernel resource controllers in the **/sys/fs/cgroup/** directory.

Three **systemd** unit types are used for resource control:

- **Service** - A process or a group of processes, which **systemd** started according to a unit configuration file. Services encapsulate the specified processes so that they can be started and stopped as one set. Services are named in the following way:

**<name>.service**

- **Scope** - A group of externally created processes. Scopes encapsulate processes that are started and stopped by the arbitrary processes through the **fork()** function and then registered by **systemd** at runtime. For example, user sessions, containers, and virtual machines are treated as scopes. Scopes are named as follows:

**<name>.scope**

- **Slice** - A group of hierarchically organized units. Slices organize a hierarchy in which scopes and services are placed. The actual processes are contained in scopes or in services. Every name of a slice unit corresponds to the path to a location in the hierarchy. The dash ("–") character acts as a separator of the path components to a slice from the **-.slice** root slice. In the following example:

**<parent-name>.slice**

**parent-name.slice** is a sub-slice of **parent.slice**, which is a sub-slice of the **-.slice** root slice. **parent-name.slice** can have its own sub-slice named **parent-name-name2.slice**, and so on.

The **service**, the **scope**, and the **slice** units directly map to objects in the control group hierarchy. When these units are activated, they map directly to control group paths built from the unit names.

The following is an abbreviated example of a control group hierarchy:

```

Control group /:
-.slice
└── user.slice
 ├── user-42.slice
 │ ├── session-c1.scope
 │ │ ├── 967 gdm-session-worker [pam/gdm-launch-environment]
 │ │ └── 1035 /usr/libexec/gdm-x-session gnome-session --autostart
 │ └── /usr/share/gdm/greeter/autostart
 └── /usr/share/gdm/greeter/autostart
 ├── 1054 /usr/libexec/Xorg vt1 -displayfd 3 -auth /run/user/42/gdm/Xauthority -background none
 └── -noreset -keeptty -verbose 3
 └── 1212 /usr/libexec/gnome-session-binary --autostart /usr/share/gdm/greeter/autostart

```

```

 └─1369 /usr/bin/gnome-shell
 └─1732 ibus-daemon --xim --panel disable
 └─1752 /usr/libexec/ibus-dconf
 └─1762 /usr/libexec/ibus-x11 --kill-daemon
 └─1912 /usr/libexec/gsd-xsettings
 └─1917 /usr/libexec/gsd-a11y-settings
 └─1920 /usr/libexec/gsd-clipboard
...
└─init.scope
 └─1 /usr/lib/systemd/systemd --switched-root --system --deserialize 18
└─system.slice
 ├─rngd.service
 | └─800 /sbin/rngd -f
 ├─systemd-udevd.service
 | └─659 /usr/lib/systemd/systemd-udevd
 ├─chronyd.service
 | └─823 /usr/sbin/chronyd
 ├─audited.service
 | ├─761 /sbin/audited
 | └─763 /usr/sbin/sedispatch
 ├─accounts-daemon.service
 | └─876 /usr/libexec/accounts-daemon
 ├─example.service
 | └─929 /bin/bash /home/jdoe/example.sh
 | └─4902 sleep 1
...

```

The example above shows that services and scopes contain processes and are placed in slices that do not contain processes of their own.

## Additional resources

- For more information about **systemd**, unit files, and a complete list of **systemd** unit types, see the relevant sections in [Configuring basic system settings](#).
- For more information about resource controllers, see the [What are kernel resource controllers](#) section and the **systemd.resource-control(5)**, **cgroups(7)** manual pages.
- For more information about **fork()**, see the **fork(2)** manual pages.

# CHAPTER 85. USING CONTROL GROUPS VERSION 1 WITH SYSTEMD

The following sections provide an overview of tasks related to creation, modification and removal of the control groups (**cgroups**). The utilities provided by the **systemd** system and service manager are the preferred way of the **cgroups** management and will be supported in the future.

## 85.1. CREATING CONTROL GROUPS VERSION 1 WITH SYSTEMD

You can use the **systemd** system and service manager to create transient and persistent control groups (**cgroups**) to set limits, prioritize, or control access to hardware resources for groups of processes.

### 85.1.1. Creating transient control groups

The transient **cgroups** set limits on resources consumed by a unit (service or scope) during its runtime.

#### Procedure

- To create a transient control group, use the **systemd-run** command in the following format:

```
systemd-run --unit=<name> --slice=<name>.slice <command>
```

This command creates and starts a transient service or a scope unit and runs a custom command in such a unit.

- The **--unit=<name>** option gives a name to the unit. If **--unit** is not specified, the name is generated automatically.
  - The **--slice=<name>.slice** option makes your service or scope unit a member of a specified slice. Replace **<name>.slice** with the name of an existing slice (as shown in the output of **systemctl -t slice**), or create a new slice by passing a unique name. By default, services and scopes are created as members of the **system.slice**.
  - Replace **<command>** with the command you wish to execute in the service or the scope unit.
- The following message is displayed to confirm that you created and started the service or the scope successfully:

```
Running as unit <name>.service
```

- Optionally, keep the unit running after its processes finished to collect run-time information:

```
systemd-run --unit=<name> --slice=<name>.slice --remain-after-exit <command>
```

The command creates and starts a transient service unit and runs a custom command in such a unit. The **--remain-after-exit** option ensures that the service keeps running after its processes have finished.

#### Additional resources

- For more information about the concept of control groups, see [Chapter 81, Understanding control groups](#).

- For more information about the role of **systemd** in control groups, see [Chapter 84, Role of systemd in control groups version 1](#).
- For more information about **systemd**, unit configuration files and their locations, and a complete list of **systemd** unit types, see the relevant sections in [Configuring basic system settings](#).
- For a detailed description of **systemd-run** including further options and examples, see the **systemd-run(1)** manual pages.

### 85.1.2. Creating persistent control groups

To assign a persistent control group to a service, it is necessary to edit its unit configuration file. The configuration is preserved after the system reboot, so it can be used to manage services that are started automatically.

#### Procedure

- To create a persistent control group, execute:

```
systemctl enable <name>.service
```

The command above automatically creates a unit configuration file into the **/usr/lib/systemd/system/** directory and by default, it assigns **<name>.service** to the **system.slice** unit.

#### Additional resources

- For more information about the concept of control groups, see [Chapter 81, Understanding control groups](#).
- For more information about the role of **systemd** in control groups, see [Chapter 84, Role of systemd in control groups version 1](#).
- For more information about **systemd**, unit configuration files and their locations, and a complete list of **systemd** unit types, see the relevant sections in [Configuring basic system settings](#).
- For a detailed description of **systemd-run** including further options and examples, see the **systemd-run(1)** manual pages.

## 85.2. MODIFYING CONTROL GROUPS VERSION 1 WITH SYSTEMD

Each persistent unit is supervised by the **systemd** system and service manager, and has a unit configuration file in the **/usr/lib/systemd/system/** directory. To change the resource control settings of the persistent units, modify its unit configuration file either manually in a text editor or from the command-line interface.

### 85.2.1. Configuring memory resource control settings on the command-line

Executing commands in the command-line interface is one of the ways how to set limits, prioritize, or control access to hardware resources for groups of processes.

#### Procedure

- To limit the memory usage of a service, run the following:

```
systemctl set-property example.service MemoryLimit=1500K
```

The command instantly assigns the memory limit of 1,500 kilobytes to processes executed in a control group the **example.service** service belongs to. The **MemoryLimit** parameter, in this configuration variant, is defined in the **/etc/systemd/system.control/example.service.d/50-MemoryLimit.conf** file and controls the value of the **/sys/fs/cgroup/memory/system.slice/example.service/memory.limit\_in\_bytes** file.

- Optionally, to temporarily limit the memory usage of a service, run:

```
systemctl set-property --runtime example.service MemoryLimit=1500K
```

The command instantly assigns the memory limit to the **example.service** service. The **MemoryLimit** parameter is defined until the next reboot in the **/run/systemd/system.control/example.service.d/50-MemoryLimit.conf** file. With a reboot, the whole **/run/systemd/system.control/** directory and **MemoryLimit** are removed.



#### NOTE

The **50-MemoryLimit.conf** file stores the memory limit as a multiple of 4096 bytes – one kernel page size specific for AMD64 and Intel 64. The actual number of bytes depends on a CPU architecture.

#### Additional resources

- For more information about the concept of control groups, see [Chapter 81, Understanding control groups](#).
- For more information about resource controllers, see [Chapter 82, What kernel resource controllers are](#) and **systemd.resource-control(5)**, **cgroups(7)** manual pages.
- For more information about the role of **systemd** in control groups, see [Chapter 84, Role of systemd in control groups version 1](#).

### 85.2.2. Configuring memory resource control settings with unit files

Manually modifying unit files is one of the ways how to set limits, prioritize, or control access to hardware resources for groups of processes.

#### Procedure

1. To limit the memory usage of a service, modify the **/usr/lib/systemd/system/example.service** file as follows:

```
...
[Service]
MemoryLimit=1500K
...
```

The configuration above places a limit on maximum memory consumption of processes executed in a control group, which **example.service** is part of.

**NOTE**

Use suffixes K, M, G, or T to identify Kilobyte, Megabyte, Gigabyte, or Terabyte as a unit of measurement.

2. Reload all unit configuration files:

```
systemctl daemon-reload
```

3. Restart the service:

```
systemctl restart example.service
```

4. Reboot the system.

5. Optionally, check that the changes took effect:

```
cat /sys/fs/cgroup/memory/system.slice/example.service/memory.limit_in_bytes
1536000
```

The example output shows that the memory consumption was limited at around 1,500 Kilobytes.

**NOTE**

The **memory.limit\_in\_bytes** file stores the memory limit as a multiple of 4096 bytes – one kernel page size specific for AMD64 and Intel 64. The actual number of bytes depends on a CPU architecture.

**Additional resources**

- For more information about the concept of control groups, see [Chapter 81, Understanding control groups](#).
- For more information about resource controllers, see [Chapter 82, What kernel resource controllers are](#) and **systemd.resource-control(5)**, **cgroups(7)** manual pages.
- For more information about **systemd**, unit configuration files and their locations, as well as a complete list of **systemd** unit types, see the relevant sections in [Configuring basic system settings](#).
- For more information about the role of **systemd** in control groups, see [Chapter 84, Role of systemd in control groups version 1](#).

## 85.3. REMOVING CONTROL GROUPS VERSION 1 WITH SYSTEMD

You can use the **systemd** system and service manager to remove transient and persistent control groups (**cgroups**) if you no longer need to limit, prioritize, or control access to hardware resources for groups of processes.

### 85.3.1. Removing transient control groups

Transient **cgroups** are automatically released once all the processes that a service or a scope unit contains, finish.

## Procedure

- To stop the service unit with all its processes, execute:

```
systemctl stop <name>.service
```

- To terminate one or more of the unit processes, execute:

```
systemctl kill <name>.service --kill-who=PID,... --signal=signal
```

The command above uses the **--kill-who** option to select process(es) from the control group you wish to terminate. To kill multiple processes at the same time, pass a comma-separated list of PIDs. The **--signal** option determines the type of POSIX signal to be sent to the specified processes. The default signal is *SIGTERM*.

## Additional resources

- For more information about the concept of control groups, see [Chapter 81, Understanding control groups](#).
- For more information about resource controllers, see [Chapter 82, What kernel resource controllers are](#) and **systemd.resource-control(5)**, **cgroups(7)** manual pages.
- For more information about the role of **systemd** in control groups, see [Chapter 84, Role of systemd in control groups version 1](#).
- For more information about **systemd**, unit configuration files and their locations, as well as a complete list of **systemd** unit types, see the relevant sections in [Configuring basic system settings](#).

### 85.3.2. Removing persistent control groups

Persistent **cgroups** are released when a service or a scope unit is stopped or disabled and its configuration file is deleted.

## Procedure

1. Stop the service unit:

```
systemctl stop <name>.service
```

2. Disable the service unit:

```
systemctl disable <name>.service
```

3. Remove the relevant unit configuration file:

```
rm /usr/lib/systemd/system/<name>.service
```

4. Reload all unit configuration files so that changes take effect:

```
systemctl daemon-reload
```

## Additional resources

- For more information about the concept of control groups, see [Chapter 81, Understanding control groups](#).
- For more information about resource controllers, see [Chapter 82, What kernel resource controllers are](#) and **systemd.resource-control(5)**, **cgroups(7)** manual pages.
- For more information about the role of **systemd** in control groups, see [Chapter 84, Role of systemd in control groups version 1](#).
- For more information about **systemd**, unit configuration files and their locations, as well as a complete list of **systemd** unit types, see the relevant sections in [Configuring basic system settings](#).
- For more information about killing processes with **systemd**, see the **systemd.kill(5)** manual page.

# CHAPTER 86. OBTAINING INFORMATION ABOUT CONTROL GROUPS VERSION 1

The following sections describe how to display various information about control groups (**cgroups**):

- Listing **systemd** units and viewing their status
- Viewing the **cgroups** hierarchy
- Monitoring resource consumption in real time

## 86.1. LISTING SYSTEMD UNITS

The following procedure describes how to use the **systemd** system and service manager to list its units.

### Prerequisites

- [Role of systemd in control groups](#)

### Procedure

- To list all active units on the system, execute the **# systemctl** command and the terminal will return an output similar to the following example:

UNIT	LOAD	ACTIVE	SUB	DESCRIPTION
...				
init.scope	loaded	active	running	System and Service Manager
session-2.scope	loaded	active	running	Session 2 of user jdoe
abrt-ccpp.service	loaded	active	exited	Install ABRT coredump hook
abrt-oops.service	loaded	active	running	ABRT kernel log watcher
abrt-vmcore.service	loaded	active	exited	Harvest vmcores for ABRT
abrt-xorg.service	loaded	active	running	ABRT Xorg log watcher
...				
-.slice	loaded	active	active	Root Slice
machine.slice	loaded	active	active	Virtual Machine and Container
Slice system-getty.slice				loaded active active
system-getty.slice				
system-lvm2\x2dpvscan.slice				loaded active active system-
lvm2\x2dpvscan.slice				
system-sshd\x2dkeygen.slice				loaded active active system-
sshd\x2dkeygen.slice				
system-systemd\x2dhibernate\x2dresume.slice				loaded active active system-
systemd\x2dhibernate\x2dresume>				
system-user\x2druntime\x2ddir.slice				loaded active active system-
user\x2druntime\x2ddir.slice				
system.slice	loaded	active	active	System Slice
user-1000.slice	loaded	active	active	User Slice of UID 1000
user-42.slice	loaded	active	active	User Slice of UID 42
user.slice	loaded	active	active	User and Session Slice
...				

- **UNIT** – a name of a unit that also reflects the unit position in a control group hierarchy. The units relevant for resource control are a *slice*, a *scope*, and a *service*.

- **LOAD** – indicates whether the unit configuration file was properly loaded. If the unit file failed to load, the field contains the state *error* instead of *loaded*. Other unit load states are: *stub*, *merged*, and *masked*.
  - **ACTIVE** – the high-level unit activation state, which is a generalization of **SUB**.
  - **SUB** – the low-level unit activation state. The range of possible values depends on the unit type.
  - **DESCRIPTION** – the description of the unit content and functionality.
- To list inactive units, execute:
 

```
systemctl --all
```
  - To limit the amount of information in the output, execute:
 

```
systemctl --type service,masked
```

The **--type** option requires a comma-separated list of unit types such as a *service* and a *slice*, or unit load states such as *loaded* and *masked*.

### Additional resources

- For more information about **systemd**, unit files, and a complete list of **systemd** unit types, see the relevant sections in [Configuring basic system settings](#).

## 86.2. VIEWING A CONTROL GROUP VERSION 1 HIERARCHY

The following procedure describes how to display control groups (**cgroups**) hierarchy and processes running in specific **cgroups**.

### Prerequisites

- [Chapter 81, Understanding control groups](#)
- [Chapter 84, Role of systemd in control groups version 1](#)

### Procedure

- To display the whole **cgroups** hierarchy on your system, execute **# `systemctl-cgls`**:

```
Control group /:
-.slice
|__user.slice
| |__user-42.slice
| |__session-c1.scope
| |__965 gdm-session-worker [pam/gdm-launch-environment]
| |__1040 /usr/libexec/gdm-x-session gnome-session --autostart
| /usr/share/gdm/greeter/autostart
...
|__init.scope
 |__1 /usr/lib/systemd/systemd --switched-root --system --deserialize 18
 |__system.slice
...
```

```

└── example.service
 ├── 6882 /bin/bash /home/jdoe/example.sh
 └── 6902 sleep 1
└── systemd-journald.service
 └── 629 /usr/lib/systemd/systemd-journald
...

```

The example output returns the entire **cgroups** hierarchy, where the highest level is formed by *slices*.

- To display the **cgroups** hierarchy filtered by a resource controller, execute **# `systemd-cgls <resource_controller>`**:

```

systemd-cgls memory
Controller memory; Control group /:
└── 1 /usr/lib/systemd/systemd --switched-root --system --deserialize 18
 └── user.slice
 ├── user-42.slice
 │ ├── session-c1.scope
 │ │ └── 965 gdm-session-worker [pam/gdm-launch-environment]
 ...
 └── system.slice
 ...
 └── chronyd.service
 └── 844 /usr/sbin/chronyd
 └── example.service
 ├── 8914 /bin/bash /home/jdoe/example.sh
 └── 8916 sleep 1
...

```

The example output of the above command lists the services that interact with the selected controller.

- To display detailed information about a certain unit and its part of the **cgroups** hierarchy, execute **# `systemctl status <system_unit>`**:

```

systemctl status example.service
● example.service - My example service
 Loaded: loaded (/usr/lib/systemd/system/example.service; enabled; vendor preset: disabled)
 Active: active (running) since Tue 2019-04-16 12:12:39 CEST; 3s ago
 Main PID: 17737 (bash)
 Tasks: 2 (limit: 11522)
 Memory: 496.0K (limit: 1.5M)
 CGroup: /system.slice/example.service
 └─17737 /bin/bash /home/jdoe/example.sh
 └─17743 sleep 1
Apr 16 12:12:39 redhat systemd[1]: Started My example service.
Apr 16 12:12:39 redhat bash[17737]: The current time is Tue Apr 16 12:12:39 CEST 2019
Apr 16 12:12:40 redhat bash[17737]: The current time is Tue Apr 16 12:12:40 CEST 2019

```

## Additional resources

- For more information about resource controllers, see [Chapter 82, What kernel resource controllers are](#) section and **systemd.resource-control(5)**, **cgroups(7)** manual pages.

## 86.3. VIEWING RESOURCE CONTROLLERS

The following procedure describes how to learn which processes use which resource controllers.

### Prerequisites

- [Chapter 82, What kernel resource controllers are](#)
- [Chapter 81, Understanding control groups](#)

### Procedure

- To view which resource controllers a process interacts with, execute the **# cat /proc/<PID>/cgroup** command:

```
cat /proc/11269/cgroup
12:freezer:/
11:cpuset:/
10:devices:/system.slice
9:memory:/system.slice/example.service
8:pids:/system.slice/example.service
7:hugetlb:/
6:rdma:/
5:perf_event:/
4:cpu,cpuacct:/
3:net_cls,net_prio:/
2:blkio:/
1:name=systemd:/system.slice/example.service
```

The example output relates to a process of interest. In this case, it is a process identified by **PID 11269**, which belongs to the **example.service** unit. You can determine whether the process was placed in a correct control group as defined by the **systemd** unit file specifications.



### NOTE

By default, the items and their ordering in the list of resource controllers is the same for all units started by **systemd**, since it automatically mounts all the default resource controllers.

### Additional resources

- For more information about resource controllers in general refer to the **cgroups(7)** manual pages.
- For a detailed description of specific resource controllers, see the documentation in the **/usr/share/doc/kernel-doc-<kernel\_version>/Documentation/cgroups-v1/** directory.

## 86.4. MONITORING RESOURCE CONSUMPTION

The following procedure describes how to view a list of currently running control groups (**cgroups**) and their resource consumption in real-time.

## Prerequisites

- [Chapter 81, Understanding control groups](#)
- [Chapter 82, What kernel resource controllers are](#)
- [Chapter 84, Role of `systemd` in control groups version 1](#)

## Procedure

1. To see a dynamic account of currently running **cgroups**, execute the **# `systemd-cgtop`** command:

Control Group	Tasks	%CPU	Memory	Input/s	Output/s
/	607	29.8	1.5G	-	-
/system.slice	125	-	428.7M	-	-
/system.slice/ModemManager.service	3	-	8.6M	-	-
/system.slice/NetworkManager.service	3	-	12.8M	-	-
/system.slice/accounts-daemon.service	3	-	1.8M	-	-
/system.slice/boot.mount	-	-	48.0K	-	-
/system.slice/chronyd.service	1	-	2.0M	-	-
/system.slice/cockpit.socket	-	-	1.3M	-	-
/system.slice/colord.service	3	-	3.5M	-	-
/system.slice/crond.service	1	-	1.8M	-	-
/system.slice/cups.service	1	-	3.1M	-	-
/system.slice/dev-hugepages.mount	-	-	244.0K	-	-
/system.slice/dev-mapper-rhel\x2dswap.swap	-	-	912.0K	-	-
/system.slice/dev-mqueue.mount	-	-	48.0K	-	-
/system.slice/example.service	2	-	2.0M	-	-
/system.slice/firewalld.service	2	-	28.8M	-	-
...					

The example output displays currently running **cgroups** ordered by their resource usage (CPU, memory, disk I/O load). The list refreshes every 1 second by default. Therefore, it offers a dynamic insight into the actual resource usage of each control group.

## Additional resources

- For more information about dynamic monitoring of resource usage, see the **`systemd-cgtop(1)`** manual pages.

# CHAPTER 87. WHAT NAMESPACES ARE

Namespaces are one of the most important methods for organizing and identifying software objects.

A namespace wraps a global system resource (for example a mount point, a network device, or a hostname) in an abstraction that makes it appear to processes within the namespace that they have their own isolated instance of the global resource. One of the most common technologies that utilize namespaces are containers.

Changes to a particular global resource are visible only to processes in that namespace and do not affect the rest of the system or other namespaces.

To inspect which namespaces a process is a member of, you can check the symbolic links in the `/proc/<PID>/ns` directory.

The following table shows supported namespaces and resources which they isolate:

Namespace	Isolates
Mount	Mount points
UTS	Hostname and NIS domain name
IPC	System V IPC, POSIX message queues
PID	Process IDs
Network	Network devices, stacks, ports, etc
User	User and group IDs
Control groups	Control group root directory

## Additional resources

- For more information about namespaces, see the **namespaces(7)** and **cgroup\_namespaces(7)** manual pages.
- For more information about **cgroups**, see [Chapter 81, Understanding control groups](#).

# CHAPTER 88. ANALYZING SYSTEM PERFORMANCE WITH BPF COMPILER COLLECTION

As a system administrator, use the BPF Compiler Collection (BCC) library to create tools for analyzing the performance of your Linux operating system and gathering information, which could be difficult to obtain through other interfaces.

## 88.1. A BRIEF INTRODUCTION TO BCC

BPF Compiler Collection (BCC) is a library, which facilitates the creation of the extended Berkeley Packet Filter (eBPF) programs. The main utility of eBPF programs is analyzing OS performance and network performance without experiencing overhead or security issues.

BCC removes the need for users to know deep technical details of eBPF, and provides many out-of-the-box starting points, such as the **bcc-tools** package with pre-created eBPF programs.



### NOTE

The eBPF programs are triggered on events, such as disk I/O, TCP connections, and process creations. It is unlikely that the programs should cause the kernel to crash, loop or become unresponsive because they run in a safe virtual machine in the kernel.

### Additional resources

- For more information about BCC, see the [/usr/share/doc/bcc/README.md](#) file.

## 88.2. INSTALLING THE BCC-TOOLS PACKAGE

This section describes how to install the **bcc-tools** package, which also installs the BPF Compiler Collection (BCC) library as a dependency.

### Prerequisites

- An active [Red Hat Enterprise Linux subscription](#)
- An [enabled repository](#) containing the **bcc-tools** package
- Introduction to [yum package manager](#)
- [Updated kernel](#)

### Procedure

- Install **bcc-tools**:

```
yum install bcc-tools
```

Once installed, the tools are placed in the [/usr/share/bcc/tools/](#) directory.

- Optionally, inspect the tools:

```
ll /usr/share/bcc/tools/
...
```

```

-rwxr-xr-x. 1 root root 4198 Dec 14 17:53 dcsnoop
-rwxr-xr-x. 1 root root 3931 Dec 14 17:53 dcstat
-rwxr-xr-x. 1 root root 20040 Dec 14 17:53 deadlock_detector
-rw-r--r--. 1 root root 7105 Dec 14 17:53 deadlock_detector.c
drwxr-xr-x. 3 root root 8192 Mar 11 10:28 doc
-rwxr-xr-x. 1 root root 7588 Dec 14 17:53 execsnoop
-rwxr-xr-x. 1 root root 6373 Dec 14 17:53 ext4dist
-rwxr-xr-x. 1 root root 10401 Dec 14 17:53 ext4slower
...

```

The **doc** directory in the listing above contains documentation for each tool.

## 88.3. USING SELECTED BCC-TOOLS FOR PERFORMANCE ANALYSES

This section describes how to use certain pre-created programs from the BPF Compiler Collection (BCC) library to efficiently and securely analyze the system performance on the per-event basis. The set of pre-created programs in the BCC library can serve as examples for creation of additional programs.

### Prerequisites

- [Introduction to BCC](#)
- [Installed BCC library](#)
- Root permissions

### Using **execsnoop** to examine the system processes

1. Execute the **execsnoop** program in one terminal:

```
/usr/share/bcc/tools/execsnoop
```

2. In another terminal execute for example:

```
$ ls /usr/share/bcc/tools/doc/
```

The above creates a short-lived process of the **ls** command.

3. The terminal running **execsnoop** shows the output similar to the following:

```

PCOMM PID PPID RET ARGS
ls 8382 8287 0 /usr/bin/ls --color=auto /usr/share/bcc/tools/doc/
sed 8385 8383 0 /usr/bin/sed s/^ *[0-9]+\+ *//
...

```

The **execsnoop** program prints a line of output for each new process, which consumes system resources. It even detects processes of programs that run very shortly, such as **ls**, and most monitoring tools would not register them.

The result above shows a parent process name (**ls**), its process ID (**5076**), parent process ID (**2931**), the return value of the **exec()** system call (**0**), which loads program code into new processes. Finally, the output displays a location of the started program with arguments (**/usr/bin/ls --color=auto /usr/share/bcc/tools/doc/**).

To see more details, examples, and options for **execsnoop**, refer to the **/usr/share/bcc/tools/doc/execsnoop\_example.txt** file.

For more information about **exec()**, see **exec(3)** manual pages.

## Using **opensnoop** to track what files a command opens

1. Execute the **opensnoop** program in one terminal:

```
/usr/share/bcc/tools/opensnoop -n uname
```

The above prints output for files, which are opened only by the process of the **uname** command.

2. In another terminal execute:

```
$ uname
```

The command above opens certain files, which are captured in the next step.

3. The terminal running **opensnoop** shows the output similar to the following:

```
PID COMM FD ERR PATH
8596 uname 3 0 /etc/ld.so.cache
8596 uname 3 0 /lib64/libc.so.6
8596 uname 3 0 /usr/lib/locale/locale-archive
...
...
```

The **opensnoop** program watches the **open()** system call across the whole system, and prints a line of output for each file that **uname** tried to open along the way.

The result above shows a process ID (**PID**), a process name (**COMM**), and a file descriptor (**FD**) – a value that **open()** returns to refer to the open file. Finally, the output displays a column for errors (**ERR**) and a location of files that **open()** tries to open (**PATH**).

If a command tries to read a non-existent file, then the **FD** column returns **-1** and the **ERR** column prints a value corresponding to the relevant error. As a result, **opensnoop** can help you identify an application that does not behave properly.

To see more details, examples, and options for **opensnoop**, refer to the **/usr/share/bcc/tools/doc/opensnoop\_example.txt** file.

For more information about **open()**, see **open(2)** manual pages.

## Using **biotop** to examine the I/O operations on the disk

1. Execute the **biotop** program in one terminal:

```
/usr/share/bcc/tools/biotop 30
```

The command enables you to monitor the top processes, which perform I/O operations on the disk. The argument ensures that the command will produce a 30 second summary.



### NOTE

When no argument provided, the output screen by default refreshes every 1 second.

2. In another terminal execute for example :

```
dd if=/dev/vda of=/dev/zero
```

The command above reads the content from the local hard disk device and writes the output to the **/dev/zero** file. This step generates certain I/O traffic to illustrate **biotop**.

3. The terminal running **biotop** shows the output similar to the following:

PID	COMM	D	MAJ	MIN	DISK	I/O	Kbytes	AVGms
9568	dd	R	252	0	vda	16294	14440636.0	3.69
48	kswapd0	W	252	0	vda	1763	120696.0	1.65
7571	gnome-shell	R	252	0	vda	834	83612.0	0.33
1891	gnome-shell	R	252	0	vda	1379	19792.0	0.15
7515	Xorg	R	252	0	vda	280	9940.0	0.28
7579	llvmpipe-1	R	252	0	vda	228	6928.0	0.19
9515	gnome-control-c	R	252	0	vda	62	6444.0	0.43
8112	gnome-terminal-	R	252	0	vda	67	2572.0	1.54
7807	gnome-software	R	252	0	vda	31	2336.0	0.73
9578	awk	R	252	0	vda	17	2228.0	0.66
7578	llvmpipe-0	R	252	0	vda	156	2204.0	0.07
9581	pgrep	R	252	0	vda	58	1748.0	0.42
7531	InputThread	R	252	0	vda	30	1200.0	0.48
7504	gdbus	R	252	0	vda	3	1164.0	0.30
1983	llvmpipe-1	R	252	0	vda	39	724.0	0.08
1982	llvmpipe-0	R	252	0	vda	36	652.0	0.06
...								

The results shows that the **dd** process, with the process ID 9568, performed 16,294 read operations from the **vda** disk. The read operations reached total of 14,440,636 Kbytes with an average I/O time 3.69 ms.

To see more details, examples, and options for **biotop**, refer to the **/usr/share/bcc/tools/doc/biotop\_example.txt** file.

For more information about **dd**, see **dd(1)** manual pages.

## Using **xfsslower** to expose unexpectedly slow file system operations

1. Execute the **xfsslower** program in one terminal:

```
/usr/share/bcc/tools/xfsslower 1
```

The command above measures the time the XFS file system spends in performing read, write, open or sync (**fsync**) operations. The **1** argument ensures that the program shows only the operations that are slower than 1 ms.



### NOTE

When no arguments provided, **xfsslower** by default displays operations slower than 10 ms.

2. In another terminal execute, for example, the following:

```
$ vim text
```

The command above creates a text file in the **vim** editor to initiate certain interaction with the XFS file system.

3. The terminal running **xfsslower** shows something similar upon saving the file from the previous step:

TIME	COMM	PID	T	BYTES	OFF_KB	LAT(ms)	FILENAME
13:07:14	b'bash'	4754	R	256	0	7.11	b'vim'
13:07:14	b'vim'	4754	R	832	0	4.03	b'libgpm.so.2.1.0'
13:07:14	b'vim'	4754	R	32	20	1.04	b'libgpm.so.2.1.0'
13:07:14	b'vim'	4754	R	1982	0	2.30	b'vimrc'
13:07:14	b'vim'	4754	R	1393	0	2.52	b'getscriptPlugin.vim'
13:07:45	b'vim'	4754	S	0	0	6.71	b'text'
13:07:45	b'pool'	2588	R	16	0	5.58	b'text'
...							

Each line above represents an operation in the file system, which took more time than a certain threshold. **xfsslower** is good at exposing possible file system problems, which can take form of unexpectedly slow operations.

The **T** column represents operation type ( **R**ead/**W**rite/**S**ync), **OFF\_KB** is a file offset in KB. **FILENAME** is the file the process (**COMM**) is trying to read, write, or sync.

To see more details, examples, and options for **xfsslower**, refer to the [`/usr/share/bcc/tools/doc/xfsslower\_example.txt`](#) file.

For more information about **fsync**, see **fsync(2)** manual pages.

## **PART VIII. DESIGN OF HIGH AVAILABILITY SYSTEM**

# CHAPTER 89. HIGH AVAILABILITY ADD-ON OVERVIEW

The High Availability Add-On is a clustered system that provides reliability, scalability, and availability to critical production services.

A cluster is two or more computers (called *nodes* or *members*) that work together to perform a task. Clusters can be used to provide highly available services or resources. The redundancy of multiple machines is used to guard against failures of many types.

High availability clusters provide highly available services by eliminating single points of failure and by failing over services from one cluster node to another in case a node becomes inoperative. Typically, services in a high availability cluster read and write data (by means of read-write mounted file systems). Therefore, a high availability cluster must maintain data integrity as one cluster node takes over control of a service from another cluster node. Node failures in a high availability cluster are not visible from clients outside the cluster. (High availability clusters are sometimes referred to as failover clusters.) The High Availability Add-On provides high availability clustering through its high availability service management component, **Pacemaker**.

## 89.1. HIGH AVAILABILITY ADD-ON COMPONENTS

The High Availability Add-On consists of the following major components:

- Cluster infrastructure – Provides fundamental functions for nodes to work together as a cluster: configuration file management, membership management, lock management, and fencing.
- High availability service management – Provides failover of services from one cluster node to another in case a node becomes inoperative.
- Cluster administration tools – Configuration and management tools for setting up, configuring, and managing the High Availability Add-On. The tools are for use with the cluster infrastructure components, the high availability and service management components, and storage.

You can supplement the High Availability Add-On with the following components:

- Red Hat GFS2 (Global File System 2) – Part of the Resilient Storage Add-On, this provides a cluster file system for use with the High Availability Add-On. GFS2 allows multiple nodes to share storage at a block level as if the storage were connected locally to each cluster node. GFS2 cluster file system requires a cluster infrastructure.
- LVM Locking Daemon (**lvmlockd**) – Part of the Resilient Storage Add-On, this provides volume management of cluster storage. **lvmlockd** support also requires cluster infrastructure.
- Load Balancer Add-On – Routing software that provides high availability load balancing and failover in layer 4 (TCP) and layer 7 (HTTP, HTTPS) services. The Load Balancer Add-On runs in a cluster of redundant virtual routers that uses load algorithms to distribute client requests to real servers, collectively acting as a virtual server. It is not necessary to use the Load Balancer Add-On in conjunction with Pacemaker.

## 89.2. PACEMAKER OVERVIEW

Pacemaker is a cluster resource manager. It achieves maximum availability for your cluster services and resources by making use of the cluster infrastructure’s messaging and membership capabilities to detect and recover from node and resource-level failure.

### 89.2.1. Pacemaker architecture components

A cluster configured with Pacemaker comprises separate component daemons that monitor cluster membership, scripts that manage the services, and resource management subsystems that monitor the disparate resources.

The following components form the Pacemaker architecture:

### Cluster Information Base (CIB)

The Pacemaker information daemon, which uses XML internally to distribute and synchronize current configuration and status information from the Designated Coordinator (DC) – a node assigned by Pacemaker to store and distribute cluster state and actions by means of the CIB – to all other cluster nodes.

### Cluster Resource Management Daemon (CRMd)

Pacemaker cluster resource actions are routed through this daemon. Resources managed by CRMd can be queried by client systems, moved, instantiated, and changed when needed.

Each cluster node also includes a local resource manager daemon (LRMd) that acts as an interface between CRMd and resources. LRMd passes commands from CRMd to agents, such as starting and stopping and relaying status information.

### Shoot the Other Node in the Head (STONITH)

STONITH is the Pacemaker fencing implementation. It acts as a cluster resource in Pacemaker that processes fence requests, forcefully shutting down nodes and removing them from the cluster to ensure data integrity. STONITH is configured in the CIB and can be monitored as a normal cluster resource. For a general overview of fencing, see [Section 89.3, “Fencing overview”](#).

### corosync

**corosync** is the component – and a daemon of the same name – that serves the core membership and member-communication needs for high availability clusters. It is required for the High Availability Add-On to function.

In addition to those membership and messaging functions, **corosync** also:

- Manages quorum rules and determination.
- Provides messaging capabilities for applications that coordinate or operate across multiple members of the cluster and thus must communicate stateful or other information between instances.
- Uses the **kronosnet** library as its network transport to provide multiple redundant links and automatic failover.

## 89.2.2. Configuration and management tools

The High Availability Add-On features two configuration tools for cluster deployment, monitoring, and management.

### pcs

The **pcs** command line interface controls and configures Pacemaker and the **corosync** heartbeat daemon. A command-line based program, **pcs** can perform the following cluster management tasks:

- Create and configure a Pacemaker/Corosync cluster
- Modify configuration of the cluster while it is running
- Remotely configure both Pacemaker and Corosync as well as start, stop, and display status information of the cluster

### pcsd Web UI

A graphical user interface to create and configure Pacemaker/Corosync clusters.

## 89.2.3. The cluster and pacemaker configuration files

The configuration files for the Red Hat High Availability Add-On are **corosync.conf** and **cib.xml**.

The **corosync.conf** file provides the cluster parameters used by **corosync**, the cluster manager that Pacemaker is built on. In general, you should not edit the **corosync.conf** directly but, instead, use the **pcs** or **pcsd** interface.

The **cib.xml** file is an XML file that represents both the cluster's configuration and the current state of all resources in the cluster. This file is used by Pacemaker's Cluster Information Base (CIB). The contents of the CIB are automatically kept in sync across the entire cluster. Do not edit the **cib.xml** file directly; use the **pcs** or **pcsd** interface instead.

## 89.3. FENCING OVERVIEW

If communication with a single node in the cluster fails, then other nodes in the cluster must be able to restrict or release access to resources that the failed cluster node may have access to. This cannot be accomplished by contacting the cluster node itself as the cluster node may not be responsive. Instead, you must provide an external method, which is called fencing with a fence agent. A fence device is an external device that can be used by the cluster to restrict access to shared resources by an errant node, or to issue a hard reboot on the cluster node.

Without a fence device configured you do not have a way to know that the resources previously used by the disconnected cluster node have been released, and this could prevent the services from running on any of the other cluster nodes. Conversely, the system may assume erroneously that the cluster node has released its resources and this can lead to data corruption and data loss. Without a fence device configured data integrity cannot be guaranteed and the cluster configuration will be unsupported.

When the fencing is in progress no other cluster operation is allowed to run. Normal operation of the cluster cannot resume until fencing has completed or the cluster node rejoins the cluster after the cluster node has been rebooted.

For more information about fencing, see [Fencing in a Red Hat High Availability Cluster](#).

## 89.4. QUORUM OVERVIEW

In order to maintain cluster integrity and availability, cluster systems use a concept known as *quorum* to prevent data corruption and loss. A cluster has quorum when more than half of the cluster nodes are online. To mitigate the chance of data corruption due to failure, Pacemaker by default stops all resources if the cluster does not have quorum.

Quorum is established using a voting system. When a cluster node does not function as it should or loses communication with the rest of the cluster, the majority working nodes can vote to isolate and, if needed, fence the node for servicing.

For example, in a 6-node cluster, quorum is established when at least 4 cluster nodes are functioning. If the majority of nodes go offline or become unavailable, the cluster no longer has quorum and Pacemaker stops clustered services.

The quorum features in Pacemaker prevent what is also known as *split-brain*, a phenomenon where the cluster is separated from communication but each part continues working as separate clusters, potentially writing to the same data and possibly causing corruption or loss. For more information on

what it means to be in a split-brain state, and on quorum concepts in general, see [Exploring Concepts of RHEL High Availability Clusters - Quorum](#).

A Red Hat Enterprise Linux High Availability Add-On cluster uses the **votequorum** service, in conjunction with fencing, to avoid split brain situations. A number of votes is assigned to each system in the cluster, and cluster operations are allowed to proceed only when a majority of votes is present.

## 89.5. RESOURCE OVERVIEW

A *cluster resource* is an instance of program, data, or application to be managed by the cluster service. These resources are abstracted by *agents* that provide a standard interface for managing the resource in a cluster environment.

To ensure that resources remain healthy, you can add a monitoring operation to a resource's definition. If you do not specify a monitoring operation for a resource, one is added by default.

You can determine the behavior of a resource in a cluster by configuring *constraints*. You can configure the following categories of constraints:

- location constraints – A location constraint determines which nodes a resource can run on.
- ordering constraints – An ordering constraint determines the order in which the resources run.
- colocation constraints – A colocation constraint determines where resources will be placed relative to other resources.

One of the most common elements of a cluster is a set of resources that need to be located together, start sequentially, and stop in the reverse order. To simplify this configuration, Pacemaker supports the concept of *groups*.

## 89.6. LVM LOGICAL VOLUMES IN A RED HAT HIGH AVAILABILITY CLUSTER

The Red Hat High Availability Add-On provides support for LVM volumes in two distinct cluster configurations:

- High availability LVM volumes (HA-LVM) in active/passive failover configurations in which only a single node of the cluster accesses the storage at any one time.
- LVM volumes that use the **lvmlockd** daemon to manage storage devices in active/active configurations in which more than one node of the cluster requires access to the storage at the same time. The **lvmlockd** daemon is part of the Resilient Storage Add-On.

### 89.6.1. Choosing HA-LVM or shared volumes

When to use HA-LVM or shared logical volumes managed by the **lvmlockd** daemon should be based on the needs of the applications or services being deployed.

- If multiple nodes of the cluster require simultaneous read/write access to LVM volumes in an active/active system, then you must use the **lvmlockd** daemon and configure your volumes as shared volumes. The **lvmlockd** daemon provides a system for coordinating activation of and changes to LVM volumes across nodes of a cluster concurrently. The **lvmlockd** daemon's locking service provides protection to LVM metadata as various nodes of the cluster interact

with volumes and make changes to their layout. This protection is contingent upon configuring any volume group that will be activated simultaneously across multiple cluster nodes as a shared volume.

- If the high availability cluster is configured to manage shared resources in an active/passive manner with only one single member needing access to a given LVM volume at a time, then you can use HA-LVM without the **lvmlockd** locking service.

Most applications will run better in an active/passive configuration, as they are not designed or optimized to run concurrently with other instances. Choosing to run an application that is not cluster-aware on shared logical volumes may result in degraded performance. This is because there is cluster communication overhead for the logical volumes themselves in these instances. A cluster-aware application must be able to achieve performance gains above the performance losses introduced by cluster file systems and cluster-aware logical volumes. This is achievable for some applications and workloads more easily than others. Determining what the requirements of the cluster are and whether the extra effort toward optimizing for an active/active cluster will pay dividends is the way to choose between the two LVM variants. Most users will achieve the best HA results from using HA-LVM.

HA-LVM and shared logical volumes using **lvmlockd** are similar in the fact that they prevent corruption of LVM metadata and its logical volumes, which could otherwise occur if multiple machines are allowed to make overlapping changes. HA-LVM imposes the restriction that a logical volume can only be activated exclusively; that is, active on only one machine at a time. This means that only local (non-clustered) implementations of the storage drivers are used. Avoiding the cluster coordination overhead in this way increases performance. A shared volume using **lvmlockd** does not impose these restrictions and a user is free to activate a logical volume on all machines in a cluster; this forces the use of cluster-aware storage drivers, which allow for cluster-aware file systems and applications to be put on top.

### 89.6.2. Configuring LVM volumes in a cluster

In Red Hat Enterprise Linux 8, clusters are managed through Pacemaker. Both HA-LVM and shared logical volumes are supported only in conjunction with Pacemaker clusters, and must be configured as cluster resources.

- For examples of procedures for configuring an HA-LVM volume as part of a Pacemaker cluster, see [Configuring an active/passive Apache HTTP server in a Red Hat High Availability cluster](#) and [Configuring an active/passive NFS server in a Red Hat High Availability cluster](#). Note that these procedures include the following steps:
  - Ensuring that only the cluster is capable of activating the volume group
  - Configuring an LVM logical volume
  - Configuring the LVM volume as a cluster resource
- For a procedure for configuring shared LVM volumes that use the **lvmlockd** daemon to manage storage devices in active/active configurations, see [Configuring a GFS2 file system in a cluster](#)

# CHAPTER 90. GETTING STARTED WITH PACEMAKER

The following procedures provide an introduction to the tools and processes you use to create a Pacemaker cluster. They are intended for users who are interested in seeing what the cluster software looks like and how it is administered, without needing to configure a working cluster.



## NOTE

These procedures do not create a supported Red Hat cluster, which requires at least two nodes and the configuration of a fencing device.

## 90.1. LEARNING TO USE PACEMAKER

This example requires a single node running RHEL 8 and it requires a floating IP address that resides on the same network as one of the node's statically assigned IP addresses.

- The node used in this example is **z1.example.com**.
- The floating IP address used in this example is 192.168.122.120.



## NOTE

Ensure that the name of the node on which you are running is in your **/etc/hosts** file.

By working through this procedure, you will learn how to use Pacemaker to set up a cluster, how to display cluster status, and how to configure a cluster service. This example creates an Apache HTTP server as a cluster resource and shows how the cluster responds when the resource fails.

1. Install the Red Hat High Availability Add-On software packages from the High Availability channel, and start and enable the **pcsd** service.

```
yum install pcs pacemaker fence-agents-all
...
systemctl start pcsd.service
systemctl enable pcsd.service
```

If you are running the **firewalld** daemon, enable the ports that are required by the Red Hat High Availability Add-On.

```
firewall-cmd --permanent --add-service=high-availability
firewall-cmd --reload
```

2. Set a password for user **hacluster** on each node in the cluster and authenticate user **hacluster** for each node in the cluster on the node from which you will be running the **pcs** commands. This example is using only a single node, the node from which you are running the commands, but this step is included here since it is a necessary step in configuring a supported Red Hat High Availability multi-node cluster.

```
passwd hacluster
...
pcs host auth z1.example.com
```

3. Create a cluster named **my\_cluster** with one member and check the status of the cluster. This command creates and starts the cluster in one step.

```
pcs cluster setup my_cluster --start z1.example.com
...
pcs cluster status
Cluster Status:
Stack: corosync
Current DC: z1.example.com (version 2.0.0-10.el8-b67d8d0de9) - partition with quorum
Last updated: Thu Oct 11 16:11:18 2018
Last change: Thu Oct 11 16:11:00 2018 by hacluster via crmd on z1.example.com
1 node configured
0 resources configured

PCSD Status:
z1.example.com: Online
```

4. A Red Hat High Availability cluster requires that you configure fencing for the cluster. The reasons for this requirement are described in [Fencing in a Red Hat High Availability Cluster](#). For this introduction, however, which is intended to show only how to use the basic Pacemaker commands, disable fencing by setting the **stonith-enabled** cluster option to **false**.



#### WARNING

The use of **stonith-enabled=false** is completely inappropriate for a production cluster. It tells the cluster to simply pretend that failed nodes are safely fenced.

```
pcs property set stonith-enabled=false
```

5. Configure a web browser on your system and create a web page to display a simple text message. If you are running the **firewalld** daemon, enable the ports that are required by **httpd**.



#### NOTE

Do not use **systemctl enable** to enable any services that will be managed by the cluster to start at system boot.

```
yum install -y httpd wget
...
firewall-cmd --permanent --add-service=http
firewall-cmd --reload

cat <<-END >/var/www/html/index.html
<html>
<body>My Test Site - $(hostname)</body>
</html>
END
```

In order for the Apache resource agent to get the status of Apache, create the following addition to the existing configuration to enable the status server URL.

```
cat <<-END > /etc/httpd/conf.d/status.conf
<Location /server-status>
SetHandler server-status
Order deny,allow
Deny from all
Allow from 127.0.0.1
Allow from ::1
</Location>
END
```

6. Create **IPaddr2** and **apache** resources for the cluster to manage. The 'IPaddr2' resource is a floating IP address that must not be one already associated with a physical node. If the 'IPaddr2' resource's NIC device is not specified, the floating IP must reside on the same network as the statically assigned IP address used by the node.

You can display a list of all available resource types with the **pcs resource list** command. You can use the **pcs resource describe resourcetype** command to display the parameters you can set for the specified resource type. For example, the following command displays the parameters you can set for a resource of type **apache**:

```
pcs resource describe apache
...
```

In this example, the IP address resource and the apache resource are both configured as part of a group named **apachegroup**, which ensures that the resources are kept together to run on the same node when you are configuring a working multi-node cluster.

```
pcs resource create ClusterIP ocf:heartbeat:IPaddr2 ip=192.168.122.120 --group
apachegroup

pcs resource create WebSite ocf:heartbeat:apache
configfile=/etc/httpd/conf/httpd.conf statusurl="http://localhost/server-status" --group
apachegroup

pcs status
Cluster name: my_cluster
Stack: corosync
Current DC: z1.example.com (version 2.0.0-10.el8-b67d8d0de9) - partition with quorum
Last updated: Fri Oct 12 09:54:33 2018
Last change: Fri Oct 12 09:54:30 2018 by root via cibadmin on z1.example.com

1 node configured
2 resources configured

Online: [z1.example.com]

Full list of resources:

Resource Group: apachegroup
 ClusterIP (ocf::heartbeat:IPaddr2): Started z1.example.com
 WebSite (ocf::heartbeat:apache): Started z1.example.com
```

```
PCSD Status:
z1.example.com: Online
...
```

After you have configured a cluster resource, you can use the **pcs resource config** command to display the options that are configured for that resource.

```
pcs resource config WebSite
Resource: WebSite (class=ocf provider=heartbeat type=apache)
Attributes: configfile=/etc/httpd/conf/httpd.conf statusurl=http://localhost/server-status
Operations: start interval=0s timeout=40s (WebSite-start-interval-0s)
 stop interval=0s timeout=60s (WebSite-stop-interval-0s)
 monitor interval=1min (WebSite-monitor-interval-1min)
```

7. Point your browser to the website you created using the floating IP address you configured. This should display the text message you defined.
8. Stop the apache web service and check the cluster status. Using **killall -9** simulates an application-level crash.

```
killall -9 httpd
```

Check the cluster status. You should see that stopping the web service caused a failed action, but that the cluster software restarted the service and you should still be able to access the website.

```
pcs status
Cluster name: my_cluster
...
Current DC: z1.example.com (version 1.1.13-10.el7-44eb2dd) - partition with quorum
1 node and 2 resources configured

Online: [z1.example.com]

Full list of resources:

Resource Group: apachegroup
 ClusterIP (ocf::heartbeat:IPaddr2): Started z1.example.com
 WebSite (ocf::heartbeat:apache): Started z1.example.com

Failed Resource Actions:
* WebSite_monitor_60000 on z1.example.com 'not running' (7): call=13, status=complete,
 exitreason='none',
 last-rc-change='Thu Oct 11 23:45:50 2016', queued=0ms, exec=0ms

PCSD Status:
z1.example.com: Online
```

You can clear the failure status on the resource that failed once the service is up and running again and the failed action notice will no longer appear when you view the cluster status.

```
pcs resource cleanup WebSite
```

9. When you are finished looking at the cluster and the cluster status, stop the cluster services on

the node. Even though you have only started services on one node for this introduction, the **--all** parameter is included since it would stop cluster services on all nodes on an actual multi-node cluster.

```
pcs cluster stop --all
```

## 90.2. LEARNING TO CONFIGURE FAILOVER

This procedure provides an introduction to creating a Pacemaker cluster running a service that will fail over from one node to another when the node on which the service is running becomes unavailable. By working through this procedure, you can learn how to create a service in a two-node cluster and you can then observe what happens to that service when it fails on the node on which it is running.

This example procedure configures a two-node Pacemaker cluster running an Apache HTTP server. You can then stop the Apache service on one node to see how the service remains available.

This procedure requires as a prerequisite that you have two nodes running Red Hat Enterprise Linux 8 that can communicate with each other, and it requires a floating IP address that resides on the same network as one of the node's statically assigned IP addresses.

- The nodes used in this example are **z1.example.com** and **z2.example.com**.
- The floating IP address used in this example is 192.168.122.120.



### NOTE

Ensure that the names of the nodes you are using are in the **/etc/hosts** file on each node.

1. On both nodes, install the Red Hat High Availability Add-On software packages from the High Availability channel, and start and enable the **pcsd** service.

```
yum install pcs pacemaker fence-agents-all
...
systemctl start pcsd.service
systemctl enable pcsd.service
```

If you are running the **firewalld** daemon, on both nodes enable the ports that are required by the Red Hat High Availability Add-On.

```
firewall-cmd --permanent --add-service=high-availability
firewall-cmd --reload
```

2. On both nodes in the cluster, set a password for user **hacluster**.

```
passwd hacluster
```

3. Authenticate user **hacluster** for each node in the cluster on the node from which you will be running the **pcs** commands.

```
pcs host auth z1.example.com z2.example.com
```

4. Create a cluster named **my\_cluster** with both nodes as cluster members. This command creates and starts the cluster in one step. You only need to run this from one node in the cluster because **pcs** configuration commands take effect for the entire cluster.  
On one node in cluster, run the following command.

```
pcs cluster setup my_cluster --start z1.example.com z2.example.com
```

5. A Red Hat High Availability cluster requires that you configure fencing for the cluster. The reasons for this requirement are described in [Fencing in a Red Hat High Availability Cluster](#). For this introduction, however, to show only how failover works in this configuration, disable fencing by setting the **stonith-enabled** cluster option to **false**



### WARNING

The use of **stonith-enabled=false** is completely inappropriate for a production cluster. It tells the cluster to simply pretend that failed nodes are safely fenced.

```
pcs property set stonith-enabled=false
```

6. After creating a cluster and disabling fencing, check the status of the cluster.



### NOTE

When you run the **pcs cluster status** command, it may show output that temporarily differs slightly from the examples as the system components start up.

```
pcs cluster status
```

Cluster Status:

Stack: corosync

Current DC: z1.example.com (version 2.0.0-10.el8-b67d8d0de9) - partition with quorum

Last updated: Thu Oct 11 16:11:18 2018

Last change: Thu Oct 11 16:11:00 2018 by hacluster via crmd on z1.example.com

2 nodes configured

0 resources configured

PCSD Status:

z1.example.com: Online

z2.example.com: Online

7. On both nodes, configure a web browser and create a web page to display a simple text message. If you are running the **firewalld** daemon, enable the ports that are required by **httpd**.



### NOTE

Do not use **systemctl enable** to enable any services that will be managed by the cluster to start at system boot.

```
yum install -y httpd wget
...
firewall-cmd --permanent --add-service=http
firewall-cmd --reload

cat <<-END >/var/www/html/index.html
<html>
<body>My Test Site - $(hostname)</body>
</html>
END
```

In order for the Apache resource agent to get the status of Apache, on each node in the cluster create the following addition to the existing configuration to enable the status server URL.

```
cat <<-END > /etc/httpd/conf.d/status.conf
<Location /server-status>
SetHandler server-status
Order deny,allow
Deny from all
Allow from 127.0.0.1
Allow from ::1
</Location>
END
```

8. Create **IPaddr2** and **apache** resources for the cluster to manage. The 'IPaddr2' resource is a floating IP address that must not be one already associated with a physical node. If the 'IPaddr2' resource's NIC device is not specified, the floating IP must reside on the same network as the statically assigned IP address used by the node.

You can display a list of all available resource types with the **pcs resource list** command. You can use the **pcs resource describe resourcetype** command to display the parameters you can set for the specified resource type. For example, the following command displays the parameters you can set for a resource of type **apache**:

```
pcs resource describe apache
...
```

In this example, the IP address resource and the apache resource are both configured as part of a group named **apachegroup**, which ensures that the resources are kept together to run on the same node.

Run the following commands from one node in the cluster:

```
pcs resource create ClusterIP ocf:heartbeat:IPaddr2 ip=192.168.122.120 --group
apachegroup

pcs resource create WebSite ocf:heartbeat:apache
configfile=/etc/httpd/conf/httpd.conf statusurl="http://localhost/server-status" --group
apachegroup

pcs status
Cluster name: my_cluster
Stack: corosync
Current DC: z1.example.com (version 2.0.0-10.el8-b67d8d0de9) - partition with quorum
Last updated: Fri Oct 12 09:54:33 2018
```

```
Last change: Fri Oct 12 09:54:30 2018 by root via cibadmin on z1.example.com
```

2 nodes configured

2 resources configured

Online: [ z1.example.com z2.example.com ]

Full list of resources:

Resource Group: apachegroup

ClusterIP (ocf::heartbeat:IPaddr2):	Started z1.example.com
WebSite (ocf::heartbeat:apache):	Started z1.example.com

PCSD Status:

z1.example.com: Online

z2.example.com: Online

...

Note that in this instance, the **apachegroup** service is running on node z1.example.com.

9. Access the website you created, stop the service on the node on which it is running, and note how the service fails over to the second node.
  - a. Point a browser to the website you created using the floating IP address you configured. This should display the text message you defined, displaying the name of the node on which the website is running.
  - b. Stop the apache web service. Using **killall -9** simulates an application-level crash.

```
killall -9 httpd
```

Check the cluster status. You should see that stopping the web service caused a failed action, but that the cluster software restarted the service on the node on which it had been running and you should still be able to access the web browser.

```
pcs status
```

Cluster name: my\_cluster

Stack: corosync

Current DC: z1.example.com (version 2.0.0-10.el8-b67d8d0de9) - partition with quorum

Last updated: Fri Oct 12 09:54:33 2018

Last change: Fri Oct 12 09:54:30 2018 by root via cibadmin on z1.example.com

2 nodes configured

2 resources configured

Online: [ z1.example.com z2.example.com ]

Full list of resources:

Resource Group: apachegroup

ClusterIP (ocf::heartbeat:IPaddr2):	Started z1.example.com
WebSite (ocf::heartbeat:apache):	Started z1.example.com

Failed Resource Actions:

```
* WebSite_monitor_60000 on z1.example.com 'not running' (7): call=31,
status=complete, exitreason='none',
last-rc-change='Fri Feb 5 21:01:41 2016', queued=0ms, exec=0ms
```

Clear the failure status once the service is up and running again.

```
pcs resource cleanup WebSite
```

- c. Put the node on which the service is running into standby mode. Note that since we have disabled fencing we can not effectively simulate a node-level failure (such as pulling a power cable) because fencing is required for the cluster to recover from such situations.

```
pcs node standby z1.example.com
```

- d. Check the status of the cluster and note where the service is now running.

```
pcs status
```

```
Cluster name: my_cluster
Stack: corosync
Current DC: z1.example.com (version 2.0.0-10.el8-b67d8d0de9) - partition with quorum
Last updated: Fri Oct 12 09:54:33 2018
Last change: Fri Oct 12 09:54:30 2018 by root via cibadmin on z1.example.com
```

```
2 nodes configured
```

```
2 resources configured
```

```
Node z1.example.com: standby
```

```
Online: [z2.example.com]
```

```
Full list of resources:
```

```
Resource Group: apachegroup
```

```
 ClusterIP (ocf::heartbeat:IPAddr2): Started z2.example.com
```

```
 WebSite (ocf::heartbeat:apache): Started z2.example.com
```

- e. Access the website. There should be no loss of service, although the display message should indicate the node on which the service is now running.

10. To restore cluster services to the first node, take the node out of standby mode. This will not necessarily move the service back to that node.

```
pcs cluster unstandby z1.example.com
```

11. For final cleanup, stop the cluster services on both nodes.

```
pcs cluster stop --all
```

# CHAPTER 91. THE PCS COMMAND LINE INTERFACE

The **pcs** command line interface controls and configures cluster services such as **corosync**, **pacemaker**, **booth**, and **sbd** by providing an easier interface to their configuration files.

Note that you should not edit the **cib.xml** configuration file directly. In most cases, Pacemaker will reject a directly modified **cib.xml** file.

## 91.1. PCS HELP DISPLAY

You can use the **-h** option of **pcs** to display the parameters of a **pcs** command and a description of those parameters. For example, the following command displays the parameters of the **pcs resource** command. Only a portion of the output is shown.

```
pcs resource -h
```

## 91.2. VIEWING THE RAW CLUSTER CONFIGURATION

Although you should not edit the cluster configuration file directly, you can view the raw cluster configuration with the **pcs cluster cib** command.

You can save the raw cluster configuration to a specified file with the **pcs cluster cib *filename*** command. If you have previously configured a cluster and there is already an active CIB, you use the following command to save the raw xml file.

```
pcs cluster cib filename
```

For example, the following command saves the raw xml from the CIB into a file named **testfile**.

```
pcs cluster cib testfile
```

## 91.3. SAVING A CONFIGURATION CHANGE TO A WORKING FILE

When configuring a cluster, you can save configuration changes to a specified file without affecting the active CIB. This allows you to specify configuration updates without immediately updating the currently running cluster configuration with each individual update.

For information on saving the CIB to a file, see [Viewing the raw cluster configuration](#). Once you have created that file, you can save configuration changes to that file rather than to the active CIB by using the **-f** option of the **pcs** command. When you have completed the changes and are ready to update the active CIB file, you can push those file updates with the **pcs cluster cib-push** command.

The following is the recommended procedure for pushing changes to the CIB file. This procedure creates a copy of the original saved CIB file and makes changes to that copy. When pushing those changes to the active CIB, this procedure specifies the **diff-against** option of the **pcs cluster cib-push** command so that only the changes between the original file and the updated file are pushed to the CIB. This allows users to make changes in parallel that do not overwrite each other, and it reduces the load on Pacemaker which does not need to parse the entire configuration file.

1. Save the active CIB to a file. This example saves the CIB to a file named **original.xml**.

```
pcs cluster cib original.xml
```

2. Copy the saved file to the working file you will be using for the configuration updates.

```
cp original.xml updated.xml
```

3. Update your configuration as needed. The following command creates a resource in the file **updated.xml** but does not add that resource to the currently running cluster configuration.

```
pcs -f updated.xml resource create VirtualIP ocf:heartbeat:IPaddr2 ip=192.168.0.120
op monitor interval=30s
```

4. Push the updated file to the active CIB, specifying that you are pushing only the changes you have made to the original file.

```
pcs cluster cib-push updated.xml diff-against=original.xml
```

Alternately, you can push the entire current content of a CIB file with the following command.

```
pcs cluster cib-push filename
```

When pushing the entire CIB file, Pacemaker checks the version and does not allow you to push a CIB file which is older than the one already in a cluster. If you need to update the entire CIB file with a version that is older than the one currently in the cluster, you can use the **--config** option of the **pcs cluster cib-push** command.

```
pcs cluster cib-push --config filename
```

## 91.4. DISPLAYING CLUSTER STATUS

You can display the status of the cluster and the cluster resources with the following command.

```
pcs status
```

You can display the status of a particular cluster component with the **commands** parameter of the **pcs status** command, specifying **resources**, **cluster**, **nodes**, or **pcsd**.

```
pcs status commands
```

For example, the following command displays the status of the cluster resources.

```
pcs status resources
```

The following command displays the status of the cluster, but not the cluster resources.

```
pcs cluster status
```

## 91.5. DISPLAYING THE FULL CLUSTER CONFIGURATION

Use the following command to display the full current cluster configuration.

```
pcs config
```

# CHAPTER 92. CREATING A RED HAT HIGH-AVAILABILITY CLUSTER WITH PACEMAKER

The following procedure creates a Red Hat High Availability two-node cluster using **pcs**.

Configuring the cluster in this example requires that your system include the following components:

- 2 nodes, which will be used to create the cluster. In this example, the nodes used are **z1.example.com** and **z2.example.com**.
- Network switches for the private network. We recommend but do not require a private network for communication among the cluster nodes and other cluster hardware such as network power switches and Fibre Channel switches.
- A fencing device for each node of the cluster. This example uses two ports of the APC power switch with a host name of **zapc.example.com**.

## 92.1. INSTALLING CLUSTER SOFTWARE

The following procedure installs the cluster software and configures your system for cluster creation.

1. On each node in the cluster, install the Red Hat High Availability Add-On software packages along with all available fence agents from the High Availability channel.

```
yum install pcs pacemaker fence-agents-all
```

Alternatively, you can install the Red Hat High Availability Add-On software packages along with only the fence agent that you require with the following command.

```
yum install pcs pacemaker fence-agents-model
```

The following command displays a list of the available fence agents.

```
rpm -q -a | grep fence
fence-agents-rhevm-4.0.2-3.el7.x86_64
fence-agents-ilo-mp-4.0.2-3.el7.x86_64
fence-agents-ipmilan-4.0.2-3.el7.x86_64
...
...
```



### WARNING

After you install the Red Hat High Availability Add-On packages, you should ensure that your software update preferences are set so that nothing is installed automatically. Installation on a running cluster can cause unexpected behaviors. For more information, see [Recommended Practices for Applying Software Updates to a RHEL High Availability or Resilient Storage Cluster](#).

2. If you are running the **firewalld** daemon, execute the following commands to enable the ports that are required by the Red Hat High Availability Add-On.



#### NOTE

You can determine whether the **firewalld** daemon is installed on your system with the **rpm -q firewalld** command. If it is installed, you can determine whether it is running with the **firewall-cmd --state** command.

```
firewall-cmd --permanent --add-service=high-availability
firewall-cmd --add-service=high-availability
```



#### NOTE

The ideal firewall configuration for cluster components depends on the local environment, where you may need to take into account such considerations as whether the nodes have multiple network interfaces or whether off-host firewalling is present. The example here, which opens the ports that are generally required by a Pacemaker cluster, should be modified to suit local conditions.

[Enabling ports for the High Availability Add-On](#) shows the ports to enable for the Red Hat High Availability Add-On and provides an explanation for what each port is used for.

3. In order to use **pcs** to configure the cluster and communicate among the nodes, you must set a password on each node for the user ID **hacluster**, which is the **pcs** administration account. It is recommended that the password for user **hacluster** be the same on each node.

```
passwd hacluster
```

Changing password for user hacluster.

New password:

Retype new password:

passwd: all authentication tokens updated successfully.

4. Before the cluster can be configured, the **pcsd** daemon must be started and enabled to start up on boot on each node. This daemon works with the **pcs** command to manage configuration across the nodes in the cluster.

On each node in the cluster, execute the following commands to start the **pcsd** service and to enable **pcsd** at system start.

```
systemctl start pcsd.service
systemctl enable pcsd.service
```

## 92.2. INSTALLING THE PCP-ZEROCONF PACKAGE (RECOMMENDED)

When you set up your cluster, it is recommended that you install the **pcp-zeroconf** package for the Performance Co-Pilot (PCP) tool. PCP is Red Hat’s recommended resource-monitoring tool for RHEL systems. Installing the **pcp-zeroconf** package allows you to have PCP running and collecting performance-monitoring data for the benefit of investigations into fencing, resource failures, and other events that disrupt the cluster.



## NOTE

Cluster deployments where PCP is enabled will need sufficient space available for PCP's captured data on the file system that contains `/var/log/pcp/`. Typical space usage by PCP varies across deployments, but 10Gb is usually sufficient when using the **pcp-zeroconf** default settings, and some environments may require less. Monitoring usage in this directory over a 14-day period of typical activity can provide a more accurate usage expectation.

To install the **pcp-zeroconf** package, run the following command.

```
yum install pcp-zeroconf
```

This package enables **pmcd** and sets up data capture at a 10-second interval.

For information on reviewing PCP data, see [Why did a RHEL High Availability cluster node reboot - and how can I prevent it from happening again?](#) on the Red Hat Customer Portal.

## 92.3. CREATING A HIGH AVAILABILITY CLUSTER

This procedure creates a Red Hat High Availability Add-On cluster that consists of the nodes **z1.example.com** and **z2.example.com**.

1. Authenticate the **pcs** user **hacluster** for each node in the cluster on the node from which you will be running **pcs**.

The following command authenticates user **hacluster** on **z1.example.com** for both of the nodes in a two-node cluster that will consist of **z1.example.com** and **z2.example.com**.

```
[root@z1 ~]# pcs host auth z1.example.com z2.example.com
Username: hacluster
Password:
z1.example.com: Authorized
z2.example.com: Authorized
```

2. Execute the following command from **z1.example.com** to create the two-node cluster **my\_cluster** that consists of nodes **z1.example.com** and **z2.example.com**. This will propagate the cluster configuration files to both nodes in the cluster. This command includes the **--start** option, which will start the cluster services on both nodes in the cluster.

```
[root@z1 ~]# pcs cluster setup my_cluster --start z1.example.com z2.example.com
```

3. Enable the cluster services to run on each node in the cluster when the node is booted.



## NOTE

For your particular environment, you may choose to leave the cluster services disabled by skipping this step. This allows you to ensure that if a node goes down, any issues with your cluster or your resources are resolved before the node rejoins the cluster. If you leave the cluster services disabled, you will need to manually start the services when you reboot a node by executing the **pcs cluster start** command on that node.

```
[root@z1 ~]# pcs cluster enable --all
```

You can display the current status of the cluster with the **pcs cluster status** command. Because there may be a slight delay before the cluster is up and running when you start the cluster services with the **--start** option of the **pcs cluster setup** command, you should ensure that the cluster is up and running before performing any subsequent actions on the cluster and its configuration.

```
[root@z1 ~]# pcs cluster status
Cluster Status:
Stack: corosync
Current DC: z2.example.com (version 2.0.0-10.el8-b67d8d0de9) - partition with quorum
Last updated: Thu Oct 11 16:11:18 2018
Last change: Thu Oct 11 16:11:00 2018 by hacluster via crmd on z2.example.com
2 Nodes configured
0 Resources configured
...
...
```

## 92.4. CREATING A HIGH AVAILABILITY CLUSTER WITH MULTIPLE LINKS

You can use the **pcs cluster setup** command to create a Red Hat High Availability cluster with multiple links by specifying all of the links for each node.

The format for the command to create a two-node cluster with two links is as follows.

```
pcs cluster setup cluster_name node1_name addr=node1_link0_address addr=node1_link1_address
node2_name addr=node2_link0_address addr=node2_link1_address
```

When creating a cluster with multiple links, you should take the following into account.

- The order of the **addr=address** parameters is important. The first address specified after a node name is for **link0**, the second one for **link1**, and so forth.
- It is possible to specify up to eight links using the **knet** transport protocol, which is the default transport protocol.
- All nodes must have the same number of **addr=** parameters.
- As of RHEL 8.1, it is possible to add, remove, and change links in an existing cluster using the **pcs cluster link add**, the **pcs cluster link remove**, the **pcs cluster link delete**, and the **pcs cluster link update** commands.
- As with single-link clusters, do not mix IPv4 and IPv6 addresses in one link, although you can have one link running IPv4 and the other running IPv6.
- As with single-link clusters, you can specify addresses as IP addresses or as names as long as the names resolve to IPv4 or IPv6 addresses for which IPv4 and IPv6 addresses are not mixed in one link.

The following example creates a two-node cluster named **my\_twolink\_cluster** with two nodes, **rh80-node1** and **rh80-node2**. **rh80-node1** has two interfaces, IP address 192.168.122.201 as **link0** and 192.168.123.201 as **link1**. **rh80-node2** has two interfaces, IP address 192.168.122.202 as **link0** and 192.168.123.202 as **link1**.

```
pcs cluster setup my_twolink_cluster rh80-node1 addr=192.168.122.201
addr=192.168.123.201 rh80-node2 addr=192.168.122.202 addr=192.168.123.202
```

For information on adding nodes to an existing cluster with multiple links, see [Adding a node to a cluster with multiple links](#).

For information on changing the links in an existing cluster with multiple links, see [Adding and modifying links in an existing cluster](#).

## 92.5. CONFIGURING FENCING

You must configure a fencing device for each node in the cluster. For information about the fence configuration commands and options, see [Configuring fencing in a Red Hat High Availability cluster](#).

For general information on fencing and its importance in a Red Hat High Availability cluster, see [Fencing in a Red Hat High Availability Cluster](#).



### NOTE

When configuring a fencing device, attention should be given to whether that device shares power with any nodes or devices in the cluster. If a node and its fence device do share power, then the cluster may be at risk of being unable to fence that node if the power to it and its fence device should be lost. Such a cluster should either have redundant power supplies for fence devices and nodes, or redundant fence devices that do not share power. Alternative methods of fencing such as SBD or storage fencing may also bring redundancy in the event of isolated power losses.

This example uses the APC power switch with a host name of **zapc.example.com** to fence the nodes, and it uses the **fence\_apc\_snmp** fencing agent. Because both nodes will be fenced by the same fencing agent, you can configure both fencing devices as a single resource, using the **pcmk\_host\_map** option.

You create a fencing device by configuring the device as a **stonith** resource with the **pcs stonith create** command. The following command configures a **stonith** resource named **myapc** that uses the **fence\_apc\_snmp** fencing agent for nodes **z1.example.com** and **z2.example.com**. The **pcmk\_host\_map** option maps **z1.example.com** to port 1, and **z2.example.com** to port 2. The login value and password for the APC device are both **apc**. By default, this device will use a monitor interval of sixty seconds for each node.

Note that you can use an IP address when specifying the host name for the nodes.

```
[root@z1 ~]# pcs stonith create myapc fence_apc_snmp \
ipaddr="zapc.example.com" pcmk_host_map="z1.example.com:1;z2.example.com:2" \
login="apc" passwd="apc"
```

The following command displays the parameters of an existing STONITH device.

```
[root@rh7-1 ~]# pcs stonith config myapc
Resource: myapc (class=stonith type=fence_apc_snmp)
 Attributes: ipaddr=zapc.example.com pcmk_host_map=z1.example.com:1;z2.example.com:2
 login=apc passwd=apc
 Operations: monitor interval=60s (myapc-monitor-interval-60s)
```

After configuring your fence device, you should test the device. For information on testing a fence device, see [Testing a fence device](#).



#### NOTE

Do not test your fence device by disabling the network interface, as this will not properly test fencing.



#### NOTE

Once fencing is configured and a cluster has been started, a network restart will trigger fencing for the node which restarts the network even when the timeout is not exceeded. For this reason, do not restart the network service while the cluster service is running because it will trigger unintentional fencing on the node.

## 92.6. BACKING UP AND RESTORING A CLUSTER CONFIGURATION

You can back up the cluster configuration in a tarball with the following command. If you do not specify a file name, the standard output will be used.

```
pcs config backup filename
```



#### NOTE

The **pcs config backup** command backs up only the cluster configuration itself as configured in the CIB; the configuration of resource daemons is out of the scope of this command. For example if you have configured an Apache resource in the cluster, the resource settings (which are in the CIB) will be backed up, while the Apache daemon settings (as set in `/etc/httpd`) and the files it serves will not be backed up. Similarly, if there is a database resource configured in the cluster, the database itself will not be backed up, while the database resource configuration (CIB) will be.

Use the following command to restore the cluster configuration files on all nodes from the backup. If you do not specify a file name, the standard input will be used. Specifying the **--local** option restores only the files on the current node.

```
pcs config restore [--local] [filename]
```

## 92.7. ENABLING PORTS FOR THE HIGH AVAILABILITY ADD-ON

The ideal firewall configuration for cluster components depends on the local environment, where you may need to take into account such considerations as whether the nodes have multiple network interfaces or whether off-host firewalling is present.

If you are running the **firewalld** daemon, execute the following commands to enable the ports that are required by the Red Hat High Availability Add-On.

```
firewall-cmd --permanent --add-service=high-availability
firewall-cmd --add-service=high-availability
```

You may need to modify which ports are open to suit local conditions.



## NOTE

You can determine whether the **firewalld** daemon is installed on your system with the **rpm -q firewalld** command. If the **firewalld** daemon is installed, you can determine whether it is running with the **firewall-cmd --state** command.

[Table 92.1, “Ports to Enable for High Availability Add-On”](#) shows the ports to enable for the Red Hat High Availability Add-On and provides an explanation for what the port is used for.

**Table 92.1. Ports to Enable for High Availability Add-On**

Port	When Required
TCP 2224	<p>Default <b>pcsd</b> port required on all nodes (needed by the <b>pcsd</b> Web UI and required for node-to-node communication). You can configure the <b>pcsd</b> port by means of the <b>PCSD_PORT</b> parameter in the <b>/etc/sysconfig/pcsd</b> file.</p> <p>It is crucial to open port 2224 in such a way that <b>pcs</b> from any node can talk to all nodes in the cluster, including itself. When using the Booth cluster ticket manager or a quorum device you must open port 2224 on all related hosts, such as Booth arbiters or the quorum device host.</p>
TCP 3121	<p>Required on all nodes if the cluster has any Pacemaker Remote nodes</p> <p>Pacemaker’s <b>pacemaker-based</b> daemon on the full cluster nodes will contact the <b>pacemaker_remoted</b> daemon on Pacemaker Remote nodes at port 3121. If a separate interface is used for cluster communication, the port only needs to be open on that interface. At a minimum, the port should open on Pacemaker Remote nodes to full cluster nodes. Because users may convert a host between a full node and a remote node, or run a remote node inside a container using the host’s network, it can be useful to open the port to all nodes. It is not necessary to open the port to any hosts other than nodes.</p>
TCP 5403	<p>Required on the quorum device host when using a quorum device with <b>corosync-qnetd</b>. The default value can be changed with the <b>-p</b> option of the <b>corosync-qnetd</b> command.</p>
UDP 5404-5412	<p>Required on corosync nodes to facilitate communication between nodes. It is crucial to open ports 5404-5412 in such a way that <b>corosync</b> from any node can talk to all nodes in the cluster, including itself.</p>

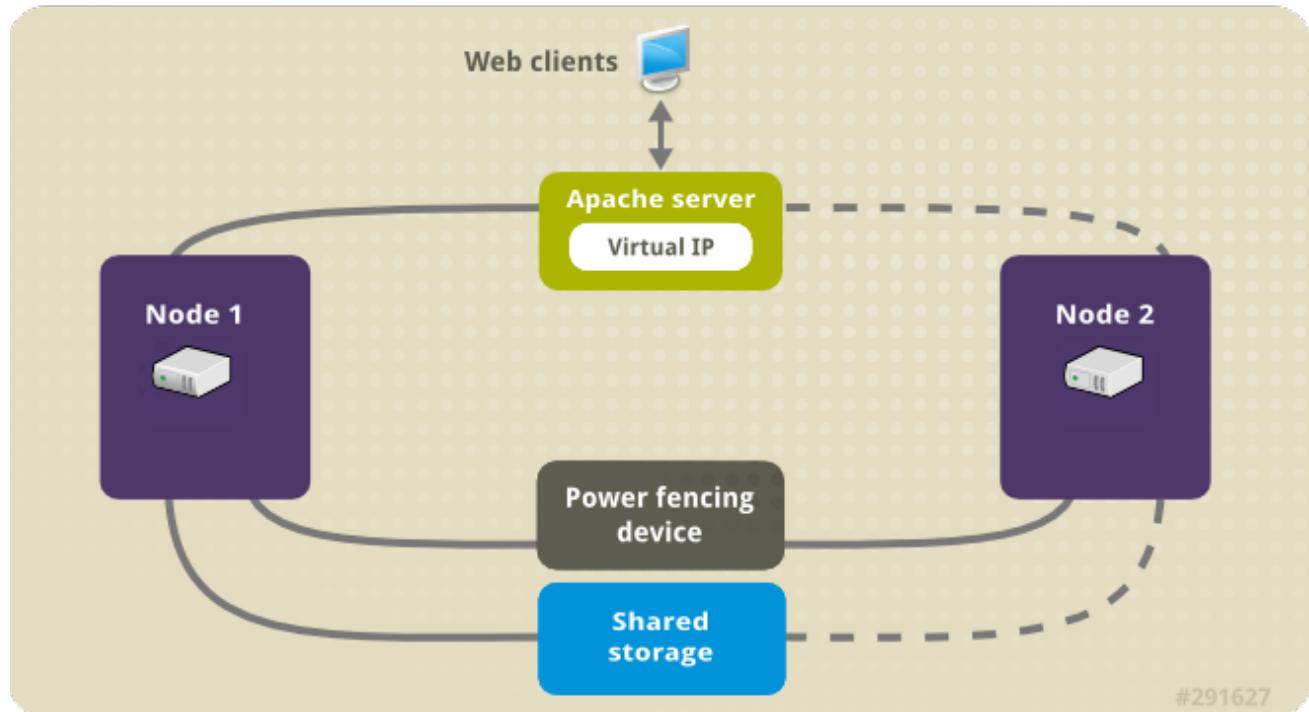
Port	When Required
TCP 21064	Required on all nodes if the cluster contains any resources requiring DLM (such as <b>GFS2</b> ).
TCP 9929, UDP 9929	Required to be open on all cluster nodes and booth arbitrator nodes to connections from any of those same nodes when the Booth ticket manager is used to establish a multi-site cluster.

# CHAPTER 93. CONFIGURING AN ACTIVE/PASSIVE APACHE HTTP SERVER IN A RED HAT HIGH AVAILABILITY CLUSTER

The following procedure configures an active/passive Apache HTTP server in a two-node Red Hat Enterprise Linux High Availability Add-On cluster using **pcs** to configure cluster resources. In this use case, clients access the Apache HTTP server through a floating IP address. The web server runs on one of two nodes in the cluster. If the node on which the web server is running becomes inoperative, the web server starts up again on the second node of the cluster with minimal service interruption.

[Figure 93.1, “Apache in a Red Hat High Availability Two-Node Cluster”](#) shows a high-level overview of the cluster in which The cluster is a two-node Red Hat High Availability cluster which is configured with a network power switch and with shared storage. The cluster nodes are connected to a public network, for client access to the Apache HTTP server through a virtual IP. The Apache server runs on either Node 1 or Node 2, each of which has access to the storage on which the Apache data is kept. In this illustration, the web server is running on Node 1 while Node 2 is available to run the server if Node 1 becomes inoperative.

Figure 93.1. Apache in a Red Hat High Availability Two-Node Cluster



This use case requires that your system include the following components:

- A two-node Red Hat High Availability cluster with power fencing configured for each node. We recommend but do not require a private network. This procedure uses the cluster example provided in [Creating a Red Hat High-Availability cluster with Pacemaker](#) .
- A public virtual IP address, required for Apache.
- Shared storage for the nodes in the cluster, using iSCSI or Fibre Channel.

The cluster is configured with an Apache resource group, which contains the cluster components that the web server requires: an LVM resource, a file system resource, an IP address resource, and a web server resource. This resource group can fail over from one node of the cluster to the other, allowing either node to run the web server. Before creating the resource group for this cluster, you will be performing the following procedures:

1. Configure an **ext4** file system on the logical volume **my\_lv**.
2. Configure a web server.

After performing these steps, you create the resource group and the resources it contains.

## 93.1. CONFIGURING AN LVM VOLUME WITH AN EXT4 FILE SYSTEM IN A PACEMAKER CLUSTER

This use case requires that you create an LVM logical volume on storage that is shared between the nodes of the cluster.



### NOTE

LVM volumes and the corresponding partitions and devices used by cluster nodes must be connected to the cluster nodes only.

The following procedure creates an LVM logical volume and then creates an ext4 file system on that volume for use in a Pacemaker cluster. In this example, the shared partition **/dev/sdb1** is used to store the LVM physical volume from which the LVM logical volume will be created.

1. On both nodes of the cluster, perform the following steps to set the value for the LVM system ID to the value of the **uname** identifier for the system. The LVM system ID will be used to ensure that only the cluster is capable of activating the volume group.
  - a. Set the **system\_id\_source** configuration option in the **/etc/lvm/lvm.conf** configuration file to **uname**.
 

```
Configuration option global/system_id_source.
system_id_source = "uname"
```
  - b. Verify that the LVM system ID on the node matches the **uname** for the node.
 

```
lvm systemid
system ID: z1.example.com
uname -n
z1.example.com
```

2. Create the LVM volume and create an ext4 file system on that volume. Since the **/dev/sdb1** partition is storage that is shared, you perform this part of the procedure on one node only.

- a. Create an LVM physical volume on partition **/dev/sdb1**.

```
pvcreate /dev/sdb1
Physical volume "/dev/sdb1" successfully created
```

- b. Create the volume group **my\_vg** that consists of the physical volume **/dev/sdb1**.

```
vgcreate my_vg /dev/sdb1
Volume group "my_vg" successfully created
```

- c. Verify that the new volume group has the system ID of the node on which you are running and from which you created the volume group.

```
vgs -o+systemid
VG #PV #LV #SN Attr VSize VFree System ID
my_vg 1 0 0 wz--n- <1.82t <1.82t z1.example.com
```

- d. Create a logical volume using the volume group **my\_vg**.

```
lvcreate -L450 -n my_lv my_vg
Rounding up size to full physical extent 452.00 MiB
Logical volume "my_lv" created
```

You can use the **lvs** command to display the logical volume.

```
lvs
LV VG Attr LSize Pool Origin Data% Move Log Copy% Convert
my_lv my_vg -wi-a--- 452.00m
...
```

- e. Create an ext4 file system on the logical volume **my\_lv**.

```
mkfs.ext4 /dev/my_vg/my_lv
mke2fs 1.44.3 (10-July-2018)
Creating filesystem with 462848 1k blocks and 115824 inodes
...
```

## 93.2. CONFIGURING AN APACHE HTTP SERVER

The following procedure configures an Apache HTTP Server.

1. Ensure that the Apache HTTP Server is installed on each node in the cluster. You also need the **wget** tool installed on the cluster to be able to check the status of the Apache HTTP Server. On each node, execute the following command.

```
yum install -y httpd wget
```

If you are running the **firewalld** daemon, on each node in the cluster enable the ports that are required by the Red Hat High Availability Add-On.

```
firewall-cmd --permanent --add-service=high-availability
firewall-cmd --reload
```

2. In order for the Apache resource agent to get the status of the Apache HTTP Server, ensure that the following text is present in the **/etc/httpd/conf/httpd.conf** file on each node in the cluster, and ensure that it has not been commented out. If this text is not already present, add the text to the end of the file.

```
<Location /server-status>
 SetHandler server-status
 Require local
</Location>
```

3. When you use the **apache** resource agent to manage Apache, it does not use **systemd**. Because of this, you must edit the **logrotate** script supplied with Apache so that it does not use **systemctl** to reload Apache.
- Remove the following line in the **/etc/logrotate.d/httpd** file on each node in the cluster.

```
/bin/systemctl reload httpd.service > /dev/null 2>/dev/null || true
```

Replace the line you removed with the following line.

```
/usr/sbin/httpd -f /etc/httpd/conf/httpd.conf -c "PidFile /var/run/httpd.pid" -k graceful > /dev/null 2>/dev/null || true
```

4. Create a web page for Apache to serve up. On one node in the cluster, mount the file system you created in [Configuring an LVM volume with an ext4 file system](#), create the file **index.html** on that file system, and then unmount the file system.

```
mount /dev/my_vg/my_lv /var/www/
mkdir /var/www/html
mkdir /var/www/cgi-bin
mkdir /var/www/error
restorecon -R /var/www
cat <<-END >/var/www/html/index.html
<html>
<body>Hello</body>
</html>
END
umount /var/www
```

### 93.3. CREATING THE RESOURCES AND RESOURCE GROUPS

This use case requires that you create four cluster resources. To ensure these resources all run on the same node, they are configured as part of the resource group **apache\_group**. The resources to create are as follows, listed in the order in which they will start.

1. An **LVM** resource named **my\_lvm** that uses the LVM volume group you created in [Configuring an LVM volume with an ext4 file system](#).
2. A **Filesystem** resource named **my\_fs**, that uses the file system device **/dev/my\_vg/my\_lv** you created in [Configuring an LVM volume with an ext4 file system](#).
3. An **IPaddr2** resource, which is a floating IP address for the **apache\_group** resource group. The IP address must not be one already associated with a physical node. If the **IPaddr2** resource's NIC device is not specified, the floating IP must reside on the same network as one of the node's statically assigned IP addresses, otherwise the NIC device to assign the floating IP address cannot be properly detected.
4. An **apache** resource named **Website** that uses the **index.html** file and the Apache configuration you defined in [Configuring an Apache HTTP server](#).

The following procedure creates the resource group **apache\_group** and the resources that the group contains. The resources will start in the order in which you add them to the group, and they will stop in the reverse order in which they are added to the group. Run this procedure from one node of the cluster only.

1. The following command creates the **LVM-activate** resource **my\_lvm**. Because the resource group **apache\_group** does not yet exist, this command creates the resource group.



### NOTE

Do not configure more than one **LVM-activate** resource that uses the same LVM volume group in an active/passive HA configuration, as this could cause data corruption. Additionally, do not configure an **LVM-activate** resource as a clone resource in an active/passive HA configuration.

```
[root@z1 ~]# pcs resource create my_lvm ocf:heartbeat:LVM-activate vgname=my_vg
vg_access_mode=system_id --group apache_group
```

When you create a resource, the resource is started automatically. You can use the following command to confirm that the resource was created and has started.

```
pcs resource status
Resource Group: apache_group
my_lvm (ocf::heartbeat:LVM-activate): Started
```

You can manually stop and start an individual resource with the **pcs resource disable** and **pcs resource enable** commands.

2. The following commands create the remaining resources for the configuration, adding them to the existing resource group **apache\_group**.

```
[root@z1 ~]# pcs resource create my_fs Filesystem \
device="/dev/my_vg/my_lv" directory="/var/www" fstype="ext4" \
--group apache_group

[root@z1 ~]# pcs resource create VirtualIP IPaddr2 ip=198.51.100.3 \
cidr_netmask=24 --group apache_group

[root@z1 ~]# pcs resource create Website apache \
configfile="/etc/httpd/conf/httpd.conf" \
statusurl="http://127.0.0.1/server-status" --group apache_group
```

3. After creating the resources and the resource group that contains them, you can check the status of the cluster. Note that all four resources are running on the same node.

```
[root@z1 ~]# pcs status
Cluster name: my_cluster
Last updated: Wed Jul 31 16:38:51 2013
Last change: Wed Jul 31 16:42:14 2013 via crm_attribute on z1.example.com
Stack: corosync
Current DC: z2.example.com (2) - partition with quorum
Version: 1.1.10-5.el7-9abe687
2 Nodes configured
6 Resources configured

Online: [z1.example.com z2.example.com]

Full list of resources:
myapc (stonith:fence_apc_snmp): Started z1.example.com
```

```
Resource Group: apachegroup
my_lvm (ocf::heartbeat:LVM): Started z1.example.com
my_fs (ocf::heartbeat:Filesystem): Started z1.example.com
VirtualIP (ocf::heartbeat:IPaddr2): Started z1.example.com
Website (ocf::heartbeat:apache): Started z1.example.com
```

Note that if you have not configured a fencing device for your cluster, by default the resources do not start.

- Once the cluster is up and running, you can point a browser to the IP address you defined as the **IPaddr2** resource to view the sample display, consisting of the simple word "Hello".

```
Hello
```

If you find that the resources you configured are not running, you can run the **pcs resource debug-start resource** command to test the resource configuration.

## 93.4. TESTING THE RESOURCE CONFIGURATION

In the cluster status display shown in [Creating the resources and resource groups](#), all of the resources are running on node **z1.example.com**. You can test whether the resource group fails over to node **z2.example.com** by using the following procedure to put the first node in **standby** mode, after which the node will no longer be able to host resources.

- The following command puts node **z1.example.com** in **standby** mode.

```
[root@z1 ~]# pcs node standby z1.example.com
```

- After putting node **z1** in **standby** mode, check the cluster status. Note that the resources should now all be running on **z2**.

```
[root@z1 ~]# pcs status
Cluster name: my_cluster
Last updated: Wed Jul 31 17:16:17 2013
Last change: Wed Jul 31 17:18:34 2013 via crm_attribute on z1.example.com
Stack: corosync
Current DC: z2.example.com (2) - partition with quorum
Version: 1.1.10-5.el7-9abe687
2 Nodes configured
6 Resources configured

Node z1.example.com (1): standby
Online: [z2.example.com]
```

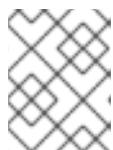
Full list of resources:

```
myapc (stonith:fence_apc_snmp): Started z1.example.com
Resource Group: apachegroup
my_lvm (ocf::heartbeat:LVM): Started z2.example.com
my_fs (ocf::heartbeat:Filesystem): Started z2.example.com
VirtualIP (ocf::heartbeat:IPaddr2): Started z2.example.com
Website (ocf::heartbeat:apache): Started z2.example.com
```

The web site at the defined IP address should still display, without interruption.

3. To remove **z1** from **standby** mode, enter the following command.

```
[root@z1 ~]# pcs cluster unstandby z1.example.com
```



#### NOTE

Removing a node from **standby** mode does not in itself cause the resources to fail back over to that node.

# CHAPTER 94. CONFIGURING AN ACTIVE/PASSIVE NFS SERVER IN A RED HAT HIGH AVAILABILITY CLUSTER

The following procedure configures a highly available active/passive NFS server on a two-node Red Hat Enterprise Linux High Availability Add-On cluster using shared storage. The procedure uses **pcs** to configure Pacemaker cluster resources. In this use case, clients access the NFS file system through a floating IP address. The NFS server runs on one of two nodes in the cluster. If the node on which the NFS server is running becomes inoperative, the NFS server starts up again on the second node of the cluster with minimal service interruption.

## 94.1. PREREQUISITES

This use case requires that your system include the following components:

- A two-node Red Hat High Availability cluster with power fencing configured for each node. We recommend but do not require a private network. This procedure uses the cluster example provided in [Creating a Red Hat High-Availability cluster with Pacemaker](#) .
- A public virtual IP address, required for the NFS server.
- Shared storage for the nodes in the cluster, using iSCSI or Fibre Channel.

## 94.2. PROCEDURAL OVERVIEW

Configuring a highly available active/passive NFS server on an existing two-node Red Hat Enterprise Linux High Availability cluster requires that you perform the following steps:

1. Configure an **ext4** file system on the LVM logical volume **my\_lv** on the shared storage for the nodes in the cluster.
2. Configure an NFS share on the shared storage on the LVM logical volume.
3. Create the cluster resources.
4. Test the NFS server you have configured.

## 94.3. CONFIGURING AN LVM VOLUME WITH AN EXT4 FILE SYSTEM IN A PACEMAKER CLUSTER

This use case requires that you create an LVM logical volume on storage that is shared between the nodes of the cluster.



### NOTE

LVM volumes and the corresponding partitions and devices used by cluster nodes must be connected to the cluster nodes only.

The following procedure creates an LVM logical volume and then creates an ext4 file system on that volume for use in a Pacemaker cluster. In this example, the shared partition **/dev/sdb1** is used to store the LVM physical volume from which the LVM logical volume will be created.

- On both nodes of the cluster, perform the following steps to set the value for the LVM system ID to the value of the **uname** identifier for the system. The LVM system ID will be used to ensure that only the cluster is capable of activating the volume group.
  - Set the **system\_id\_source** configuration option in the **/etc/lvm/lvm.conf** configuration file to **uname**.

```
Configuration option global/system_id_source.
system_id_source = "uname"
```

- Verify that the LVM system ID on the node matches the **uname** for the node.

```
lvm systemid
system ID: z1.example.com
uname -n
z1.example.com
```

- Create the LVM volume and create an ext4 file system on that volume. Since the **/dev/sdb1** partition is storage that is shared, you perform this part of the procedure on one node only.

- Create an LVM physical volume on partition **/dev/sdb1**.

```
pvcreate /dev/sdb1
Physical volume "/dev/sdb1" successfully created
```

- Create the volume group **my\_vg** that consists of the physical volume **/dev/sdb1**.

```
vgcreate my_vg /dev/sdb1
Volume group "my_vg" successfully created
```

- Verify that the new volume group has the system ID of the node on which you are running and from which you created the volume group.

```
vgs -o+systemid
VG #PV #LV #SN Attr VSize VFree System ID
my_vg 1 0 0 wz--n- <1.82t <1.82t z1.example.com
```

- Create a logical volume using the volume group **my\_vg**.

```
lvcreate -L450 -n my_lv my_vg
Rounding up size to full physical extent 452.00 MiB
Logical volume "my_lv" created
```

You can use the **lvs** command to display the logical volume.

```
lvs
LV VG Attr LSize Pool Origin Data% Move Log Copy% Convert
my_lv my_vg -wi-a--- 452.00m
...
```

- Create an ext4 file system on the logical volume **my\_lv**.

```
mkfs.ext4 /dev/my_vg/my_lv
mke2fs 1.44.3 (10-July-2018)
```

```
Creating filesystem with 462848 1k blocks and 115824 inodes
```

```
...
```

## 94.4. CONFIGURING AN NFS SHARE

The following procedure configures the NFS share for the NFS service failover.

1. On both nodes in the cluster, create the **/nfsshare** directory.

```
mkdir /nfsshare
```

2. On one node in the cluster, perform the following procedure.

- a. Mount the ext4 file system that you created in [Configuring an LVM volume with an ext4 file system](#) on the **/nfsshare** directory.

```
[root@z1 ~]# mount /dev/my_vg/my_lv /nfsshare
```

- b. Create an **exports** directory tree on the **/nfsshare** directory.

```
[root@z1 ~]# mkdir -p /nfsshare/exports
[root@z1 ~]# mkdir -p /nfsshare/exports/export1
[root@z1 ~]# mkdir -p /nfsshare/exports/export2
```

- c. Place files in the **exports** directory for the NFS clients to access. For this example, we are creating test files named **clientdatafile1** and **clientdatafile2**.

```
[root@z1 ~]# touch /nfsshare/exports/export1/clientdatafile1
[root@z1 ~]# touch /nfsshare/exports/export2/clientdatafile2
```

- d. Unmount the ext4 file system and deactivate the LVM volume group.

```
[root@z1 ~]# umount /dev/my_vg/my_lv
[root@z1 ~]# vgchange -an my_vg
```

## 94.5. CONFIGURING THE RESOURCES AND RESOURCE GROUP FOR AN NFS SERVER IN A CLUSTER

This section provides the procedure for configuring the cluster resources for this use case.



### NOTE

If you have not configured a fencing device for your cluster, by default the resources do not start.

If you find that the resources you configured are not running, you can run the **pcs resource debug-start resource** command to test the resource configuration. This starts the service outside of the cluster's control and knowledge. At the point the configured resources are running again, run **pcs resource cleanup resource** to make the cluster aware of the updates.

The following procedure configures the system resources. To ensure these resources all run on the

same node, they are configured as part of the resource group **nfsgroup**. The resources will start in the order in which you add them to the group, and they will stop in the reverse order in which they are added to the group. Run this procedure from one node of the cluster only.

1. Create the LVM-activate resource named **my\_lvm**. Because the resource group **nfsgroup** does not yet exist, this command creates the resource group.



### WARNING

Do not configure more than one **LVM-activate** resource that uses the same LVM volume group in an active/passive HA configuration, as this risks data corruption. Additionally, do not configure an **LVM-activate** resource as a clone resource in an active/passive HA configuration.

```
[root@z1 ~]# pcs resource create my_lvm ocf:heartbeat:LVM-activate vgname=my_vg
vg_access_mode=system_id --group nfsgroup
```

2. Check the status of the cluster to verify that the resource is running.

```
root@z1 ~]# pcs status
Cluster name: my_cluster
Last updated: Thu Jan 8 11:13:17 2015
Last change: Thu Jan 8 11:13:08 2015
Stack: corosync
Current DC: z2.example.com (2) - partition with quorum
Version: 1.1.12-a14efad
2 Nodes configured
3 Resources configured
```

Online: [ z1.example.com z2.example.com ]

Full list of resources:

```
myapc (stonith:fence_apc_snmp): Started z1.example.com
Resource Group: nfsgroup
 my_lvm (ocf::heartbeat:LVM): Started z1.example.com
```

PCSD Status:

```
z1.example.com: Online
z2.example.com: Online
```

Daemon Status:

```
corosync: active/enabled
pacemaker: active/enabled
pcsd: active/enabled
```

3. Configure a **Filesystem** resource for the cluster.

The following command configures an ext4 **Filesystem** resource named **nfsshare** as part of the **nfsgroup** resource group. This file system uses the LVM volume group and ext4 file system you created in [Configuring an LVM volume with an ext4 file system](#) and will be mounted on the **/nfsshare** directory you created in [Configuring an NFS share](#).

```
[root@z1 ~]# pcs resource create nfsshare Filesystem \
device=/dev/my_vg/my_lv directory=/nfsshare \
fstype=ext4 --group nfsgroup
```

You can specify mount options as part of the resource configuration for a **Filesystem** resource with the **options=options** parameter. Run the **pcs resource describe Filesystem** command for full configuration options.

4. Verify that the **my\_lvm** and **nfsshare** resources are running.

```
[root@z1 ~]# pcs status
...
Full list of resources:
myapc (stonith:fence_apc_snmp): Started z1.example.com
Resource Group: nfsgroup
 my_lvm (ocf::heartbeat:LVM): Started z1.example.com
 nfsshare (ocf::heartbeat:Filesystem): Started z1.example.com
...
...
```

5. Create the **nfsserver** resource named **nfs-daemon** as part of the resource group **nfsgroup**.

#### NOTE

The **nfsserver** resource allows you to specify an **nfs\_shared\_infodir** parameter, which is a directory that NFS servers use to store NFS-related stateful information.

It is recommended that this attribute be set to a subdirectory of one of the **Filesystem** resources you created in this collection of exports. This ensures that the NFS servers are storing their stateful information on a device that will become available to another node if this resource group needs to relocate. In this example;

- **/nfsshare** is the shared-storage directory managed by the **Filesystem** resource
- **/nfsshare/exports/export1** and **/nfsshare/exports/export2** are the export directories
- **/nfsshare/nfsinfo** is the shared-information directory for the **nfsserver** resource

```
[root@z1 ~]# pcs resource create nfs-daemon nfsserver \
nfs_shared_infodir=/nfsshare/nfsinfo nfs_no_notify=true \
--group nfsgroup
```

```
[root@z1 ~]# pcs status
...
```

6. Add the **exports** resources to export the **/nfsshare/exports** directory. These resources are part of the resource group **nfsgroup**. This builds a virtual directory for NFSv4 clients. NFSv3 clients can access these exports as well.



## NOTE

The **fsid=0** option is required only if you want to create a virtual directory for NFSv4 clients. For more information, see [How do I configure the fsid option in an NFS server's /etc/exports file?](#).

```
[root@z1 ~]# pcs resource create nfs-root exports \
clientspec=192.168.122.0/255.255.255.0 \
options=rw,sync,no_root_squash \
directory=/nfsshare/exports \
fsid=0 --group nfsgroup

[root@z1 ~]# # pcs resource create nfs-export1 exports \
clientspec=192.168.122.0/255.255.255.0 \
options=rw,sync,no_root_squash directory=/nfsshare/exports/export1 \
fsid=1 --group nfsgroup

[root@z1 ~]# # pcs resource create nfs-export2 exports \
clientspec=192.168.122.0/255.255.255.0 \
options=rw,sync,no_root_squash directory=/nfsshare/exports/export2 \
fsid=2 --group nfsgroup
```

7. Add the floating IP address resource that NFS clients will use to access the NFS share. This resource is part of the resource group **nfsgroup**. For this example deployment, we are using 192.168.122.200 as the floating IP address.

```
[root@z1 ~]# pcs resource create nfs_ip IPaddr2 \
ip=192.168.122.200 cidr_netmask=24 --group nfsgroup
```

8. Add an **nfsnotify** resource for sending NFSv3 reboot notifications once the entire NFS deployment has initialized. This resource is part of the resource group **nfsgroup**.



## NOTE

For the NFS notification to be processed correctly, the floating IP address must have a host name associated with it that is consistent on both the NFS servers and the NFS client.

```
[root@z1 ~]# pcs resource create nfs-notify nfsnotify \
source_host=192.168.122.200 --group nfsgroup
```

9. After creating the resources and the resource constraints, you can check the status of the cluster. Note that all resources are running on the same node.

```
[root@z1 ~]# pcs status
...
Full list of resources:
myapc (stonith:fence_apc_snmp): Started z1.example.com
Resource Group: nfsgroup
 my_lvm (ocf::heartbeat:LVM): Started z1.example.com
 nfsshare (ocf::heartbeat:Filesystem): Started z1.example.com
 nfs-daemon (ocf::heartbeat:nfsserver): Started z1.example.com
 nfs-root (ocf::heartbeat:exportfs): Started z1.example.com
```

```

nfs-export1 (ocf::heartbeat:exportfs): Started z1.example.com
nfs-export2 (ocf::heartbeat:exportfs): Started z1.example.com
nfs_ip (ocf::heartbeat:IPaddr2): Started z1.example.com
nfs-notify (ocf::heartbeat:nfsnotify): Started z1.example.com
...

```

## 94.6. TESTING THE NFS RESOURCE CONFIGURATION

You can validate your system configuration with the following procedures. You should be able to mount the exported file system with either NFSv3 or NFSv4.

### 94.6.1. Testing the NFS export

1. On a node outside of the cluster, residing in the same network as the deployment, verify that the NFS share can be seen by mounting the NFS share. For this example, we are using the 192.168.122.0/24 network.

```

showmount -e 192.168.122.200
Export list for 192.168.122.200:
/nfsshare/exports/export1 192.168.122.0/255.255.255.0
/nfsshare/exports 192.168.122.0/255.255.255.0
/nfsshare/exports/export2 192.168.122.0/255.255.255.0

```

2. To verify that you can mount the NFS share with NFSv4, mount the NFS share to a directory on the client node. After mounting, verify that the contents of the export directories are visible. Unmount the share after testing.

```

mkdir nfsshare
mount -o "vers=4" 192.168.122.200:export1 nfsshare
ls nfsshare
clientdatafile1
umount nfsshare

```

3. Verify that you can mount the NFS share with NFSv3. After mounting, verify that the test file **clientdatafile1** is visible. Unlike NFSv4, since NFSv3 does not use the virtual file system, you must mount a specific export. Unmount the share after testing.

```

mkdir nfsshare
mount -o "vers=3" 192.168.122.200:/nfsshare/exports/export2 nfsshare
ls nfsshare
clientdatafile2
umount nfsshare

```

### 94.6.2. Testing for failover

1. On a node outside of the cluster, mount the NFS share and verify access to the **clientdatafile1** we created in [Configuring an NFS share](#)

```

mkdir nfsshare
mount -o "vers=4" 192.168.122.200:export1 nfsshare
ls nfsshare
clientdatafile1

```

- From a node within the cluster, determine which node in the cluster is running **nfsgroup**. In this example, **nfsgroup** is running on **z1.example.com**.

```
[root@z1 ~]# pcs status
...
Full list of resources:
myapc (stonith:fence_apc_snmp): Started z1.example.com
Resource Group: nfsgroup
 my_lvm (ocf::heartbeat:LVM): Started z1.example.com
 nfsshare (ocf::heartbeat:Filesystem): Started z1.example.com
 nfs-daemon (ocf::heartbeat:nfsserver): Started z1.example.com
 nfs-root (ocf::heartbeat:exportfs): Started z1.example.com
 nfs-export1 (ocf::heartbeat:exportfs): Started z1.example.com
 nfs-export2 (ocf::heartbeat:exportfs): Started z1.example.com
 nfs_ip (ocf::heartbeat:IPaddr2): Started z1.example.com
 nfs-notify (ocf::heartbeat:nfsnotify): Started z1.example.com
...
...
```

- From a node within the cluster, put the node that is running **nfsgroup** in standby mode.

```
[root@z1 ~]# pcs node standby z1.example.com
```

- Verify that **nfsgroup** successfully starts on the other cluster node.

```
[root@z1 ~]# pcs status
...
Full list of resources:
Resource Group: nfsgroup
 my_lvm (ocf::heartbeat:LVM): Started z2.example.com
 nfsshare (ocf::heartbeat:Filesystem): Started z2.example.com
 nfs-daemon (ocf::heartbeat:nfsserver): Started z2.example.com
 nfs-root (ocf::heartbeat:exportfs): Started z2.example.com
 nfs-export1 (ocf::heartbeat:exportfs): Started z2.example.com
 nfs-export2 (ocf::heartbeat:exportfs): Started z2.example.com
 nfs_ip (ocf::heartbeat:IPaddr2): Started z2.example.com
 nfs-notify (ocf::heartbeat:nfsnotify): Started z2.example.com
...
...
```

- From the node outside the cluster on which you have mounted the NFS share, verify that this outside node still continues to have access to the test file within the NFS mount.

```
ls nfsshare
clientdatafile1
```

Service will be lost briefly for the client during the failover but the client should recover it with no user intervention. By default, clients using NFSv4 may take up to 90 seconds to recover the mount; this 90 seconds represents the NFSv4 file lease grace period observed by the server on startup. NFSv3 clients should recover access to the mount in a matter of a few seconds.

- From a node within the cluster, remove the node that was initially running running **nfsgroup** from standby mode. This will not in itself move the cluster resources back to this node.

```
[root@z1 ~]# pcs cluster unstandby z1.example.com
```

# CHAPTER 95. GFS2 FILE SYSTEMS IN A CLUSTER

This section provides:

- A procedure to set up a Pacemaker cluster that includes GFS2 file systems
- A procedure to migrate RHEL 7 logical volumes that contain GFS2 file systems to a RHEL 8 cluster

## 95.1. CONFIGURING A GFS2 FILE SYSTEM IN A CLUSTER

This procedure is an outline of the steps required to set up a Pacemaker cluster that includes GFS2 file systems. This example creates three GFS2 file systems on three logical volumes.

As a prerequisite for this procedure, you must install and start the cluster software on all nodes and create a basic two-node cluster. You must also configure fencing for the cluster. For information on creating a Pacemaker cluster and configuring fencing for the cluster, see [Creating a Red Hat High-Availability cluster with Pacemaker](#).

1. On both nodes of the cluster, install the **lvm2-lockd**, **gfs2-utils**, and **dlm** packages. The **lvm2-lockd** package is part of the AppStream channel and the **gfs2-utils** and **dlm** packages are part of the Resilient Storage channel.

```
yum install lvm2-lockd gfs2-utils dlm
```

2. Set up a **dlm** resource. This is a required dependency for configuring a GFS2 file system in a cluster. This example creates the **dlm** resource as part of a resource group named **locking**.

```
[root@z1 ~]# pcs resource create dlm --group locking ocf:pacemaker:controld op monitor interval=30s on-fail=fence
```

3. Clone the **locking** resource group so that the resource group can be active on both nodes of the cluster.

```
[root@z1 ~]# pcs resource clone locking interleave=true
```

4. Set up an **lvmlockd** resource as part of the group **locking**.

```
[root@z1 ~]# pcs resource create lvmlockd --group locking ocf:heartbeat:lvmlockd op monitor interval=30s on-fail=fence
```

5. Check the status of the cluster to ensure that the **locking** resource group has started on both nodes of the cluster.

```
[root@z1 ~]# pcs status --full
Cluster name: my_cluster
[...]
Online: [z1.example.com (1) z2.example.com (2)]

Full list of resources:

smoke-apc (stonith:fence_apc): Started z1.example.com
Clone Set: locking-clone [locking]
```

```

Resource Group: locking:0
 dlm (ocf::pacemaker:controld): Started z1.example.com
 lvmlockd (ocf::heartbeat:lvmlockd): Started z1.example.com
Resource Group: locking:1
 dlm (ocf::pacemaker:controld): Started z2.example.com
 lvmlockd (ocf::heartbeat:lvmlockd): Started z2.example.com
Started: [z1.example.com z2.example.com]

```

- Verify that the **lvmlockd** daemon is running on both nodes of the cluster.

```

[root@z1 ~]# ps -ef | grep lvmlockd
root 12257 1 0 17:45 ? 00:00:00 lvmlockd -p /run/lvmlockd.pid -A 1 -g dlm
[root@z2 ~]# ps -ef | grep lvmlockd
root 12270 1 0 17:45 ? 00:00:00 lvmlockd -p /run/lvmlockd.pid -A 1 -g dlm

```

- On one node of the cluster, create two shared volume groups. One volume group will contain two GFS2 file systems, and the other volume group will contain one GFS2 file system. The following command creates the shared volume group **shared\_vg1** on **/dev/vdb**.

```

[root@z1 ~]# vgcreate --shared shared_vg1 /dev/vdb
Physical volume "/dev/vdb" successfully created.
Volume group "shared_vg1" successfully created
VG shared_vg1 starting dlm lockspace
Starting locking. Waiting until locks are ready...

```

The following command creates the shared volume group **shared\_vg2** on **/dev/vdc**.

```

[root@z1 ~]# vgcreate --shared shared_vg2 /dev/vdc
Physical volume "/dev/vdc" successfully created.
Volume group "shared_vg2" successfully created
VG shared_vg2 starting dlm lockspace
Starting locking. Waiting until locks are ready...

```

- On the second node in the cluster, start the lock manager for each of the shared volume groups.

```

[root@z2 ~]# vgchange --lock-start shared_vg1
VG shared_vg1 starting dlm lockspace
Starting locking. Waiting until locks are ready...
[root@z2 ~]# vgchange --lock-start shared_vg2
VG shared_vg2 starting dlm lockspace
Starting locking. Waiting until locks are ready...

```

- On one node in the cluster, create the shared logical volumes and format the volumes with a GFS2 file system. One journal is required for each node that mounts the file system. Ensure that you create enough journals for each of the nodes in your cluster.

```

[root@z1 ~]# lvcreate --activate sy -L5G -n shared_lv1 shared_vg1
Logical volume "shared_lv1" created.
[root@z1 ~]# lvcreate --activate sy -L5G -n shared_lv2 shared_vg1
Logical volume "shared_lv2" created.
[root@z1 ~]# lvcreate --activate sy -L5G -n shared_lv1 shared_vg2
Logical volume "shared_lv1" created.

[root@z1 ~]# mkfs.gfs2 -j2 -p lock_dlm -t my_cluster:gfs2-demo1

```

```
/dev/shared_vg1/shared_lv1
[root@z1 ~]# mkfs.gfs2 -j2 -p lock_dlm -t my_cluster:gfs2-demo2
/dev/shared_vg1/shared_lv2
[root@z1 ~]# mkfs.gfs2 -j2 -p lock_dlm -t my_cluster:gfs2-demo3
/dev/shared_vg2/shared_lv1
```

10. Create an **LVM-activate** resource for each logical volume to automatically activate that logical volume on all nodes.

- a. Create an **LVM-activate** resource named **sharedlv1** for the logical volume **shared\_lv1** in volume group **shared\_vg1**. This command also creates the resource group **shared\_vg1** that includes the resource. In this example, the resource group has the same name as the shared volume group that includes the logical volume.

```
[root@z1 ~]# pcs resource create sharedlv1 --group shared_vg1 ocf:heartbeat:LVM-activate lvname=shared_lv1 vgname=shared_vg1 activation_mode=shared vg_access_mode=lvmlockd
```

- b. Create an **LVM-activate** resource named **sharedlv2** for the logical volume **shared\_lv2** in volume group **shared\_vg1**. This resource will also be part of the resource group **shared\_vg1**.

```
[root@z1 ~]# pcs resource create sharedlv2 --group shared_vg1 ocf:heartbeat:LVM-activate lvname=shared_lv2 vgname=shared_vg1 activation_mode=shared vg_access_mode=lvmlockd
```

- c. Create an **LVM-activate** resource named **sharedlv3** for the logical volume **shared\_lv1** in volume group **shared\_vg2**. This command also creates the resource group **shared\_vg2** that includes the resource.

```
[root@z1 ~]# pcs resource create sharedlv3 --group shared_vg2 ocf:heartbeat:LVM-activate lvname=shared_lv1 vgname=shared_vg2 activation_mode=shared vg_access_mode=lvmlockd
```

11. Clone the two new resource groups.

```
[root@z1 ~]# pcs resource clone shared_vg1 interleave=true
[root@z1 ~]# pcs resource clone shared_vg2 interleave=true
```

12. Configure ordering constraints to ensure that the **locking** resource group that includes the **dlm** and **lvmlockd** resources starts first.

```
[root@z1 ~]# pcs constraint order start locking-clone then shared_vg1-clone
Adding locking-clone shared_vg1-clone (kind: Mandatory) (Options: first-action=start then-action=start)
[root@z1 ~]# pcs constraint order start locking-clone then shared_vg2-clone
Adding locking-clone shared_vg2-clone (kind: Mandatory) (Options: first-action=start then-action=start)
```

13. Configure colocation constraints to ensure that the **vg1** and **vg2** resource groups start on the same node as the **locking** resource group.

```
[root@z1 ~]# pcs constraint colocation add shared_vg1-clone with locking-clone
[root@z1 ~]# pcs constraint colocation add shared_vg2-clone with locking-clone
```

14. On both nodes in the cluster, verify that the logical volumes are active. There may be a delay of a few seconds.

```
[root@z1 ~]# lvs
 LV VG Attr LSize
 shared_lv1 shared_vg1 -wi-a---- 5.00g
 shared_lv2 shared_vg1 -wi-a---- 5.00g
 shared_lv1 shared_vg2 -wi-a---- 5.00g

[root@z2 ~]# lvs
 LV VG Attr LSize
 shared_lv1 shared_vg1 -wi-a---- 5.00g
 shared_lv2 shared_vg1 -wi-a---- 5.00g
 shared_lv1 shared_vg2 -wi-a---- 5.00g
```

15. Create a file system resource to automatically mount each GFS2 file system on all nodes. You should not add the file system to the **/etc/fstab** file because it will be managed as a Pacemaker cluster resource. Mount options can be specified as part of the resource configuration with **options=options**. Run the **pcs resource describe Filesystem** command for full configuration options.

The following commands create the file system resources. These commands add each resource to the resource group that includes the logical volume resource for that file system.

```
[root@z1 ~]# pcs resource create sharedfs1 --group shared_vg1
 ocf:heartbeat:Filesystem device="/dev/shared_vg1/shared_lv1" directory="/mnt/gfs1"
 fstype="gfs2" options=noatime op monitor interval=10s on-fail=fence
[root@z1 ~]# pcs resource create sharedfs2 --group shared_vg1
 ocf:heartbeat:Filesystem device="/dev/shared_vg1/shared_lv2" directory="/mnt/gfs2"
 fstype="gfs2" options=noatime op monitor interval=10s on-fail=fence
[root@z1 ~]# pcs resource create sharedfs3 --group shared_vg2
 ocf:heartbeat:Filesystem device="/dev/shared_vg2/shared_lv1" directory="/mnt/gfs3"
 fstype="gfs2" options=noatime op monitor interval=10s on-fail=fence
```

16. Verify that the GFS2 file systems are mounted on both nodes of the cluster.

```
[root@z1 ~]# mount | grep gfs2
/dev/mapper/shared_vg1-shared_lv1 on /mnt/gfs1 type gfs2 (rw,noatime,seclabel)
/dev/mapper/shared_vg1-shared_lv2 on /mnt/gfs2 type gfs2 (rw,noatime,seclabel)
/dev/mapper/shared_vg2-shared_lv1 on /mnt/gfs3 type gfs2 (rw,noatime,seclabel)

[root@z2 ~]# mount | grep gfs2
/dev/mapper/shared_vg1-shared_lv1 on /mnt/gfs1 type gfs2 (rw,noatime,seclabel)
/dev/mapper/shared_vg1-shared_lv2 on /mnt/gfs2 type gfs2 (rw,noatime,seclabel)
/dev/mapper/shared_vg2-shared_lv1 on /mnt/gfs3 type gfs2 (rw,noatime,seclabel)
```

17. Check the status of the cluster.

```
[root@z1 ~]# pcs status --full
Cluster name: my_cluster
[...1

Full list of resources:

smoke-apc (stonith:fence_apc): Started z1.example.com
```

```

Clone Set: locking-clone [locking]
 Resource Group: locking:0
 dlm (ocf::pacemaker:controld): Started z2.example.com
 lvmlockd (ocf::heartbeat:lvmlockd): Started z2.example.com
 Resource Group: locking:1
 dlm (ocf::pacemaker:controld): Started z1.example.com
 lvmlockd (ocf::heartbeat:lvmlockd): Started z1.example.com
 Started: [z1.example.com z2.example.com]
Clone Set: shared_vg1-clone [shared_vg1]
 Resource Group: shared_vg1:0
 sharedlv1 (ocf::heartbeat:LVM-activate): Started z2.example.com
 sharedlv2 (ocf::heartbeat:LVM-activate): Started z2.example.com
 shareddfs1 (ocf::heartbeat:Filesystem): Started z2.example.com
 shareddfs2 (ocf::heartbeat:Filesystem): Started z2.example.com
 Resource Group: shared_vg1:1
 sharedlv1 (ocf::heartbeat:LVM-activate): Started z1.example.com
 sharedlv2 (ocf::heartbeat:LVM-activate): Started z1.example.com
 shareddfs1 (ocf::heartbeat:Filesystem): Started z1.example.com
 shareddfs2 (ocf::heartbeat:Filesystem): Started example.co
 Started: [z1.example.com z2.example.com]
Clone Set: shared_vg2-clone [shared_vg2]
 Resource Group: shared_vg2:0
 sharedlv3 (ocf::heartbeat:LVM-activate): Started z2.example.com
 shareddfs3 (ocf::heartbeat:Filesystem): Started z2.example.com
 Resource Group: shared_vg2:1
 sharedlv3 (ocf::heartbeat:LVM-activate): Started z1.example.com
 shareddfs3 (ocf::heartbeat:Filesystem): Started z1.example.com
 Started: [z1.example.com z2.example.com]
...
```

## 95.2. MIGRATING A GFS2 FILE SYSTEM FROM RHEL7 TO RHEL8

In Red Hat Enterprise Linux 8, LVM uses the LVM lock daemon **lvmlockd** instead of **clvmd** for managing shared storage devices in an active/active cluster. This requires that you configure the logical volumes that your active/active cluster will require as shared logical volumes. Additionally, this requires that you use the **LVM-activate** resource to manage an LVM volume and that you use the **lvmlockd** resource agent to manage the **lvmlockd** daemon. See [Configuring a GFS2 file system in a cluster](#) for a full procedure for configuring a Pacemaker cluster that includes GFS2 file systems using shared logical volumes.

To use your existing Red Hat Enterprise Linux 7 logical volumes when configuring a RHEL8 cluster that includes GFS2 file systems, perform the following procedure from the RHEL8 cluster. In this example, the clustered RHEL 7 logical volume is part of the volume group **upgrade\_gfs\_vg**.



### NOTE

The RHEL8 cluster must have the same name as the RHEL7 cluster that includes the GFS2 file system in order for the existing file system to be valid.

1. Ensure that the logical volumes containing the GFS2 file systems are currently inactive. This procedure is safe only if all nodes have stopped using the volume group.
2. From one node in the cluster, forcibly change the volume group to be local.

```
[root@rhel8-01 ~]# vgchange --lock-type none --lock-opt force upgrade_gfs_vg
Forcibly change VG lock type to none? [y/n]: y
Volume group "upgrade_gfs_vg" successfully changed
```

3. From one node in the cluster, change the local volume group to a shared volume group

```
[root@rhel8-01 ~]# vgchange --lock-type dlm upgrade_gfs_vg
Volume group "upgrade_gfs_vg" successfully changed
```

4. On each node in the cluster, start locking for the volume group.

```
[root@rhel8-01 ~]# vgchange --lock-start upgrade_gfs_vg
VG upgrade_gfs_vg starting dlm lockspace
Starting locking. Waiting until locks are ready...
[root@rhel8-02 ~]# vgchange --lock-start upgrade_gfs_vg
VG upgrade_gfs_vg starting dlm lockspace
Starting locking. Waiting until locks are ready...
```

After performing this procedure, you can create an **LVM-activate** resource for each logical volume.

# CHAPTER 96. CONFIGURING FENCING IN A RED HAT HIGH AVAILABILITY CLUSTER

A node that is unresponsive may still be accessing data. The only way to be certain that your data is safe is to fence the node using STONITH. STONITH is an acronym for "Shoot The Other Node In The Head" and it protects your data from being corrupted by rogue nodes or concurrent access. Using STONITH, you can be certain that a node is truly offline before allowing the data to be accessed from another node.

STONITH also has a role to play in the event that a clustered service cannot be stopped. In this case, the cluster uses STONITH to force the whole node offline, thereby making it safe to start the service elsewhere.

For more complete general information on fencing and its importance in a Red Hat High Availability cluster, see [Fencing in a Red Hat High Availability Cluster](#).

You implement STONITH in a Pacemaker cluster by configuring fence devices for the nodes of the cluster.

## 96.1. DISPLAYING AVAILABLE FENCE AGENTS AND THEIR OPTIONS

Use the following command to view of list of all available STONITH agents. When you specify a filter, this command displays only the STONITH agents that match the filter.

```
pcs stonith list [filter]
```

Use the following command to view the options for the specified STONITH agent.

```
pcs stonith describe stonith_agent
```

For example, the following command displays the options for the fence agent for APC over telnet/SSH.

```
pcs stonith describe fence_apc
Stonith options for: fence_apc
 ipaddr (required): IP Address or Hostname
 login (required): Login Name
 passwd: Login password or passphrase
 passwd_script: Script to retrieve password
 cmd_prompt: Force command prompt
 secure: SSH connection
 port (required): Physical plug number or name of virtual machine
 identity_file: Identity file for ssh
 switch: Physical switch number on device
 inet4_only: Forces agent to use IPv4 addresses only
 inet6_only: Forces agent to use IPv6 addresses only
 ipport: TCP port to use for connection with device
 action (required): Fencing Action
 verbose: Verbose mode
 debug: Write debug information to given file
 version: Display version information and exit
 help: Display help and exit
 separator: Separator for CSV created by operation list
 power_timeout: Test X seconds for status change after ON/OFF
 shell_timeout: Wait X seconds for cmd prompt after issuing command
```

`login_timeout`: Wait X seconds for cmd prompt after login  
`power_wait`: Wait X seconds after issuing ON/OFF  
`delay`: Wait X seconds before fencing is started  
`retry_on`: Count of attempts to retry power on



### WARNING

For fence agents that provide a `method` option, a value of `cycle` is unsupported and should not be specified, as it may cause data corruption.

## 96.2. CREATING A FENCE DEVICE

The format for the command to create a stonith device is as follows. For a listing of the available stonith creation options, see the `pcs stonith -h` display.

```
pcs stonith create stonith_id stonith_device_type [stonith_device_options] [op operation_action operation_options]
```

The following command creates a single fencing device for a single node.

```
pcs stonith create MyStonith fence_virt pcmk_host_list=f1 op monitor interval=30s
```

Some fence devices can fence only a single node, while other devices can fence multiple nodes. The parameters you specify when you create a fencing device depend on what your fencing device supports and requires.

- Some fence devices can automatically determine what nodes they can fence.
- You can use the `pcmk_host_list` parameter when creating a fencing device to specify all of the machines that are controlled by that fencing device.
- Some fence devices require a mapping of host names to the specifications that the fence device understands. You can map host names with the `pcmk_host_map` parameter when creating a fencing device.

For information on the `pcmk_host_list` and `pcmk_host_map` parameters, see [General Properties of Fencing Devices](#).

After configuring a fence device, it is imperative that you test the device to ensure that it is working correctly. For information on testing a fence device, see [Testing a fence device](#).

## 96.3. GENERAL PROPERTIES OF FENCING DEVICES

Any cluster node can fence any other cluster node with any fence device, regardless of whether the fence resource is started or stopped. Whether the resource is started controls only the recurring monitor for the device, not whether it can be used, with the following exceptions:

- You can disable a fencing device by running the `pcs stonith disable stonith_id` command. This will prevent any node from using that device.

- To prevent a specific node from using a fencing device, you can configure location constraints for the fencing resource with the **pcs constraint location ... avoids** command.
- Configuring **stonith-enabled=false** will disable fencing altogether. Note, however, that Red Hat does not support clusters when fencing is disabled, as it is not suitable for a production environment.

[Table 96.1, “General Properties of Fencing Devices”](#) describes the general properties you can set for fencing devices.

**Table 96.1. General Properties of Fencing Devices**

Field	Type	Default	Description
<b>pcmk_host_map</b>	string		A mapping of host names to port numbers for devices that do not support host names. For example: <b>node1:1;node2:2,3</b> tells the cluster to use port 1 for node1 and ports 2 and 3 for node2
<b>pcmk_host_list</b>	string		A list of machines controlled by this device (Optional unless <b>pcmk_host_check=static-list</b> ).
<b>pcmk_host_check</b>	string	<ul style="list-style-type: none"> <li>* <b>static-list</b> if either <b>pcmk_host_list</b> or <b>pcmk_host_map</b> is set</li> <li>* Otherwise, <b>dynamic-list</b> if the fence device supports the <b>list</b> action</li> <li>* Otherwise, <b>status</b> if the fence device supports the <b>status</b> action</li> <li>*Otherwise, <b>none</b>.</li> </ul>	How to determine which machines are controlled by the device. Allowed values: <b>dynamic-list</b> (query the device), <b>static-list</b> (check the <b>pcmk_host_list</b> attribute), <b>none</b> (assume every device can fence every machine)

## 96.4. ADVANCED FENCING CONFIGURATION OPTIONS

[Table 96.2, “Advanced Properties of Fencing Devices”](#) summarizes additional properties you can set for fencing devices. Note that these properties are for advanced use only.

**Table 96.2. Advanced Properties of Fencing Devices**

Field	Type	Default	Description
<b>pcmk_host_argument</b>	string	port	An alternate parameter to supply instead of port. Some devices do not support the standard port parameter or may provide additional ones. Use this to specify an alternate, device-specific parameter that should indicate the machine to be fenced. A value of <b>none</b> can be used to tell the cluster not to supply any additional parameters.
<b>pcmk_reboot_action</b>	string	reboot	An alternate command to run instead of <b>reboot</b> . Some devices do not support the standard commands or may provide additional ones. Use this to specify an alternate, device-specific, command that implements the reboot action.
<b>pcmk_reboot_timeout</b>	time	60s	Specify an alternate timeout to use for reboot actions instead of <b>stonith-timeout</b> . Some devices need much more/less time to complete than normal. Use this to specify an alternate, device-specific, timeout for reboot actions.
<b>pcmk_reboot_retries</b>	integer	2	The maximum number of times to retry the <b>reboot</b> command within the timeout period. Some devices do not support multiple connections. Operations may fail if the device is busy with another task so Pacemaker will automatically retry the operation, if there is time remaining. Use this option to alter the number of times Pacemaker retries reboot actions before giving up.
<b>pcmk_off_action</b>	string	off	An alternate command to run instead of <b>off</b> . Some devices do not support the standard commands or may provide additional ones. Use this to specify an alternate, device-specific, command that implements the off action.
<b>pcmk_off_timeout</b>	time	60s	Specify an alternate timeout to use for off actions instead of <b>stonith-timeout</b> . Some devices need much more or much less time to complete than normal. Use this to specify an alternate, device-specific, timeout for off actions.

Field	Type	Default	Description
<b>pcmk_off_retries</b>	integer	2	The maximum number of times to retry the off command within the timeout period. Some devices do not support multiple connections. Operations may fail if the device is busy with another task so Pacemaker will automatically retry the operation, if there is time remaining. Use this option to alter the number of times Pacemaker retries off actions before giving up.
<b>pcmk_list_action</b>	string	list	An alternate command to run instead of <b>list</b> . Some devices do not support the standard commands or may provide additional ones. Use this to specify an alternate, device-specific, command that implements the list action.
<b>pcmk_list_timeout</b>	time	60s	Specify an alternate timeout to use for list actions. Some devices need much more or much less time to complete than normal. Use this to specify an alternate, device-specific, timeout for list actions.
<b>pcmk_list_retries</b>	integer	2	The maximum number of times to retry the <b>list</b> command within the timeout period. Some devices do not support multiple connections. Operations may fail if the device is busy with another task so Pacemaker will automatically retry the operation, if there is time remaining. Use this option to alter the number of times Pacemaker retries list actions before giving up.
<b>pcmk_monitor_action</b>	string	monitor	An alternate command to run instead of <b>monitor</b> . Some devices do not support the standard commands or may provide additional ones. Use this to specify an alternate, device-specific, command that implements the monitor action.
<b>pcmk_monitor_timeout</b>	time	60s	Specify an alternate timeout to use for monitor actions instead of <b>stonith-timeout</b> . Some devices need much more or much less time to complete than normal. Use this to specify an alternate, device-specific, timeout for monitor actions.

Field	Type	Default	Description
<b>pcmk_monitor_retries</b>	integer	2	The maximum number of times to retry the <b>monitor</b> command within the timeout period. Some devices do not support multiple connections. Operations may fail if the device is busy with another task so Pacemaker will automatically retry the operation, if there is time remaining. Use this option to alter the number of times Pacemaker retries monitor actions before giving up.
<b>pcmk_status_action</b>	string	status	An alternate command to run instead of <b>status</b> . Some devices do not support the standard commands or may provide additional ones. Use this to specify an alternate, device-specific, command that implements the status action.
<b>pcmk_status_timeout</b>	time	60s	Specify an alternate timeout to use for status actions instead of <b>stonith-timeout</b> . Some devices need much more or much less time to complete than normal. Use this to specify an alternate, device-specific, timeout for status actions.
<b>pcmk_status_retries</b>	integer	2	The maximum number of times to retry the status command within the timeout period. Some devices do not support multiple connections. Operations may fail if the device is busy with another task so Pacemaker will automatically retry the operation, if there is time remaining. Use this option to alter the number of times Pacemaker retries status actions before giving up.

Field	Type	Default	Description
<b>pcmk_delay_base</b>	time	0s	<p>Enable a base delay for stonith actions and specify a base delay value. In a cluster with an even number of nodes, configuring a delay can help avoid nodes fencing each other at the same time in an even split. A random delay can be useful when the same fence device is used for all nodes, and differing static delays can be useful on each fencing device when a separate device is used for each node. The overall delay is derived from a random delay value adding this static delay so that the sum is kept below the maximum delay. If you set <b>pcmk_delay_base</b> but do not set <b>pcmk_delay_max</b>, there is no random component to the delay and it will be the value of <b>pcmk_delay_base</b>.</p> <p>Some individual fence agents implement a "delay" parameter, which is independent of delays configured with a <b>pcmk_delay_*</b> property. If both of these delays are configured, they are added together and thus would generally not be used in conjunction.</p>
<b>pcmk_delay_max</b>	time	0s	<p>Enable a random delay for stonith actions and specify the maximum of random delay. In a cluster with an even number of nodes, configuring a delay can help avoid nodes fencing each other at the same time in an even split. A random delay can be useful when the same fence device is used for all nodes, and differing static delays can be useful on each fencing device when a separate device is used for each node. The overall delay is derived from this random delay value adding a static delay so that the sum is kept below the maximum delay. If you set <b>pcmk_delay_max</b> but do not set <b>pcmk_delay_base</b> there is no static component to the delay.</p> <p>Some individual fence agents implement a "delay" parameter, which is independent of delays configured with a <b>pcmk_delay_*</b> property. If both of these delays are configured, they are added together and thus would generally not be used in conjunction.</p>

Field	Type	Default	Description
<b>pcmk_action_limit</b>	integer	1	The maximum number of actions that can be performed in parallel on this device. The cluster property <b>concurrent-fencing=true</b> needs to be configured first (this is the default value for RHEL 8.1 and later). A value of -1 is unlimited.
<b>pcmk_on_action</b>	string	on	For advanced use only: An alternate command to run instead of <b>on</b> . Some devices do not support the standard commands or may provide additional ones. Use this to specify an alternate, device-specific, command that implements the <b>on</b> action.
<b>pcmk_on_timeout</b>	time	60s	For advanced use only: Specify an alternate timeout to use for <b>on</b> actions instead of <b>stonith-timeout</b> . Some devices need much more or much less time to complete than normal. Use this to specify an alternate, device-specific, timeout for <b>on</b> actions.
<b>pcmk_on_retries</b>	integer	2	For advanced use only: The maximum number of times to retry the <b>on</b> command within the timeout period. Some devices do not support multiple connections. Operations may <b>fail</b> if the device is busy with another task so Pacemaker will automatically retry the operation, if there is time remaining. Use this option to alter the number of times Pacemaker retries <b>on</b> actions before giving up.

In addition to the properties you can set for individual fence devices, there are also cluster properties you can set that determine fencing behavior, as described in [Table 96.3, “Cluster properties that determine fencing behavior”](#).

**Table 96.3. Cluster properties that determine fencing behavior**

Option	Default	Description

Option	Default	Description
<b>stonith-enabled</b>	true	<p>Indicates that failed nodes and nodes with resources that cannot be stopped should be fenced. Protecting your data requires that you set this <b>true</b>.</p> <p>If <b>true</b>, or unset, the cluster will refuse to start resources unless one or more STONITH resources have been configured also.</p> <p>Red Hat only supports clusters with this value set to <b>true</b>.</p>
<b>stonith-action</b>	reboot	Action to send to STONITH device. Allowed values: <b>reboot, off</b> . The value <b>poweroff</b> is also allowed, but is only used for legacy devices.
<b>stonith-timeout</b>	60s	How long to wait for a STONITH action to complete.
<b>stonith-max-attempts</b>	10	How many times fencing can fail for a target before the cluster will no longer immediately re-attempt it.
<b>stonith-watchdog-timeout</b>		The maximum time to wait until a node can be assumed to have been killed by the hardware watchdog. It is recommended that this value be set to twice the value of the hardware watchdog timeout. This option is needed only if watchdog-based SBD is used for fencing.
<b>concurrent-fencing</b>	true (RHEL 8.1 and later)	Allow fencing operations to be performed in parallel.

Option	Default	Description
<b>fence-reaction</b>	stop	<p>(Red Hat Enterprise Linux 8.2 and later)  Determines how a cluster node should react if notified of its own fencing. A cluster node may receive notification of its own fencing if fencing is misconfigured, or if fabric fencing is in use that does not cut cluster communication. Allowed values are <b>stop</b> to attempt to immediately stop Pacemaker and stay stopped, or <b>panic</b> to attempt to immediately reboot the local node, falling back to stop on failure.</p> <p>Although the default value for this property is <b>stop</b>, the safest choice for this value is <b>panic</b>, which attempts to immediately reboot the local node. If you prefer the stop behavior, as is most likely to be the case in conjunction with fabric fencing, it is recommended that you set this explicitly.</p>

For information on setting cluster properties, see [Setting and removing cluster properties](#).

## 96.5. TESTING A FENCE DEVICE

Fencing is a fundamental part of the Red Hat Cluster infrastructure and it is therefore important to validate or test that fencing is working properly.

Use the following procedure to test a fence device.

1. Use ssh, telnet, HTTP, or whatever remote protocol is used to connect to the device to manually log in and test the fence device or see what output is given. For example, if you will be configuring fencing for an IPMI-enabled device, then try to log in remotely with **ipmitool**. Take note of the options used when logging in manually because those options might be needed when using the fencing agent.  
If you are unable to log in to the fence device, verify that the device is pingable, there is nothing such as a firewall configuration that is preventing access to the fence device, remote access is enabled on the fencing device, and the credentials are correct.
2. Run the fence agent manually, using the fence agent script. This does not require that the cluster services are running, so you can perform this step before the device is configured in the cluster. This can ensure that the fence device is responding properly before proceeding.



### NOTE

The examples in this section use the **fence\_ipmilan** fence agent script for an iLO device. The actual fence agent you will use and the command that calls that agent will depend on your server hardware. You should consult the man page for the fence agent you are using to determine which options to specify. You will usually need to know the login and password for the fence device and other information related to the fence device.

The following example shows the format you would use to run the **fence\_ipmilan** fence agent script with **-o status** parameter to check the status of the fence device interface on another node without actually fencing it. This allows you to test the device and get it working before attempting to reboot the node. When running this command, you specify the name and password of an iLO user that has power on and off permissions for the iLO device.

```
fence_ipmilan -a ipaddress -l username -p password -o status
```

The following example shows the format you would use to run the **fence\_ipmilan** fence agent script with the **-o reboot** parameter. Running this command on one node reboots the node managed by this iLO device.

```
fence_ipmilan -a ipaddress -l username -p password -o reboot
```

If the fence agent failed to properly do a status, off, on, or reboot action, you should check the hardware, the configuration of the fence device, and the syntax of your commands. In addition, you can run the fence agent script with the debug output enabled. The debug output is useful for some fencing agents to see where in the sequence of events the fencing agent script is failing when logging into the fence device.

```
fence_ipmilan -a ipaddress -l username -p password -o status -D /tmp/${hostname}-fence_agent.debug
```

When diagnosing a failure that has occurred, you should ensure that the options you specified when manually logging in to the fence device are identical to what you passed on to the fence agent with the fence agent script.

For fence agents that support an encrypted connection, you may see an error due to certificate validation failing, requiring that you trust the host or that you use the fence agent's **ssl-insecure** parameter. Similarly, if SSL/TLS is disabled on the target device, you may need to account for this when setting the SSL parameters for the fence agent.



#### NOTE

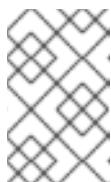
If the fence agent that is being tested is a **fence\_drac**, **fence\_ilo**, or some other fencing agent for a systems management device that continues to fail, then fall back to trying **fence\_ipmilan**. Most systems management cards support IPMI remote login and the only supported fencing agent is **fence\_ipmilan**.

- Once the fence device has been configured in the cluster with the same options that worked manually and the cluster has been started, test fencing with the **pcs stonith fence** command from any node (or even multiple times from different nodes), as in the following example. The **pcs stonith fence** command reads the cluster configuration from the CIB and calls the fence agent as configured to execute the fence action. This verifies that the cluster configuration is correct.

```
pcs stonith fence node_name
```

If the **pcs stonith fence** command works properly, that means the fencing configuration for the cluster should work when a fence event occurs. If the command fails, it means that cluster management cannot invoke the fence device through the configuration it has retrieved. Check for the following issues and update your cluster configuration as needed.

- Check your fence configuration. For example, if you have used a host map you should ensure that the system can find the node using the host name you have provided.
  - Check whether the password and user name for the device include any special characters that could be misinterpreted by the bash shell. Making sure that you enter passwords and user names surrounded by quotation marks could address this issue.
  - Check whether you can connect to the device using the exact IP address or host name you specified in the **pcs stonith** command. For example, if you give the host name in the stonith command but test by using the IP address, that is not a valid test.
  - If the protocol that your fence device uses is accessible to you, use that protocol to try to connect to the device. For example many agents use ssh or telnet. You should try to connect to the device with the credentials you provided when configuring the device, to see if you get a valid prompt and can log in to the device.  
If you determine that all your parameters are appropriate but you still have trouble connecting to your fence device, you can check the logging on the fence device itself, if the device provides that, which will show if the user has connected and what command the user issued. You can also search through the **/var/log/messages** file for instances of stonith and error, which could give some idea of what is transpiring, but some agents can provide additional information.
4. Once the fence device tests are working and the cluster is up and running, test an actual failure. To do this, take an action in the cluster that should initiate a token loss.
- Take down a network. How you take a network depends on your specific configuration. In many cases, you can physically pull the network or power cables out of the host. For information on simulating a network failure, see [What is the proper way to simulate a network failure on a RHEL Cluster?](#).



#### NOTE

Disabling the network interface on the local host rather than physically disconnecting the network or power cables is not recommended as a test of fencing because it does not accurately simulate a typical real-world failure.

- Block corosync traffic both inbound and outbound using the local firewall. The following example blocks corosync, assuming the default corosync port is used, **firewalld** is used as the local firewall, and the network interface used by corosync is in the default firewall zone:

```
firewall-cmd --direct --add-rule ipv4 filter OUTPUT 2 -p udp --dport=5405 -j DROP
firewall-cmd --add-rich-rule='rule family="ipv4" port port="5405" protocol="udp" drop'
```

- Simulate a crash and panic your machine with **sysrq-trigger**. Note, however, that triggering a kernel panic can cause data loss; it is recommended that you disable your cluster resources first.

```
echo c > /proc/sysrq-trigger
```

## 96.6. CONFIGURING FENCING LEVELS

Pacemaker supports fencing nodes with multiple devices through a feature called fencing topologies. To implement topologies, create the individual devices as you normally would and then define one or more fencing levels in the fencing topology section in the configuration.

- Each level is attempted in ascending numeric order, starting at 1.
- If a device fails, processing terminates for the current level. No further devices in that level are exercised and the next level is attempted instead.
- If all devices are successfully fenced, then that level has succeeded and no other levels are tried.
- The operation is finished when a level has passed (success), or all levels have been attempted (failed).

Use the following command to add a fencing level to a node. The devices are given as a comma-separated list of stonith ids, which are attempted for the node at that level.

```
pcs stonith level add level node devices
```

The following command lists all of the fencing levels that are currently configured.

```
pcs stonith level
```

In the following example, there are two fence devices configured for node **rh7-2**: an ilo fence device called **my\_il0** and an apc fence device called **my\_apc**. These commands set up fence levels so that if the device **my\_il0** fails and is unable to fence the node, then Pacemaker will attempt to use the device **my\_apc**. This example also shows the output of the **pcs stonith level** command after the levels are configured.

```
pcs stonith level add 1 rh7-2 my_il0
pcs stonith level add 2 rh7-2 my_apc
pcs stonith level
Node: rh7-2
 Level 1 - my_il0
 Level 2 - my_apc
```

The following command removes the fence level for the specified node and devices. If no nodes or devices are specified then the fence level you specify is removed from all nodes.

```
pcs stonith level remove level [node_id] [stonith_id] ... [stonith_id]
```

The following command clears the fence levels on the specified node or stonith id. If you do not specify a node or stonith id, all fence levels are cleared.

```
pcs stonith level clear [node|stonith_id(s)]
```

If you specify more than one stonith id, they must be separated by a comma and no spaces, as in the following example.

```
pcs stonith level clear dev_a,dev_b
```

The following command verifies that all fence devices and nodes specified in fence levels exist.

```
pcs stonith level verify
```

You can specify nodes in fencing topology by a regular expression applied on a node name and by a node attribute and its value. For example, the following commands configure nodes **node1**, **node2**, and **`node3`** to use fence devices **apc1** and **apc2**, and nodes **`node4**, **node5**, and **`node6`** to use fence devices **apc3** and **apc4**.

```
pcs stonith level add 1 "regexp%node[1-3]" apc1,apc2
pcs stonith level add 1 "regexp%node[4-6]" apc3,apc4
```

The following commands yield the same results by using node attribute matching.

```
pcs node attribute node1 rack=1
pcs node attribute node2 rack=1
pcs node attribute node3 rack=1
pcs node attribute node4 rack=2
pcs node attribute node5 rack=2
pcs node attribute node6 rack=2
pcs stonith level add 1 attrib%rack=1 apc1,apc2
pcs stonith level add 1 attrib%rack=2 apc3,apc4
```

## 96.7. CONFIGURING FENCING FOR REDUNDANT POWER SUPPLIES

When configuring fencing for redundant power supplies, the cluster must ensure that when attempting to reboot a host, both power supplies are turned off before either power supply is turned back on.

If the node never completely loses power, the node may not release its resources. This opens up the possibility of nodes accessing these resources simultaneously and corrupting them.

You need to define each device only once and to specify that both are required to fence the node, as in the following example.

```
pcs stonith create apc1 fence_apc_snmp ipaddr=apc1.example.com login=user
passwd='7a4D#1j!pz864' pcmk_host_map="node1.example.com:1;node2.example.com:2"

pcs stonith create apc2 fence_apc_snmp ipaddr=apc2.example.com login=user
passwd='7a4D#1j!pz864' pcmk_host_map="node1.example.com:1;node2.example.com:2"

pcs stonith level add 1 node1.example.com apc1,apc2
pcs stonith level add 1 node2.example.com apc1,apc2
```

## 96.8. DISPLAYING CONFIGURED FENCE DEVICES

The following command shows all currently configured fence devices. If a *stonith\_id* is specified, the command shows the options for that configured stonith device only. If the **--full** option is specified, all configured stonith options are displayed.

```
pcs stonith config [stonith_id] [--full]
```

## 96.9. MODIFYING AND DELETING FENCE DEVICES

Use the following command to modify or add options to a currently configured fencing device.

```
pcs stonith update stonith_id [stonith_device_options]
```

Use the following command to remove a fencing device from the current configuration.

```
pcs stonith delete stonith_id
```

## 96.10. MANUALLY FENCING A CLUSTER NODE

You can fence a node manually with the following command. If you specify **--off** this will use the **off** API call to stonith which will turn the node off instead of rebooting it.

```
pcs stonith fence node [--off]
```

In a situation where no stonith device is able to fence a node even if it is no longer active, the cluster may not be able to recover the resources on the node. If this occurs, after manually ensuring that the node is powered down you can enter the following command to confirm to the cluster that the node is powered down and free its resources for recovery.



### WARNING

If the node you specify is not actually off, but running the cluster software or services normally controlled by the cluster, data corruption/cluster failure will occur.

```
pcs stonith confirm node
```

## 96.11. DISABLING A FENCE DEVICE

To disable a fencing device/resource, you run the **pcs stonith disable** command.

The following command disables the fence device **myapc**.

```
pcs stonith disable myapc
```

## 96.12. PREVENTING A NODE FROM USING A FENCE DEVICE

To prevent a specific node from using a fencing device, you can configure location constraints for the fencing resource.

The following example prevents fence device **node1-ipmi** from running on **node1**.

```
pcs constraint location node1-ipmi avoids node1
```

## 96.13. CONFIGURING ACPI FOR USE WITH INTEGRATED FENCE DEVICES

If your cluster uses integrated fence devices, you must configure ACPI (Advanced Configuration and Power Interface) to ensure immediate and complete fencing.

If a cluster node is configured to be fenced by an integrated fence device, disable ACPI Soft-Off for that node. Disabling ACPI Soft-Off allows an integrated fence device to turn off a node immediately and completely rather than attempting a clean shutdown (for example, **shutdown -h now**). Otherwise, if ACPI Soft-Off is enabled, an integrated fence device can take four or more seconds to turn off a node (see the note that follows). In addition, if ACPI Soft-Off is enabled and a node panics or freezes during shutdown, an integrated fence device may not be able to turn off the node. Under those circumstances, fencing is delayed or unsuccessful. Consequently, when a node is fenced with an integrated fence device and ACPI Soft-Off is enabled, a cluster recovers slowly or requires administrative intervention to recover.



#### NOTE

The amount of time required to fence a node depends on the integrated fence device used. Some integrated fence devices perform the equivalent of pressing and holding the power button; therefore, the fence device turns off the node in four to five seconds. Other integrated fence devices perform the equivalent of pressing the power button momentarily, relying on the operating system to turn off the node; therefore, the fence device turns off the node in a time span much longer than four to five seconds.

- The preferred way to disable ACPI Soft-Off is to change the BIOS setting to "instant-off" or an equivalent setting that turns off the node without delay, as described in [Section 96.13.1, "Disabling ACPI Soft-Off with the BIOS"](#).

Disabling ACPI Soft-Off with the BIOS may not be possible with some systems. If disabling ACPI Soft-Off with the BIOS is not satisfactory for your cluster, you can disable ACPI Soft-Off with one of the following alternate methods:

- Setting **HandlePowerKey=ignore** in the **/etc/systemd/logind.conf** file and verifying that the node turns off immediately when fenced, as described in [Section 96.13.2, "Disabling ACPI Soft-Off in the logind.conf file"](#). This is the first alternate method of disabling ACPI Soft-Off.
- Appending **acpi=off** to the kernel boot command line, as described in [Section 96.13.3, "Disabling ACPI completely in the GRUB 2 File"](#). This is the second alternate method of disabling ACPI Soft-Off, if the preferred or the first alternate method is not available.



#### IMPORTANT

This method completely disables ACPI; some computers do not boot correctly if ACPI is completely disabled. Use this method *only* if the other methods are not effective for your cluster.

### 96.13.1. Disabling ACPI Soft-Off with the BIOS

You can disable ACPI Soft-Off by configuring the BIOS of each cluster node with the following procedure.



#### NOTE

The procedure for disabling ACPI Soft-Off with the BIOS may differ among server systems. You should verify this procedure with your hardware documentation.

1. Reboot the node and start the **BIOS CMOS Setup Utility** program.
2. Navigate to the Power menu (or equivalent power management menu).

- At the Power menu, set the **Soft-Off by PWR-BTTN** function (or equivalent) to **Instant-Off** (or the equivalent setting that turns off the node by means of the power button without delay). **BIOS CMOS Setup Utility:** shows a Power menu with **ACPI Function** set to **Enabled** and **Soft-Off by PWR-BTTN** set to **Instant-Off**.



#### NOTE

The equivalents to **ACPI Function**, **Soft-Off by PWR-BTTN**, and **Instant-Off** may vary among computers. However, the objective of this procedure is to configure the BIOS so that the computer is turned off by means of the power button without delay.

- Exit the **BIOS CMOS Setup Utility** program, saving the BIOS configuration.
- Verify that the node turns off immediately when fenced. For information on testing a fence device, see [Testing a fence device](#).

#### BIOS CMOS Setup Utility:

`Soft-Off by PWR-BTTN` set to  
`Instant-Off`

ACPI Function	[Enabled]	Item Help	
ACPI Suspend Type	[S1(POS)]	-----	
x Run VGABIOS if S3 Resume	Auto	Menu Level	*
Suspend Mode	[Disabled]		
HDD Power Down	[Disabled]		
Soft-Off by PWR-BTTN	[Instant-Off]		
CPU THR-M-Throttling	[50.0%]		
Wake-Up by PCI card	[Enabled]		
Power On by Ring	[Enabled]		
Wake Up On LAN	[Enabled]		
x USB KB Wake-Up From S3	Disabled		
Resume by Alarm	[Disabled]		
x Date(of Month) Alarm	0		
x Time(hh:mm:ss) Alarm	0 : 0 :		
POWER ON Function	[BUTTON ONLY]		
x KB Power ON Password	Enter		
x Hot Key Power ON	Ctrl-F1		

This example shows **ACPI Function** set to **Enabled**, and **Soft-Off by PWR-BTTN** set to **Instant-Off**.

#### 96.13.2. Disabling ACPI Soft-Off in the `logind.conf` file

To disable power-key handing in the `/etc/systemd/logind.conf` file, use the following procedure.

- Define the following configuration in the `/etc/systemd/logind.conf` file:

HandlePowerKey=ignore

2. Restart the **systemd-logind** service:

```
systemctl restart systemd-logind.service
```

3. Verify that the node turns off immediately when fenced. For information on testing a fence device, see [Testing a fence device](#).

### 96.13.3. Disabling ACPI completely in the GRUB 2 File

You can disable ACPI Soft-Off by appending **acpi=off** to the GRUB menu entry for a kernel.



#### IMPORTANT

This method completely disables ACPI; some computers do not boot correctly if ACPI is completely disabled. Use this method *only* if the other methods are not effective for your cluster.

Use the following procedure to disable ACPI in the GRUB 2 file:

1. Use the **--args** option in combination with the **--update-kernel** option of the **grubby** tool to change the **grub.cfg** file of each cluster node as follows:

```
grubby --args=acpi=off --update-kernel=ALL
```

2. Reboot the node.
3. Verify that the node turns off immediately when fenced. For information on testing a fence device, see [Testing a fence device](#).

# CHAPTER 97. CONFIGURING CLUSTER RESOURCES

The format for the command to create a cluster resource is as follows:

```
pcs resource create resource_id [standard:[provider:]] type [resource_options] [op operation_action operation_options [operation_action operation_options]...] [meta meta_options...] [clone clone_options] | master [master_options] | --group group_name [-before resource_id] | --after resource_id] | [bundle bundle_id] [--disabled] [--wait[=n]]
```

Key cluster resource creation options include the following:

- When you specify the **--group** option, the resource is added to the resource group named. If the group does not exist, this creates the group and adds this resource to the group.
- The **--before** and **--after** options specify the position of the added resource relative to a resource that already exists in a resource group.
- Specifying the **--disabled** option indicates that the resource is not started automatically.

You can determine the behavior of a resource in a cluster by configuring constraints for that resource.

## Resource creation examples

The following command creates a resource with the name **VirtualIP** of standard **ocf**, provider **heartbeat**, and type **IPaddr2**. The floating address of this resource is 192.168.0.120, and the system will check whether the resource is running every 30 seconds.

```
pcs resource create VirtualIP ocf:heartbeat:IPaddr2 ip=192.168.0.120 cidr_netmask=24 op monitor interval=30s
```

Alternately, you can omit the *standard* and *provider* fields and use the following command. This will default to a standard of **ocf** and a provider of **heartbeat**.

```
pcs resource create VirtualIP IPaddr2 ip=192.168.0.120 cidr_netmask=24 op monitor interval=30s
```

## Deleting a configured resource

Use the following command to delete a configured resource.

```
pcs resource delete resource_id
```

For example, the following command deletes an existing resource with a resource ID of **VirtualIP**.

```
pcs resource delete VirtualIP
```

## 97.1. RESOURCE AGENT IDENTIFIERS

The identifiers that you define for a resource tell the cluster which agent to use for the resource, where to find that agent and what standards it conforms to. [Table 97.1, “Resource Agent Identifiers”](#), describes these properties.

**Table 97.1. Resource Agent Identifiers**

Field	Description
standard	<p>The standard the agent conforms to. Allowed values and their meaning:</p> <ul style="list-style-type: none"> <li>* <b>ocf</b> - The specified <i>type</i> is the name of an executable file conforming to the Open Cluster Framework Resource Agent API and located beneath <b>/usr/lib/ocf/resource.d/provider</b></li> <li>* <b>lsb</b> - The specified <i>type</i> is the name of an executable file conforming to Linux Standard Base Init Script Actions. If the type does not specify a full path, the system will look for it in the <b>/etc/init.d</b> directory.</li> <li>* <b>systemd</b> - The specified <i>type</i> is the name of an installed <b>systemd</b> unit</li> <li>* <b>service</b> - Pacemaker will search for the specified <i>type</i>, first as an <b>lsb</b> agent, then as a <b>systemd</b> agent</li> <li>* <b>nagios</b> - The specified <i>type</i> is the name of an executable file conforming to the Nagios Plugin API and located in the <b>/usr/libexec/nagios/plugins</b> directory, with OCF-style metadata stored separately in the <b>/usr/share/nagios/plugins-metadata</b> directory (available in the <b>nagios-agents-metadata</b> package for certain common plugins).</li> </ul>
type	The name of the resource agent you wish to use, for example <b>IPAddr</b> or <b>Filesystem</b>
provider	The OCF spec allows multiple vendors to supply the same resource agent. Most of the agents shipped by Red Hat use <b>heartbeat</b> as the provider.

Table 97.2, “Commands to Display Resource Properties” summarizes the commands that display the available resource properties.

**Table 97.2. Commands to Display Resource Properties**

pcs Display Command	Output
<b>pcs resource list</b>	Displays a list of all available resources.
<b>pcs resource standards</b>	Displays a list of available resource agent standards.
<b>pcs resource providers</b>	Displays a list of available resource agent providers.
<b>pcs resource list <i>string</i></b>	Displays a list of available resources filtered by the specified string. You can use this command to display resources filtered by the name of a standard, a provider, or a type.

## 97.2. DISPLAYING RESOURCE-SPECIFIC PARAMETERS

For any individual resource, you can use the following command to display a description of the resource, the parameters you can set for that resource, and the default values that are set for the resource.

```
pcs resource describe [standard:[provider:]]type
```

For example, the following command displays information for a resource of type **apache**.

```
pcs resource describe ocf:heartbeat:apache
This is the resource agent for the Apache Web server.
This resource agent operates both version 1.x and version 2.x Apache
servers.
```

...

## 97.3. CONFIGURING RESOURCE META OPTIONS

In addition to the resource-specific parameters, you can configure additional resource options for any resource. These options are used by the cluster to decide how your resource should behave.

[Table 97.3, “Resource Meta Options”](#) describes the resource meta options.

**Table 97.3. Resource Meta Options**

Field	Default	Description
<b>priority</b>	<b>0</b>	If not all resources can be active, the cluster will stop lower priority resources in order to keep higher priority ones active.
<b>target-role</b>	<b>Started</b>	What state should the cluster attempt to keep this resource in? Allowed values:  * <i>Stopped</i> - Force the resource to be stopped  * <i>Started</i> - Allow the resource to be started (and in the case of promotable clones, promoted to master role if appropriate)  * <i>Master</i> - Allow the resource to be started and, if appropriate, promoted  * <i>Slave</i> - Allow the resource to be started, but only in Slave mode if the resource is promotable
<b>is-managed</b>	<b>true</b>	Is the cluster allowed to start and stop the resource? Allowed values: <b>true</b> , <b>false</b>

Field	Default	Description
<b>resource-stickiness</b>	0	Value to indicate how much the resource prefers to stay where it is.
<b>requires</b>	Calculated	<p>Indicates under what conditions the resource can be started.</p> <p>Defaults to <b>fencing</b> except under the conditions noted below. Possible values:</p> <ul style="list-style-type: none"> <li>* <b>nothing</b> - The cluster can always start the resource.</li> <li>* <b>quorum</b> - The cluster can only start this resource if a majority of the configured nodes are active. This is the default value if <b>stonith-enabled</b> is <b>false</b> or the resource's <b>standard</b> is <b>stonith</b>.</li> <li>* <b>fencing</b> - The cluster can only start this resource if a majority of the configured nodes are active <i>and</i> any failed or unknown nodes have been fenced.</li> <li>* <b>unfencing</b> - The cluster can only start this resource if a majority of the configured nodes are active <i>and</i> any failed or unknown nodes have been fenced <i>and</i> only on nodes that have been <i>unfenced</i>. This is the default value if the <b>provides=unfencing stonith</b> meta option has been set for a fencing device.</li> </ul>
<b>migration-threshold</b>	<b>INFINITY</b>	How many failures may occur for this resource on a node, before this node is marked ineligible to host this resource. A value of 0 indicates that this feature is disabled (the node will never be marked ineligible); by contrast, the cluster treats <b>INFINITY</b> (the default) as a very large but finite number. This option has an effect only if the failed operation has <b>on-fail=restart</b> (the default), and additionally for failed start operations if the cluster property <b>start-failure-is-fatal</b> is <b>false</b> .

Field	Default	Description
<b>failure-timeout</b>	<b>0</b> (disabled)	Used in conjunction with the <b>migration-threshold</b> option, indicates how many seconds to wait before acting as if the failure had not occurred, and potentially allowing the resource back to the node on which it failed. As with any time-based actions, this is not guaranteed to be checked more frequently than the value of the <b>cluster-recheck-interval</b> cluster parameter.
<b>multiple-active</b>	<b>stop_start</b>	What should the cluster do if it ever finds the resource active on more than one node. Allowed values:  * <b>block</b> - mark the resource as unmanaged  * <b>stop_only</b> - stop all active instances and leave them that way  * <b>stop_start</b> - stop all active instances and start the resource in one location only

### 97.3.1. Changing the default value of a resource option

To change the default value of a resource option, use the following command.

```
pcs resource defaults options
```

For example, the following command resets the default value of **resource-stickiness** to 100.

```
pcs resource defaults resource-stickiness=100
```

### 97.3.2. Displaying currently configured resource defaults

Omitting the *options* parameter from the **pcs resource defaults** displays a list of currently configured default values for resource options. The following example shows the output of this command after you have reset the default value of **resource-stickiness** to 100.

```
pcs resource defaults
resource-stickiness: 100
```

### 97.3.3. Setting meta options on resource creation

Whether you have reset the default value of a resource meta option or not, you can set a resource option for a particular resource to a value other than the default when you create the resource. The following shows the format of the **pcs resource create** command you use when specifying a value for a

resource meta option.

```
pcs resource create resource_id [standard:[provider:]] type [resource options] [meta meta_options...]
```

For example, the following command creates a resource with a **resource-stickiness** value of 50.

```
pcs resource create VirtualIP ocf:heartbeat:IPaddr2 ip=192.168.0.120 meta resource-stickiness=50
```

You can also set the value of a resource meta option for an existing resource, group, cloned resource, or master resource with the following command.

```
pcs resource meta resource_id | group_id | clone_id meta_options
```

In the following example, there is an existing resource named **dummy\_resource**. This command sets the **failure-timeout** meta option to 20 seconds, so that the resource can attempt to restart on the same node in 20 seconds.

```
pcs resource meta dummy_resource failure-timeout=20s
```

After executing this command, you can display the values for the resource to verify that **failure-timeout=20s** is set.

```
pcs resource config dummy_resource
```

Resource: dummy\_resource (class=ocf provider=heartbeat type=Dummy)

Meta Attrs: failure-timeout=20s

...

## 97.4. CONFIGURING RESOURCE GROUPS

One of the most common elements of a cluster is a set of resources that need to be located together, start sequentially, and stop in the reverse order. To simplify this configuration, Pacemaker supports the concept of resource groups.

### 97.4.1. Creating a resource group

You create a resource group with the following command, specifying the resources to include in the group. If the group does not exist, this command creates the group. If the group exists, this command adds additional resources to the group. The resources will start in the order you specify them with this command, and will stop in the reverse order of their starting order.

```
pcs resource group add group_name resource_id [resource_id] ... [resource_id] [--before resource_id] | [--after resource_id]
```

You can use the **--before** and **--after** options of this command to specify the position of the added resources relative to a resource that already exists in the group.

You can also add a new resource to an existing group when you create the resource, using the following command. The resource you create is added to the group named *group\_name*. If the group *group\_name* does not exist, it will be created.

```
pcs resource create resource_id [standard:[provider:]]type [resource_options] [op operation_action operation_options] --group group_name
```

There is no limit to the number of resources a group can contain. The fundamental properties of a group are as follows.

- Resources are colocated within a group.
- Resources are started in the order in which you specify them. If a resource in the group cannot run anywhere, then no resource specified after that resource is allowed to run.
- Resources are stopped in the reverse order in which you specify them.

The following example creates a resource group named **shortcut** that contains the existing resources **IPAddr** and **Email**.

```
pcs resource group add shortcut IPAddr Email
```

In this example:

- The **IPAddr** is started first, then **Email**.
- The **Email** resource is stopped first, then **IPAddr**.
- If **IPAddr** cannot run anywhere, neither can **Email**.
- If **Email** cannot run anywhere, however, this does not affect **IPAddr** in any way.

### 97.4.2. Removing a resource group

You remove a resource from a group with the following command. If there are no remaining resources in the group, this command removes the group itself.

```
pcs resource group remove group_name resource_id...
```

### 97.4.3. Displaying resource groups

The following command lists all currently configured resource groups.

```
pcs resource group list
```

### 97.4.4. Group options

You can set the following options for a resource group, and they maintain the same meaning as when they are set for a single resource: **priority**, **target-role**, **is-managed**. For information on resource meta options, see [Configuring resource meta options](#).

### 97.4.5. Group stickiness

Stickiness, the measure of how much a resource wants to stay where it is, is additive in groups. Every active resource of the group will contribute its stickiness value to the group's total. So if the default **resource-stickiness** is 100, and a group has seven members, five of which are active, then the group as a whole will prefer its current location with a score of 500.

## 97.5. DETERMINING RESOURCE BEHAVIOR

You can determine the behavior of a resource in a cluster by configuring constraints for that resource. You can configure the following categories of constraints:

- **location** constraints – A location constraint determines which nodes a resource can run on. For information on configuring location constraints, see [Determining which nodes a resource can run on](#).
- **order** constraints – An ordering constraint determines the order in which the resources run. For information on configuring ordering constraints, see [Determining the order in which cluster resources are run](#).
- **colocation** constraints – A colocation constraint determines where resources will be placed relative to other resources. For information on colocation constraints, see [Colocating cluster resources](#).

As a shorthand for configuring a set of constraints that will locate a set of resources together and ensure that the resources start sequentially and stop in reverse order, Pacemaker supports the concept of resource groups. After you have created a resource group, you can configure constraints on the group itself just as you configure constraints for individual resources. For information on resource groups, see [Configuring resource groups](#).

# CHAPTER 98. DETERMINING WHICH NODES A RESOURCE CAN RUN ON

Location constraints determine which nodes a resource can run on. You can configure location constraints to determine whether a resource will prefer or avoid a specified node.

In addition to location constraints, the node on which a resource runs is influenced by the **resource-stickiness** value for that resource, which determines to what degree a resource prefers to remain on the node where it is currently running. For information on setting the **resource-stickiness** value, see [Configuring a resource to prefer its current node](#).

## 98.1. CONFIGURING LOCATION CONSTRAINTS

You can configure a basic location constraint to specify whether a resource prefers or avoids a node, with an optional **score** value to indicate the relative degree of preference for the constraint.

The following command creates a location constraint for a resource to prefer the specified node or nodes. Note that it is possible to create constraints on a particular resource for more than one node with a single command.

```
pcs constraint location rsc prefers node[=score] [node[=score]] ...
```

The following command creates a location constraint for a resource to avoid the specified node or nodes.

```
pcs constraint location rsc avoids node[=score] [node[=score]] ...
```

[Table 98.1, “Location Constraint Options”](#) summarizes the meanings of the basic options for configuring location constraints.

Table 98.1. Location Constraint Options

Field	Description
<b>rsc</b>	A resource name
<b>node</b>	A node’s name

Field	Description
<b>score</b>	<p>Positive integer value to indicate the degree of preference for whether the given resource should prefer or avoid the given node. <b>INFINITY</b> is the default <b>score</b> value for a resource location constraint.</p> <p>A value of <b>INFINITY</b> for score in a command that configures a resource to prefer a node indicates that the resource will prefer that node if the node is available, but does not prevent the resource from running on another node if the specified node is unavailable. A value of <b>INFINITY</b> in a command that configures a resource to avoid a node indicates that the resource will never run on that node, even if no other node is available.</p> <p>A numeric score (that is, not <b>INFINITY</b>) means the constraint is optional, and will be honored unless some other factor outweighs it. For example, if the resource is already placed on a different node, and its <b>resource-stickiness</b> score is higher than a <b>prefers</b> location constraint's score, then the resource will be left where it is.</p>

The following command creates a location constraint to specify that the resource **Webserver** prefers node **node1**.

```
pcs constraint location Webserver prefers node1
```

**pcs** supports regular expressions in location constraints on the command line. These constraints apply to multiple resources based on the regular expression matching resource name. This allows you to configure multiple location constraints with a single command line.

The following command creates a location constraint to specify that resources **dummy0** to **dummy9** prefer **node1**.

```
pcs constraint location 'regexp%dummy[0-9]' prefers node1
```

Since Pacemaker uses POSIX extended regular expressions as documented at [http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1\\_chap09.html#tag\\_09\\_04](http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap09.html#tag_09_04), you can specify the same constraint with the following command.

```
pcs constraint location 'regexp%dummy[:digit:]' prefers node1
```

## 98.2. LIMITING RESOURCE DISCOVERY TO A SUBSET OF NODES

Before Pacemaker starts a resource anywhere, it first runs a one-time monitor operation (often referred to as a "probe") on every node, to learn whether the resource is already running. This process of resource discovery can result in errors on nodes that are unable to execute the monitor.

When configuring a location constraint on a node, you can use the **resource-discovery** option of the **pcs constraint location** command to indicate a preference for whether Pacemaker should perform resource discovery on this node for the specified resource. Limiting resource discovery to a subset of nodes the resource is physically capable of running on can significantly boost performance when a large set of nodes is present. When **pacemaker\_remote** is in use to expand the node count into the hundreds of nodes range, this option should be considered.

The following command shows the format for specifying the **resource-discovery** option of the **pcs**

**constraint location** command. In this command, a positive value for `score` corresponds to a basic location constraint that configures a resource to prefer a node, while a negative value for `score` corresponds to a basic location constraint that configures a resource to avoid a node. As with basic location constraints, you can use regular expressions for resources with these constraints as well.

`pcs constraint location add id rsc node score [resource-discovery=option]`

[Table 98.2, “Resource Discovery Constraint Parameters”](#) summarizes the meanings of the basic parameters for configuring constraints for resource discovery.

**Table 98.2. Resource Discovery Constraint Parameters**

Field	Description
<b>id</b>	A user-chosen name for the constraint itself.
<b>rsc</b>	A resource name
<b>node</b>	A node’s name
<b>score</b>	<p>Integer value to indicate the degree of preference for whether the given resource should prefer or avoid the given node. A positive value for <code>score</code> corresponds to a basic location constraint that configures a resource to prefer a node, while a negative value for <code>score</code> corresponds to a basic location constraint that configures a resource to avoid a node.</p> <p>A value of <b>INFINITY</b> for <code>score</code> indicates that the resource will prefer that node if the node is available, but does not prevent the resource from running on another node if the specified node is unavailable. A value of <b>-INFINITY</b> in a command that configures a resource to avoid a node indicates that the resource will never run on that node, even if no other node is available.</p> <p>A numeric score (that is, not <b>INFINITY</b> or <b>-INFINITY</b>) means the constraint is optional, and will be honored unless some other factor outweighs it. For example, if the resource is already placed on a different node, and its <b>resource-stickiness</b> score is higher than a <b>prefers</b> location constraint’s score, then the resource will be left where it is.</p>

<b>resource-discovery</b> options	<ul style="list-style-type: none"> <li>* <b>always</b> - Always perform resource discovery for the specified resource on this node. This is the default <b>resource-discovery</b> value for a resource location constraint.</li> <li>* <b>never</b> - Never perform resource discovery for the specified resource on this node.</li> <li>* <b>exclusive</b> - Perform resource discovery for the specified resource only on this node (and other nodes similarly marked as <b>exclusive</b>). Multiple location constraints using <b>exclusive</b> discovery for the same resource across different nodes creates a subset of nodes <b>resource-discovery</b> is exclusive to. If a resource is marked for <b>exclusive</b> discovery on one or more nodes, that resource is only allowed to be placed within that subset of nodes.</li> </ul>
-----------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------



### WARNING

Setting **resource-discovery** to **never** or **exclusive** removes Pacemaker's ability to detect and stop unwanted instances of a service running where it is not supposed to be. It is up to the system administrator to make sure that the service can never be active on nodes without resource discovery (such as by leaving the relevant software uninstalled).

## 98.3. CONFIGURING A LOCATION CONSTRAINT STRATEGY

When using location constraints, you can configure a general strategy for specifying which nodes a resource can run on:

- Opt-In Clusters – Configure a cluster in which, by default, no resource can run anywhere and then selectively enable allowed nodes for specific resources.
- Opt-Out Clusters – Configure a cluster in which, by default, all resources can run anywhere and then create location constraints for resources that are not allowed to run on specific nodes.

Whether you should choose to configure your cluster as an opt-in or opt-out cluster depends on both your personal preference and the make-up of your cluster. If most of your resources can run on most of the nodes, then an opt-out arrangement is likely to result in a simpler configuration. On the other hand, if most resources can only run on a small subset of nodes an opt-in configuration might be simpler.

### 98.3.1. Configuring an "Opt-In" Cluster

To create an opt-in cluster, set the **symmetric-cluster** cluster property to **false** to prevent resources from running anywhere by default.

```
pcs property set symmetric-cluster=false
```

Enable nodes for individual resources. The following commands configure location constraints so that

the resource **Webserver** prefers node **example-1**, the resource **Database** prefers node **example-2**, and both resources can fail over to node **example-3** if their preferred node fails. When configuring location constraints for an opt-in cluster, setting a score of zero allows a resource to run on a node without indicating any preference to prefer or avoid the node.

```
pcs constraint location Webserver prefers example-1=200
pcs constraint location Webserver prefers example-3=0
pcs constraint location Database prefers example-2=200
pcs constraint location Database prefers example-3=0
```

### 98.3.2. Configuring an "Opt-Out" Cluster

To create an opt-out cluster, set the **symmetric-cluster** cluster property to **true** to allow resources to run everywhere by default. This is the default configuration if **symmetric-cluster** is not set explicitly.

```
pcs property set symmetric-cluster=true
```

The following commands will then yield a configuration that is equivalent to the example in [Section 98.3.1, "Configuring an "Opt-In" Cluster"](#). Both resources can fail over to node **example-3** if their preferred node fails, since every node has an implicit score of 0.

```
pcs constraint location Webserver prefers example-1=200
pcs constraint location Webserver avoids example-2=INFINITY
pcs constraint location Database avoids example-1=INFINITY
pcs constraint location Database prefers example-2=200
```

Note that it is not necessary to specify a score of INFINITY in these commands, since that is the default value for the score.

## 98.4. CONFIGURING A RESOURCE TO PREFER ITS CURRENT NODE

Resources have a **resource-stickiness** value that you can set as a meta attribute when you create the resource, as described in [Configuring resource meta options](#). The **resource-stickiness** value determines how much a resource wants to remain on the node where it is currently running. Pacemaker considers the **resource-stickiness** value in conjunction with other settings (for example, the score values of location constraints) to determine whether to move a resource to another node or to leave it in place.

By default, a resource is created with a **resource-stickiness** value of 0. Pacemaker's default behavior when **resource-stickiness** is set to 0 and there are no location constraints is to move resources so that they are evenly distributed among the cluster nodes. This may result in healthy resources moving more often than you desire. To prevent this behavior, you can set the default **resource-stickiness** value to 1. This default will apply to all resources in the cluster. This small value can be easily overridden by other constraints that you create, but it is enough to prevent Pacemaker from needlessly moving healthy resources around the cluster.

The following command sets the default **resource-stickiness** value to 1.

```
pcs resource defaults resource-stickiness=1
```

If the **resource-stickiness** value is set, then no resources will move to a newly-added node. If resource balancing is desired at that point, you can temporarily set the **resource-stickiness** value back to 0.

Note that if a location constraint score is higher than the **resource-stickiness** value, the cluster may still move a healthy resource to the node where the location constraint points.

For further information about how Pacemaker determines where to place a resource, see [Configuring a node placement strategy](#).

# CHAPTER 99. DETERMINING THE ORDER IN WHICH CLUSTER RESOURCES ARE RUN

To determine the order in which the resources run, you configure an ordering constraint.

The following shows the format for the command to configure an ordering constraint.

```
pcs constraint order [action] resource_id then [action] resource_id [options]
```

[Table 99.1, “Properties of an Order Constraint”](#), summarizes the properties and options for configuring ordering constraints.

**Table 99.1. Properties of an Order Constraint**

Field	Description
resource_id	The name of a resource on which an action is performed.
action	<p>The action to perform on a resource. Possible values of the <i>action</i> property are as follows:</p> <ul style="list-style-type: none"> <li>* <b>start</b> - Start the resource.</li> <li>* <b>stop</b> - Stop the resource.</li> <li>* <b>promote</b> - Promote the resource from a slave resource to a master resource.</li> <li>* <b>demote</b> - Demote the resource from a master resource to a slave resource.</li> </ul> <p>If no action is specified, the default action is <b>start</b>.</p>
<b>kind</b> option	<p>How to enforce the constraint. The possible values of the <b>kind</b> option are as follows:</p> <ul style="list-style-type: none"> <li>* <b>Optional</b> - Only applies if both resources are executing the specified action. For information on optional ordering, see <a href="#">Configuring advisory ordering</a>.</li> <li>* <b>Mandatory</b> - Always (default value). If the first resource you specified is stopping or cannot be started, the second resource you specified must be stopped. For information on mandatory ordering, see <a href="#">Configuring mandatory ordering</a>.</li> <li>* <b>Serialize</b> - Ensure that no two stop/start actions occur concurrently for the resources you specify. The first and second resource you specify can start in either order, but one must complete starting before the other can be started. A typical use case is when resource startup puts a high load on the host.</li> </ul>

Field	Description
<b>symmetrical</b> option	If true, the reverse of the constraint applies for the opposite action (for example, if B starts after A starts, then B stops before Ordering constraints for which <b>kind</b> is <b>Serialize</b> cannot be symmetrical. The default value is <b>true</b> for <b>Mandatory</b> and <b>Ordering</b> kinds, <b>false</b> for <b>Serialize</b> .

Use the following command to remove resources from any ordering constraint.

```
pcs constraint order remove resource1 [resourceN]...
```

## 99.1. CONFIGURING MANDATORY ORDERING

A mandatory ordering constraint indicates that the second action should not be initiated for the second resource unless and until the first action successfully completes for the first resource. Actions that may be ordered are **stop**, **start**, and additionally for promotable clones, **demote** and **promote**. For example, "A then B" (which is equivalent to "start A then start B") means that B will not be started unless and until A successfully starts. An ordering constraint is mandatory if the **kind** option for the constraint is set to **Mandatory** or left as default.

If the **symmetrical** option is set to **true** or left to default, the opposite actions will be ordered in reverse. The **start** and **stop** actions are opposites, and **demote** and **promote** are opposites. For example, a symmetrical "promote A then start B" ordering implies "stop B then demote A", which means that A cannot be demoted until and unless B successfully stops. A symmetrical ordering means that changes in A's state can cause actions to be scheduled for B. For example, given "A then B", if A restarts due to failure, B will be stopped first, then A will be stopped, then A will be started, then B will be started.

Note that the cluster reacts to each state change. If the first resource is restarted and is in a started state again before the second resource initiated a stop operation, the second resource will not need to be restarted.

## 99.2. CONFIGURING ADVISORY ORDERING

When the **kind=Optional** option is specified for an ordering constraint, the constraint is considered optional and only applies if both resources are executing the specified actions. Any change in state by the first resource you specify will have no effect on the second resource you specify.

The following command configures an advisory ordering constraint for the resources named **VirtualIP** and **dummy\_resource**.

```
pcs constraint order VirtualIP then dummy_resource kind=Optional
```

## 99.3. CONFIGURING ORDERED RESOURCE SETS

A common situation is for an administrator to create a chain of ordered resources, where, for example, resource A starts before resource B which starts before resource C. If your configuration requires that you create a set of resources that is colocated and started in order, you can configure a resource group that contains those resources, as described in [Configuring resource groups](#).

There are some situations, however, where configuring the resources that need to start in a specified order as a resource group is not appropriate:

- You may need to configure resources to start in order and the resources are not necessarily colocated.
- You may have a resource C that must start after either resource A or B has started but there is no relationship between A and B.
- You may have resources C and D that must start after both resources A and B have started, but there is no relationship between A and B or between C and D.

In these situations, you can create an ordering constraint on a set or sets of resources with the **pcs constraint order set** command.

You can set the following options for a set of resources with the **pcs constraint order set** command.

- **sequential**, which can be set to **true** or **false** to indicate whether the set of resources must be ordered relative to each other. The default value is **true**.  
Setting **sequential** to **false** allows a set to be ordered relative to other sets in the ordering constraint, without its members being ordered relative to each other. Therefore, this option makes sense only if multiple sets are listed in the constraint; otherwise, the constraint has no effect.
- **require-all**, which can be set to **true** or **false** to indicate whether all of the resources in the set must be active before continuing. Setting **require-all** to **false** means that only one resource in the set needs to be started before continuing on to the next set. Setting **require-all** to **false** has no effect unless used in conjunction with unordered sets, which are sets for which **sequential** is set to **false**. The default value is **true**.
- **action**, which can be set to **start**, **promote**, **demote** or **stop**, as described in [Properties of an Order Constraint](#).
- **role**, which can be set to **Stopped**, **Started**, **Master**, or **Slave**.

You can set the following constraint options for a set of resources following the **setoptions** parameter of the **pcs constraint order set** command.

- **id**, to provide a name for the constraint you are defining.
- **kind**, which indicates how to enforce the constraint, as described in [Properties of an Order Constraint](#).
- **symmetrical**, to set whether the reverse of the constraint applies for the opposite action, as described in [Properties of an Order Constraint](#).

```
pcs constraint order set resource1 resource2 [resourceN]... [options] [set resourceX resourceY ... [options]] [setoptions [constraint_options]]
```

If you have three resources named **D1**, **D2**, and **D3**, the following command configures them as an ordered resource set.

```
pcs constraint order set D1 D2 D3
```

If you have six resources named **A**, **B**, **C**, **D**, **E**, and **F**, this example configures an ordering constraint for the set of resources that will start as follows:

- **A** and **B** start independently of each other
- **C** starts once either **A** or **B** has started
- **D** starts once **C** has started
- **E** and **F** start independently of each other once **D** has started

Stopping the resources is not influenced by this constraint since **symmetrical=false** is set.

```
pcs constraint order set A B sequential=false require-all=false set C D set E F
sequential=false setoptions symmetrical=false
```

## 99.4. CONFIGURING STARTUP ORDER FOR RESOURCE DEPENDENCIES NOT MANAGED BY PACEMAKER

It is possible for a cluster to include resources with dependencies that are not themselves managed by the cluster. In this case, you must ensure that those dependencies are started before Pacemaker is started and stopped after Pacemaker is stopped.

You can configure your startup order to account for this situation by means of the **systemd resource-agents-deps** target. You can create a **systemd** drop-in unit for this target and Pacemaker will order itself appropriately relative to this target.

For example, if a cluster includes a resource that depends on the external service **foo** that is not managed by the cluster, perform the following procedure.

1. Create the drop-in unit **/etc/systemd/system/resource-agents-deps.target.d/foo.conf** that contains the following:

```
[Unit]
Requires=foo.service
After=foo.service
```

2. Run the **systemctl daemon-reload** command.

A cluster dependency specified in this way can be something other than a service. For example, you may have a dependency on mounting a file system at **/srv**, in which case you would perform the following procedure:

1. Ensure that **/srv** is listed in the **/etc/fstab** file. This will be converted automatically to the **systemd** file **srv.mount** at boot when the configuration of the system manager is reloaded. For more information, see the **systemd.mount(5)** and the **systemd-fstab-generator(8)** man pages.
2. To make sure that Pacemaker starts after the disk is mounted, create the drop-in unit **/etc/systemd/system/resource-agents-deps.target.d/srv.conf** that contains the following.

```
[Unit]
Requires=srv.mount
After=srv.mount
```

3. Run the **systemctl daemon-reload** command.

# CHAPTER 100. COLOCATING CLUSTER RESOURCES

To specify that the location of one resource depends on the location of another resource, you configure a colocation constraint.

There is an important side effect of creating a colocation constraint between two resources: it affects the order in which resources are assigned to a node. This is because you cannot place resource A relative to resource B unless you know where resource B is. So when you are creating colocation constraints, it is important to consider whether you should colocate resource A with resource B or resource B with resource A.

Another thing to keep in mind when creating colocation constraints is that, assuming resource A is colocated with resource B, the cluster will also take into account resource A's preferences when deciding which node to choose for resource B.

The following command creates a colocation constraint.

```
pcs constraint colocation add [master|slave] source_resource with [master|slave] target_resource [score] [options]
```

[Table 100.1, "Properties of a Colocation Constraint"](#), summarizes the properties and options for configuring colocation constraints.

**Table 100.1. Properties of a Colocation Constraint**

Field	Description
<i>source_resource</i>	The colocation source. If the constraint cannot be satisfied, the cluster may decide not to allow the resource to run at all.
<i>target_resource</i>	The colocation target. The cluster will decide where to put this resource first and then decide where to put the source resource.
<i>score</i>	Positive values indicate the resource should run on the same node. Negative values indicate the resources should not run on the same node. A value of <b>+INFINITY</b> , the default value, indicates that the <i>source_resource</i> must run on the same node as the <i>target_resource</i> . A value of <b>-INFINITY</b> indicates that the <i>source_resource</i> must not run on the same node as the <i>target_resource</i> .

## 100.1. SPECIFYING MANDATORY PLACEMENT OF RESOURCES

Mandatory placement occurs any time the constraint's score is **+INFINITY** or **-INFINITY**. In such cases, if the constraint cannot be satisfied, then the *source\_resource* is not permitted to run. For **score=INFINITY**, this includes cases where the *target\_resource* is not active.

If you need **myresource1** to always run on the same machine as **myresource2**, you would add the following constraint:

```
pcs constraint colocation add myresource1 with myresource2 score=INFINITY
```

Because **INFINITY** was used, if **myresource2** cannot run on any of the cluster nodes (for whatever reason) then **myresource1** will not be allowed to run.

Alternatively, you may want to configure the opposite, a cluster in which **myresource1** cannot run on the same machine as **myresource2**. In this case use **score=-INFINITY**

```
pcs constraint colocation add myresource1 with myresource2 score=-INFINITY
```

Again, by specifying **-INFINITY**, the constraint is binding. So if the only place left to run is where **myresource2** already is, then **myresource1** may not run anywhere.

## 100.2. SPECIFYING ADVISORY PLACEMENT OF RESOURCES

If mandatory placement is about "must" and "must not", then advisory placement is the "I would prefer if" alternative. For constraints with scores greater than **-INFINITY** and less than **INFINITY**, the cluster will try to accommodate your wishes but may ignore them if the alternative is to stop some of the cluster resources.

## 100.3. COLOCATING SETS OF RESOURCES

If your configuration requires that you create a set of resources that are colocated and started in order, you can configure a resource group that contains those resources, as described in [Configuring resource groups](#). There are some situations, however, where configuring the resources that need to be colocated as a resource group is not appropriate:

- You may need to colocate a set of resources but the resources do not necessarily need to start in order.
- You may have a resource C that must be colocated with either resource A or B, but there is no relationship between A and B.
- You may have resources C and D that must be colocated with both resources A and B, but there is no relationship between A and B or between C and D.

In these situations, you can create a colocation constraint on a set or sets of resources with the **pcs constraint colocation set** command.

You can set the following options for a set of resources with the **pcs constraint colocation set** command.

- **sequential**, which can be set to **true** or **false** to indicate whether the members of the set must be colocated with each other.  
Setting **sequential** to **false** allows the members of this set to be colocated with another set listed later in the constraint, regardless of which members of this set are active. Therefore, this option makes sense only if another set is listed after this one in the constraint; otherwise, the constraint has no effect.
- **role**, which can be set to **Stopped**, **Started**, **Master**, or **Slave**.

You can set the following constraint option for a set of resources following the **setoptions** parameter of the **pcs constraint colocation set** command.

- **id**, to provide a name for the constraint you are defining.

- **score**, to indicate the degree of preference for this constraint. For information on this option, see [Location Constraint Options](#).

When listing members of a set, each member is colocated with the one before it. For example, "set A B" means "B is colocated with A". However, when listing multiple sets, each set is colocated with the one after it. For example, "set C D sequential=false set A B" means "set C D (where C and D have no relation between each other) is colocated with set A B (where B is colocated with A)".

The following command creates a colocation constraint on a set or sets of resources.

```
pcs constraint colocation set resource1 resource2 [resourceN]... [options] [set resourceX resourceY ... [options]] [setoptions [constraint_options]]
```

## 100.4. REMOVING COLOCATION CONSTRAINTS

Use the following command to remove colocation constraints with *source\_resource*.

```
pcs constraint colocation remove source_resource target_resource
```

# CHAPTER 101. DISPLAYING RESOURCE CONSTRAINTS

There are a several commands you can use to display constraints that have been configured.

## 101.1. DISPLAYING ALL CONFIGURED CONSTRAINTS

The following command lists all current location, order, and colocation constraints. If the **--full** option is specified, show the internal constraint IDs.

```
pcs constraint [list|show] [--full]
```

As of RHEL 8.2, listing resource constraints no longer by default displays expired constraints. To include expired constraints, use the **--all** option of the **pcs constraint** command. This will list expired constraints, noting the constraints and their associated rules as **(expired)** in the display.

## 101.2. DISPLAYING LOCATION CONSTRAINTS

The following command lists all current location constraints.

- If **resources** is specified, location constraints are displayed per resource. This is the default behavior.
- If **nodes** is specified, location constraints are displayed per node.
- If specific resources or nodes are specified, then only information about those resources or nodes is displayed.

```
pcs constraint location [show [resources [resource...]] | [nodes [node...]]] [--full]
```

## 101.3. DISPLAYING ORDERING CONSTRAINTS

The following command lists all current ordering constraints.

```
pcs constraint order [show]
```

## 101.4. DISPLAYING COLOCATION CONSTRAINTS

The following command lists all current colocation constraints.

```
pcs constraint colocation [show]
```

## 101.5. DISPLAYING RESOURCE-SPECIFIC CONSTRAINTS

The following command lists the constraints that reference specific resources.

```
pcs constraint ref resource ...
```

## 101.6. DISPLAYING RESOURCE DEPENDENCIES (RED HAT ENTERPRISE LINUX 8.2 AND LATER)

The following command displays the relations between cluster resources in a tree structure.

```
pcs resource relations resource [--full]
```

If the **--full** option is used, the command displays additional information, including the constraint IDs and the resource types.

In the following example, there are 3 configured resources: C, D, and E.

```
pcs constraint order start C then start D
Adding C D (kind: Mandatory) (Options: first-action=start then-action=start)
pcs constraint order start D then start E
Adding D E (kind: Mandatory) (Options: first-action=start then-action=start)

pcs resource relations C
C
`- order
 | start C then start D
 `- D
 `- order
 | start D then start E
 `- E
pcs resource relations D
D
|- order
| | start C then start D
| `- C
`- order
 | start D then start E
 `- E
pcs resource relations E
E
`- order
 | start D then start E
 `- D
 `- order
 | start C then start D
 `- C
```

In the following example, there are 2 configured resources: A and B. Resources A and B are part of resource group G.

```
pcs resource relations A
A
`- outer resource
 `- G
 `- inner resource(s)
 | members: A B
 `- B
pcs resource relations B
B
`- outer resource
 `- G
 `- inner resource(s)
 | members: A B
```

```
 `- A
pcs resource relations G
G
`- inner resource(s)
| members: A B
|- A
`- B
```

# CHAPTER 102. DETERMINING RESOURCE LOCATION WITH RULES

For more complicated location constraints, you can use Pacemaker rules to determine a resource's location.

## 102.1. PACEMAKER RULES

Rules can be used to make your configuration more dynamic. One use of rules might be to assign machines to different processing groups (using a node attribute) based on time and to then use that attribute when creating location constraints.

Each rule can contain a number of expressions, date-expressions and even other rules. The results of the expressions are combined based on the rule's **boolean-op** field to determine if the rule ultimately evaluates to **true** or **false**. What happens next depends on the context in which the rule is being used.

**Table 102.1. Properties of a Rule**

Field	Description
<b>role</b>	Limits the rule to apply only when the resource is in that role. Allowed values: <b>Started</b> , <b>Slave</b> , and <b>Master</b> . NOTE: A rule with <b>role="Master"</b> cannot determine the initial location of a clone instance. It will only affect which of the active instances will be promoted.
<b>score</b>	The score to apply if the rule evaluates to <b>true</b> . Limited to use in rules that are part of location constraints.
<b>score-attribute</b>	The node attribute to look up and use as a score if the rule evaluates to <b>true</b> . Limited to use in rules that are part of location constraints.
<b>boolean-op</b>	How to combine the result of multiple expression objects. Allowed values: <b>and</b> and <b>or</b> . The default value is <b>and</b> .

### 102.1.1. Node attribute expressions

Node attribute expressions are used to control a resource based on the attributes defined by a node or nodes.

**Table 102.2. Properties of an Expression**

Field	Description
<b>attribute</b>	The node attribute to test

Field	Description
<b>type</b>	Determines how the value(s) should be tested. Allowed values: <b>string</b> , <b>integer</b> , <b>version</b> . The default value is <b>string</b> .
<b>operation</b>	<p>The comparison to perform. Allowed values:</p> <ul style="list-style-type: none"> <li>* <b>lt</b> - True if the node attribute's value is less than <b>value</b></li> <li>* <b>gt</b> - True if the node attribute's value is greater than <b>value</b></li> <li>* <b>lte</b> - True if the node attribute's value is less than or equal to <b>value</b></li> <li>* <b>gte</b> - True if the node attribute's value is greater than or equal to <b>value</b></li> <li>* <b>eq</b> - True if the node attribute's value is equal to <b>value</b></li> <li>* <b>ne</b> - True if the node attribute's value is not equal to <b>value</b></li> <li>* <b>defined</b> - True if the node has the named attribute</li> <li>* <b>not_defined</b> - True if the node does not have the named attribute</li> </ul>
<b>value</b>	User supplied value for comparison (required unless <b>operation</b> is <b>defined</b> or <b>not_defined</b> )

In addition to any attributes added by the administrator, the cluster defines special, built-in node attributes for each node that can also be used, as described in [Table 102.3, “Built-in Node Attributes”](#).

**Table 102.3. Built-in Node Attributes**

Name	Description
<b>#uname</b>	Node name
<b>#id</b>	Node ID

Name	Description
<b>#kind</b>	Node type. Possible values are <b>cluster</b> , <b>remote</b> , and <b>container</b> . The value of <b>kind</b> is <b>remote</b> for Pacemaker Remote nodes created with the <b>ocf:pacemaker:remote</b> resource, and <b>container</b> for Pacemaker Remote guest nodes and bundle nodes.
<b>#is_dc</b>	<b>true</b> if this node is a Designated Controller (DC), <b>false</b> otherwise
<b>#cluster_name</b>	The value of the <b>cluster-name</b> cluster property, if set
<b>#site_name</b>	The value of the <b>site-name</b> node attribute, if set, otherwise identical to <b>#cluster-name</b>
<b>#role</b>	The role the relevant promotable clone has on this node. Valid only within a rule for a location constraint for a promotable clone.

### 102.1.2. Time/date based expressions

Date expressions are used to control a resource or cluster option based on the current date/time. They can contain an optional date specification.

Table 102.4. Properties of a Date Expression

Field	Description
<b>start</b>	A date/time conforming to the ISO8601 specification.
<b>end</b>	A date/time conforming to the ISO8601 specification.
<b>operation</b>	<p>Compares the current date/time with the start or the end date or both the start and end date, depending on the context. Allowed values:</p> <ul style="list-style-type: none"> <li>* <b>gt</b> - True if the current date/time is after <b>start</b></li> <li>* <b>lt</b> - True if the current date/time is before <b>end</b></li> <li>* <b>in_range</b> - True if the current date/time is after <b>start</b> and before <b>end</b></li> <li>* <b>date-spec</b> - performs a cron-like comparison to the current date/time</li> </ul>

### 102.1.3. Date specifications

Date specifications are used to create cron-like expressions relating to time. Each field can contain a single number or a single range. Instead of defaulting to zero, any field not supplied is ignored.

For example, **monthdays="1"** matches the first day of every month and **hours="09-17"** matches the hours between 9 am and 5 pm (inclusive). However, you cannot specify **weekdays="1,2"** or **weekdays="1-2,5-6"** since they contain multiple ranges.

Table 102.5. Properties of a Date Specification

Field	Description
<b>id</b>	A unique name for the date
<b>hours</b>	Allowed values: 0-23
<b>monthdays</b>	Allowed values: 0-31 (depending on month and year)
<b>weekdays</b>	Allowed values: 1-7 (1=Monday, 7=Sunday)
<b>yeardays</b>	Allowed values: 1-366 (depending on the year)
<b>months</b>	Allowed values: 1-12
<b>weeks</b>	Allowed values: 1-53 (depending on <b>weekyear</b> )
<b>years</b>	Year according the Gregorian calendar
<b>weekyears</b>	May differ from Gregorian years; for example, <b>2005-001 Ordinal</b> is also <b>2005-01-01 Gregorian</b> is also <b>2004-W53-6 Weekly</b>
<b>moon</b>	Allowed values: 0-7 (0 is new, 4 is full moon).

## 102.2. CONFIGURING A PACEMAKER LOCATION CONSTRAINT USING RULES

Use the following command to configure a Pacemaker constraint that uses rules. If **score** is omitted, it defaults to INFINITY. If **resource-discovery** is omitted, it defaults to **always**.

For information on the **resource-discovery** option, see [Limiting resource discovery to a subset of nodes](#).

As with basic location constraints, you can use regular expressions for resources with these constraints as well.

When using rules to configure location constraints, the value of **score** can be positive or negative, with a positive value indicating "prefers" and a negative value indicating "avoids".

```
pcs constraint location rsc rule [resource-discovery=option] [role=master|slave] [score=score | score-attribute=attribute] expression
```

The *expression* option can be one of the following where *duration\_options* and *date\_spec\_options* are: hours, monthdays, weekdays, yeardays, months, weeks, years, weekyears, moon as described in [Properties of a Date Specification](#).

- **defined|not\_defined *attribute***
- ***attribute* lt|gt|lte|gte|eq|ne [string|integer|version] *value***
- **date gt|lt *date***
- **date in\_range *date to date***
- **date in\_range *date to duration duration\_options* ...**
- **date-spec *date\_spec\_options***
- ***expression and|or expression***
- ***(expression)***

Note that durations are an alternative way to specify an end for **in\_range** operations by means of calculations. For example, you can specify a duration of 19 months.

The following location constraint configures an expression that is true if now is any time in the year 2018.

```
pcs constraint location Webserver rule score=INFINITY date-spec years=2018
```

The following command configures an expression that is true from 9 am to 5 pm, Monday through Friday. Note that the hours value of 16 matches up to 16:59:59, as the numeric value (hour) still matches.

```
pcs constraint location Webserver rule score=INFINITY date-spec hours="9-16"
weekdays="1-5"
```

The following command configures an expression that is true when there is a full moon on Friday the thirteenth.

```
pcs constraint location Webserver rule date-spec weekdays=5 monthdays=13 moon=4
```

To remove a rule, use the following command. If the rule that you are removing is the last rule in its constraint, the constraint will be removed.

```
pcs constraint rule remove rule_id
```

# CHAPTER 103. MANAGING CLUSTER RESOURCES

This section describes various commands you can use to manage cluster resources.

## 103.1. DISPLAYING CONFIGURED RESOURCES

To display a list of all configured resources, use the following command.

```
pcs resource status
```

For example, if your system is configured with a resource named **VirtualIP** and a resource named **WebSite**, the **pcs resource show** command yields the following output.

```
pcs resource status
VirtualIP (ocf::heartbeat:IPAddr2): Started
WebSite (ocf::heartbeat:apache): Started
```

To display a list of all configured resources and the parameters configured for those resources, use the **-full** option of the **pcs resource config** command, as in the following example.

```
pcs resource config
Resource: VirtualIP (type=IPAddr2 class=ocf provider=heartbeat)
 Attributes: ip=192.168.0.120 cidr_netmask=24
 Operations: monitor interval=30s
Resource: WebSite (type=apache class=ocf provider=heartbeat)
 Attributes: statusurl=http://localhost/server-status configfile=/etc/httpd/conf/httpd.conf
 Operations: monitor interval=1min
```

To display the configured parameters for a resource, use the following command.

```
pcs resource config resource_id
```

For example, the following command displays the currently configured parameters for resource **VirtualIP**.

```
pcs resource config VirtualIP
Resource: VirtualIP (type=IPAddr2 class=ocf provider=heartbeat)
 Attributes: ip=192.168.0.120 cidr_netmask=24
 Operations: monitor interval=30s
```

## 103.2. MODIFYING RESOURCE PARAMETERS

To modify the parameters of a configured resource, use the following command.

```
pcs resource update resource_id [resource_options]
```

The following sequence of commands show the initial values of the configured parameters for resource **VirtualIP**, the command to change the value of the **ip** parameter, and the values following the update command.

```
pcs resource config VirtualIP
Resource: VirtualIP (type=IPAddr2 class=ocf provider=heartbeat)
```

```
Attributes: ip=192.168.0.120 cidr_netmask=24
Operations: monitor interval=30s
pcs resource update VirtualIP ip=192.169.0.120
pcs resource config VirtualIP
Resource: VirtualIP (type=IPaddr2 class=ocf provider=heartbeat)
Attributes: ip=192.169.0.120 cidr_netmask=24
Operations: monitor interval=30s
```



#### NOTE

When you update a resource's operation with the **pcs resource update** command, any options you do not specifically call out are reset to their default values.

### 103.3. CLEARING FAILURE STATUS OF CLUSTER RESOURCES

If a resource has failed, a failure message appears when you display the cluster status. If you resolve that resource, you can clear that failure status with the **pcs resource cleanup** command. This command resets the resource status and **failcount**, telling the cluster to forget the operation history of a resource and re-detect its current state.

The following command cleans up the resource specified by *resource\_id*.

```
pcs resource cleanup resource_id
```

If you do not specify a *resource\_id*, this command resets the resource status and **failcount** for all resources.

The **pcs resource cleanup** command probes only the resources that display as a failed action. To probe all resources on all nodes you can enter the following command:

```
pcs resource refresh
```

By default, the **pcs resource refresh** command probes only the nodes where a resource's state is known. To probe all resources even if the state is not known, enter the following command:

```
pcs resource refresh --full
```

### 103.4. MOVING RESOURCES IN A CLUSTER

Pacemaker provides a variety of mechanisms for configuring a resource to move from one node to another and to manually move a resource when needed.

You can manually move resources in a cluster with the **pcs resource move** and **pcs resource relocate** commands, as described in [Manually moving cluster resources](#).

In addition to these commands, you can also control the behavior of cluster resources by enabling, disabling, and banning resources, as described in [Enabling, disabling, and banning cluster resources](#).

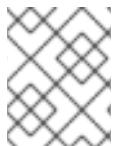
You can configure a resource so that it will move to a new node after a defined number of failures, and you can configure a cluster to move resources when external connectivity is lost.

#### 103.4.1. Moving resources due to failure

When you create a resource, you can configure the resource so that it will move to a new node after a defined number of failures by setting the **migration-threshold** option for that resource. Once the threshold has been reached, this node will no longer be allowed to run the failed resource until:

- The administrator manually resets the resource's **failcount** using the **pcs resource cleanup** command.
- The resource's **failure-timeout** value is reached.

The value of **migration-threshold** is set to **INFINITY** by default. **INFINITY** is defined internally as a very large but finite number. A value of 0 disables the **migration-threshold** feature.



#### NOTE

Setting a **migration-threshold** for a resource is not the same as configuring a resource for migration, in which the resource moves to another location without loss of state.

The following example adds a migration threshold of 10 to the resource named **dummy\_resource**, which indicates that the resource will move to a new node after 10 failures.

```
pcs resource meta dummy_resource migration-threshold=10
```

You can add a migration threshold to the defaults for the whole cluster with the following command.

```
pcs resource defaults migration-threshold=10
```

To determine the resource's current failure status and limits, use the **pcs resource failcount show** command.

There are two exceptions to the migration threshold concept; they occur when a resource either fails to start or fails to stop. If the cluster property **start-failure-is-fatal** is set to **true** (which is the default), start failures cause the **failcount** to be set to **INFINITY** and thus always cause the resource to move immediately.

Stop failures are slightly different and crucial. If a resource fails to stop and STONITH is enabled, then the cluster will fence the node in order to be able to start the resource elsewhere. If STONITH is not enabled, then the cluster has no way to continue and will not try to start the resource elsewhere, but will try to stop it again after the failure timeout.

#### 103.4.2. Moving resources due to connectivity changes

Setting up the cluster to move resources when external connectivity is lost is a two step process.

1. Add a **ping** resource to the cluster. The **ping** resource uses the system utility of the same name to test if a list of machines (specified by DNS host name or IPv4/IPv6 address) are reachable and uses the results to maintain a node attribute called **pingd**.
2. Configure a location constraint for the resource that will move the resource to a different node when connectivity is lost.

Table 97.1, “Resource Agent Identifiers” describes the properties you can set for a **ping** resource.

Table 103.1. Properties of a ping resources

Field	Description
<b>dampen</b>	The time to wait (dampening) for further changes to occur. This prevents a resource from bouncing around the cluster when cluster nodes notice the loss of connectivity at slightly different times.
<b>multiplier</b>	The number of connected ping nodes gets multiplied by this value to get a score. Useful when there are multiple ping nodes configured.
<b>host_list</b>	The machines to contact in order to determine the current connectivity status. Allowed values include resolvable DNS host names, IPv4 and IPv6 addresses. The entries in the host list are space separated.

The following example command creates a **ping** resource that verifies connectivity to **gateway.example.com**. In practice, you would verify connectivity to your network gateway/router. You configure the **ping** resource as a clone so that the resource will run on all cluster nodes.

```
pcs resource create ping ocf:pacemaker:ping dampen=5s multiplier=1000
host_list=gateway.example.com clone
```

The following example configures a location constraint rule for the existing resource named **Webserver**. This will cause the **Webserver** resource to move to a host that is able to ping **gateway.example.com** if the host that it is currently running on cannot ping **gateway.example.com**.

```
pcs constraint location Webserver rule score=-INFINITY pingd lt 1 or not_defined pingd
```

Module included in the following assemblies:

```
//
// <List assemblies here, each on a new line>
// rhel-8-docs/enterprise/assemblies/assembly_managing-cluster-resources.adoc
```

## 103.5. DISABLING A MONITOR OPERATION

The easiest way to stop a recurring monitor is to delete it. However, there can be times when you only want to disable it temporarily. In such cases, add **enabled="false"** to the operation's definition. When you want to reinstate the monitoring operation, set **enabled="true"** to the operation's definition.

When you update a resource's operation with the **pcs resource update** command, any options you do not specifically call out are reset to their default values. For example, if you have configured a monitoring operation with a custom timeout value of 600, running the following commands will reset the timeout value to the default value of 20 (or whatever you have set the default value to with the **pcs resource ops default** command).

```
pcs resource update resourceXZY op monitor enabled=false
pcs resource update resourceXZY op monitor enabled=true
```

In order to maintain the original value of 600 for this option, when you reinstate the monitoring operation you must specify that value, as in the following example.

```
pcs resource update resourceXZY op monitor timeout=600 enabled=true
```

# CHAPTER 104. CREATING CLUSTER RESOURCES THAT ARE ACTIVE ON MULTIPLE NODES (CLONED RESOURCES)

You can clone a cluster resource so that the resource can be active on multiple nodes. For example, you can use cloned resources to configure multiple instances of an IP resource to distribute throughout a cluster for node balancing. You can clone any resource provided the resource agent supports it. A clone consists of one resource or one resource group.



## NOTE

Only resources that can be active on multiple nodes at the same time are suitable for cloning. For example, a **Filesystem** resource mounting a non-clustered file system such as **ext4** from a shared memory device should not be cloned. Since the **ext4** partition is not cluster aware, this file system is not suitable for read/write operations occurring from multiple nodes at the same time.

## 104.1. CREATING AND REMOVING A CLONED RESOURCE

You can create a resource and a clone of that resource at the same time with the following command.

```
pcs resource create resource_id [standard:[provider:]] type [resource options] [meta resource meta options] clone [clone options]
```

The name of the clone will be ***resource\_id*-clone**.

You cannot create a resource group and a clone of that resource group in a single command.

Alternately, you can create a clone of a previously-created resource or resource group with the following command.

```
pcs resource clone resource_id | group_name [clone options]...
```

The name of the clone will be ***resource\_id*-clone** or ***group\_name*-clone**.



## NOTE

You need to configure resource configuration changes on one node only.



## NOTE

When configuring constraints, always use the name of the group or clone.

When you create a clone of a resource, the clone takes on the name of the resource with **-clone** appended to the name. The following commands creates a resource of type **apache** named **webfarm** and a clone of that resource named **webfarm-clone**.

```
pcs resource create webfarm apache clone
```



## NOTE

When you create a resource or resource group clone that will be ordered after another clone, you should almost always set the **interleave=true** option. This ensures that copies of the dependent clone can stop or start when the clone it depends on has stopped or started on the same node. If you do not set this option, if a cloned resource B depends on a cloned resource A and a node leaves the cluster, when the node returns to the cluster and resource A starts on that node, then all of the copies of resource B on all of the nodes will restart. This is because when a dependent cloned resource does not have the **interleave** option set, all instances of that resource depend on any running instance of the resource it depends on.

Use the following command to remove a clone of a resource or a resource group. This does not remove the resource or resource group itself.

```
pcs resource unclone resource_id | group_name
```

[Table 104.1, “Resource Clone Options”](#) describes the options you can specify for a cloned resource.

**Table 104.1. Resource Clone Options**

Field	Description
<b>priority</b> , <b>target-role</b> , <b>is-managed</b>	Options inherited from resource that is being cloned, as described in <a href="#">Table 97.3, “Resource Meta Options”</a> .
<b>clone-max</b>	How many copies of the resource to start. Defaults to the number of nodes in the cluster.
<b>clone-node-max</b>	How many copies of the resource can be started on a single node; the default value is <b>1</b> .
<b>notify</b>	When stopping or starting a copy of the clone, tell all the other copies beforehand and when the action was successful. Allowed values: <b>false</b> , <b>true</b> . The default value is <b>false</b> .
<b>globally-unique</b>	Does each copy of the clone perform a different function? Allowed values: <b>false</b> , <b>true</b>  If the value of this option is <b>false</b> , these resources behave identically everywhere they are running and thus there can be only one copy of the clone active per machine.  If the value of this option is <b>true</b> , a copy of the clone running on one machine is not equivalent to another instance, whether that instance is running on another node or on the same node. The default value is <b>true</b> if the value of <b>clone-node-max</b> is greater than one; otherwise the default value is <b>false</b> .

Field	Description
<b>ordered</b>	Should the copies be started in series (instead of in parallel). Allowed values: <b>false</b> , <b>true</b> . The default value is <b>false</b> .
<b>interleave</b>	Changes the behavior of ordering constraints (between clones) so that copies of the first clone can start or stop as soon as the copy on the same node of the second clone has started or stopped (rather than waiting until every instance of the second clone has started or stopped). Allowed values: <b>false</b> , <b>true</b> . The default value is <b>false</b> .
<b>clone-min</b>	If a value is specified, any clones which are ordered after this clone will not be able to start until the specified number of instances of the original clone are running, even if the <b>interleave</b> option is set to <b>true</b> .

To achieve a stable allocation pattern, clones are slightly sticky by default, which indicates that they have a slight preference for staying on the node where they are running. If no value for **resource-stickiness** is provided, the clone will use a value of 1. Being a small value, it causes minimal disturbance to the score calculations of other resources but is enough to prevent Pacemaker from needlessly moving copies around the cluster. For information on setting the **resource-stickiness** resource meta-option, see [Configuring resource meta options](#).

## 104.2. CONFIGURING CLONE RESOURCE CONSTRAINTS

In most cases, a clone will have a single copy on each active cluster node. You can, however, set **clone-max** for the resource clone to a value that is less than the total number of nodes in the cluster. If this is the case, you can indicate which nodes the cluster should preferentially assign copies to with resource location constraints. These constraints are written no differently to those for regular resources except that the clone's id must be used.

The following command creates a location constraint for the cluster to preferentially assign resource clone **webfarm-clone** to **node1**.

```
pcs constraint location webfarm-clone prefers node1
```

Ordering constraints behave slightly differently for clones. In the example below, because the **interleave** clone option is left to default as **false**, no instance of **webfarm-stats** will start until all instances of **webfarm-clone** that need to be started have done so. Only if no copies of **webfarm-clone** can be started then **webfarm-stats** will be prevented from being active. Additionally, **webfarm-clone** will wait for **webfarm-stats** to be stopped before stopping itself.

```
pcs constraint order start webfarm-clone then webfarm-stats
```

Colocation of a regular (or group) resource with a clone means that the resource can run on any machine with an active copy of the clone. The cluster will choose a copy based on where the clone is running and the resource's own location preferences.

Colocation between clones is also possible. In such cases, the set of allowed locations for the clone is limited to nodes on which the clone is (or will be) active. Allocation is then performed as normally.

The following command creates a colocation constraint to ensure that the resource **webfarm-stats** runs on the same node as an active copy of **webfarm-clone**.

```
pcs constraint colocation add webfarm-stats with webfarm-clone
```

## 104.3. CREATING PROMOTABLE CLONE RESOURCES

Promutable clone resources are clone resources with the **promutable** meta attribute set to **true**. They allow the instances to be in one of two operating modes; these are called **Master** and **Slave**. The names of the modes do not have specific meanings, except for the limitation that when an instance is started, it must come up in the **Slave** state.

### 104.3.1. Creating a promotable resource

You can create a resource as a promotable clone with the following single command.

```
pcs resource create resource_id [standard:[provider:]] type [resource options] promutable [clone options]
```

The name of the promotable clone will be ***resource\_id*-clone**.

Alternately, you can create a promotable resource from a previously-created resource or resource group with the following command. The name of the promotable clone will be ***resource\_id*-clone** or ***group\_name*-clone**.

```
pcs resource promotable resource_id [clone options]
```

[Table 104.2, “Extra Clone Options Available for Promutable Clones”](#) describes the extra clone options you can specify for a promotable resource.

**Table 104.2. Extra Clone Options Available for Promutable Clones**

Field	Description
<b>promoted-max</b>	How many copies of the resource can be promoted; default 1.
<b>promoted-node-max</b>	How many copies of the resource can be promoted on a single node; default 1.

### 104.3.2. Configuring promotable resource constraints

In most cases, a promotable resources will have a single copy on each active cluster node. If this is not the case, you can indicate which nodes the cluster should preferentially assign copies to with resource location constraints. These constraints are written no differently than those for regular resources.

You can create a colocation constraint which specifies whether the resources are operating in a master or slave role. The following command creates a resource colocation constraint.

```
pcs constraint colocation add [master|slave] source_resource with [master|slave] target_resource [score] [options]
```

For information on colocation constraints, see [Colocating cluster resources](#).

When configuring an ordering constraint that includes promotable resources, one of the actions that you can specify for the resources is **promote**, indicating that the resource be promoted from slave role to master role. Additionally, you can specify an action of **demote**, indicated that the resource be demoted from master role to slave role.

The command for configuring an order constraint is as follows.

```
pcs constraint order [action] resource_id then [action] resource_id [options]
```

[Determining the order in which cluster resources are run](#) .

# CHAPTER 105. MANAGING CLUSTER NODES

The following sections describe the commands you use to manage cluster nodes, including commands to start and stop cluster services and to add and remove cluster nodes.

## 105.1. STOPPING CLUSTER SERVICES

The following command stops cluster services on the specified node or nodes. As with the **pcs cluster start**, the **--all** option stops cluster services on all nodes and if you do not specify any nodes, cluster services are stopped on the local node only.

```
pcs cluster stop [--all | node] [...]
```

You can force a stop of cluster services on the local node with the following command, which performs a **kill -9** command.

```
pcs cluster kill
```

## 105.2. ENABLING AND DISABLING CLUSTER SERVICES

Use the following command to enables the cluster services, which configures the cluster services to run on startup on the specified node or nodes. Enabling allows nodes to automatically rejoin the cluster after they have been fenced, minimizing the time the cluster is at less than full strength. If the cluster services are not enabled, an administrator can manually investigate what went wrong before starting the cluster services manually, so that, for example, a node with hardware issues is not allowed back into the cluster when it is likely to fail again.

- If you specify the **--all** option, the command enables cluster services on all nodes.
- If you do not specify any nodes, cluster services are enabled on the local node only.

```
pcs cluster enable [--all | node] [...]
```

Use the following command to configure the cluster services not to run on startup on the specified node or nodes.

- If you specify the **--all** option, the command disables cluster services on all nodes.
- If you do not specify any nodes, cluster services are disabled on the local node only.

```
pcs cluster disable [--all | node] [...]
```

## 105.3. ADDING CLUSTER NODES



### NOTE

It is highly recommended that you add nodes to existing clusters only during a production maintenance window. This allows you to perform appropriate resource and deployment testing for the new node and its fencing configuration.

Use the following procedure to add a new node to an existing cluster. This procedure adds standard clusters nodes running **corosync**. For information on integrating non-corosync nodes into a cluster, see [Integrating non-corosync nodes into a cluster: the pacemaker\\_remote service](#) .

In this example, the existing cluster nodes are **clusternode-01.example.com**, **clusternode-02.example.com**, and **clusternode-03.example.com**. The new node is **newnode.example.com**.

On the new node to add to the cluster, perform the following tasks.

1. Install the cluster packages. If the cluster uses SBD, the Booth ticket manager, or a quorum device, you must manually install the respective packages (**sbd**, **booth-site**, **corosync-qdevice**) on the new node as well.

```
[root@newnode ~]# yum install -y pcs fence-agents-all
```

In addition to the cluster packages, you will also need to install and configure all of the services that you are running in the cluster, which you have installed on the existing cluster nodes. For example, if you are running an Apache HTTP server in a Red Hat high availability cluster, you will need to install the server on the node you are adding, as well as the **wget** tool that checks the status of the server.

2. If you are running the **firewalld** daemon, execute the following commands to enable the ports that are required by the Red Hat High Availability Add-On.

```
firewall-cmd --permanent --add-service=high-availability
firewall-cmd --add-service=high-availability
```

3. Set a password for the user ID **hacluster**. It is recommended that you use the same password for each node in the cluster.

```
[root@newnode ~]# passwd hacluster
Changing password for user hacluster.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

4. Execute the following commands to start the **pcsd** service and to enable **pcsd** at system start.

```
systemctl start pcsd.service
systemctl enable pcsd.service
```

On a node in the existing cluster, perform the following tasks.

1. Authenticate user **hacluster** on the new cluster node.

```
[root@clusternode-01 ~]# pcs host auth newnode.example.com
Username: hacluster
Password:
newnode.example.com: Authorized
```

2. Add the new node to the existing cluster. This command also syncs the cluster configuration file **corosync.conf** to all nodes in the cluster, including the new node you are adding.

```
[root@clusternode-01 ~]# pcs cluster node add newnode.example.com
```

On the new node to add to the cluster, perform the following tasks.

1. Start and enable cluster services on the new node.

```
[root@newnode ~]# pcs cluster start
Starting Cluster...
[root@newnode ~]# pcs cluster enable
```

2. Ensure that you configure and test a fencing device for the new cluster node.

## 105.4. REMOVING CLUSTER NODES

The following command shuts down the specified node and removes it from the cluster configuration file, **corosync.conf**, on all of the other nodes in the cluster.

```
pcs cluster node remove node
```

## 105.5. ADDING A NODE TO A CLUSTER WITH MULTIPLE LINKS

When adding a node to a cluster with multiple links, you must specify addresses for all links. The following example adds the node **rh80-node3** to a cluster, specifying IP address 192.168.122.203 for the first link and 192.168.123.203 as the second link.

```
pcs cluster node add rh80-node3 addr=192.168.122.203 addr=192.168.123.203
```

## 105.6. ADDING AND MODIFYING LINKS IN AN EXISTING CLUSTER (RHEL 8.1 AND LATER)

In most cases, you can add or modify the links in an existing cluster without restarting the cluster.

### 105.6.1. Adding and removing links in an existing cluster

To add a new link to a running cluster, use the **pcs cluster link add** command.

- When adding a link, you must specify an address for each node.
- Adding and removing a link is only possible when you are using the knet transport protocol.
- At least one link in the cluster must be defined at any time.
- The maximum number of links in a cluster is 8, numbered 0-7. It does not matter which links are defined, so, for example, you can define only links 3, 6 and 7.
- When you add a link without specifying its link number, **pcs** uses the lowest link available.
- The link numbers of currently configured links are contained in the **corosync.conf** file. To display the **corosync.conf** file, run the **pcs cluster corosync** command.

The following command adds link number 5 to a three node cluster.

```
[root@node1 ~] # pcs cluster link add node1=10.0.5.11 node2=10.0.5.12 node3=10.0.5.31
options linknumber=5
```

To remove an existing link, use the **pcs cluster link delete** or **pcs cluster link remove** command. Either of the following commands will remove link number 5 from the cluster.

```
[root@node1 ~] # pcs cluster link delete 5
```

```
[root@node1 ~] # pcs cluster link remove 5
```

### 105.6.2. Modifying a link in a cluster with multiple links

If there are multiple links in the cluster and you want to change one of them, perform the following procedure.

1. Remove the link you want to change.

```
[root@node1 ~] # pcs cluster link remove 2
```

2. Add the link back to the cluster with the updated addresses and options.

```
[root@node1 ~] # pcs cluster link add node1=10.0.5.11 node2=10.0.5.12
node3=10.0.5.31 options linknumber=2
```

### 105.6.3. Modifying the link addresses in a cluster with a single link

If your cluster uses only one link and you want to modify that link to use different addresses, perform the following procedure. In this example, the original link is link 1.

1. Add a new link with the new addresses and options.

```
[root@node1 ~] # pcs cluster link add node1=10.0.5.11 node2=10.0.5.12
node3=10.0.5.31 options linknumber=2
```

2. Remove the original link.

```
[root@node1 ~] # pcs cluster link remove 1
```

Note that you cannot specify addresses that are currently in use when adding links to a cluster. This means, for example, that if you have a two-node cluster with one link and you want to change the address for one node only, you cannot use the above procedure to add a new link that specifies one new address and one existing address. Instead, you can add a temporary link before removing the existing link and adding it back with the updated address, as in the following example.

In this example:

- The link for the existing cluster is link 1, which uses the address 10.0.5.11 for node 1 and the address 10.0.5.12 for node 2.
- You would like to change the address for node 2 to 10.0.5.31.

To update only one of the addresses for a two-node cluster with a single link, use the following procedure.

1. Add a new temporary link to the existing cluster, using addresses that are not currently in use.

```
[root@node1 ~] # pcs cluster link add node1=10.0.5.13 node2=10.0.5.14 options
linknumber=2
```

2. Remove the original link.

```
[root@node1 ~] # pcs cluster link remove 1
```

3. Add the new, modified link.

```
[root@node1 ~] # pcs cluster link add node1=10.0.5.11 node2=10.0.5.31 options
linknumber=1
```

4. Remove the temporary link you created

```
[root@node1 ~] # pcs cluster link remove 2
```

#### 105.6.4. Modifying the link options for a link in a cluster with a single link

If your cluster uses only one link and you want to modify the options for that link but you do not want to change the address to use, you can add a temporary link before removing and updating the link to modify.

In this example:

- The link for the existing cluster is link 1, which uses the address 10.0.5.11 for node 1 and the address 10.0.5.12 for node 2.
- You would like to change the link option **link\_priority** to 11.

Use the following procedure to modify the link option in a cluster with a single link.

1. Add a new temporary link to the existing cluster, using addresses that are not currently in use.

```
[root@node1 ~] # pcs cluster link add node1=10.0.5.13 node2=10.0.5.14 options
linknumber=2
```

2. Remove the original link.

```
[root@node1 ~] # pcs cluster link remove 1
```

3. Add back the original link with the updated options.

```
[root@node1 ~] # pcs cluster link add node1=10.0.5.11 node2=10.0.5.12 options
linknumber=1 link_priority=11
```

4. Remove the temporary link.

```
[root@node1 ~] # pcs cluster link remove 2
```

#### 105.6.5. Modifying a link when adding a new link is not possible

If for some reason adding a new link is not possible in your configuration and your only option is to modify a single existing link, you can use the following procedure, which requires that you shut your cluster down.

The following example procedure updates link number 1 in the cluster and sets the **link\_priority** option for the link to 11.

1. Stop the cluster services for the cluster.

```
[root@node1 ~] # pcs cluster stop --all
```

2. Update the link addresses and options.

The **pcs cluster link update** command does not require that you specify all of the node addresses and options. Instead, you can specify only the addresses to change. This example modifies the addresses for **node1** and **node3** and the **link\_priority** option only.

```
[root@node1 ~] # pcs cluster link update 1 node1=10.0.5.11 node3=10.0.5.31 options link_priority=11
```

To remove an option, you can set the option to a null value with the **option=** format.

3. Restart the cluster

```
[root@node1 ~] # pcs cluster start --all
```

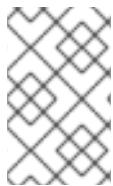
# CHAPTER 106. PACEMAKER CLUSTER PROPERTIES

Cluster properties control how the cluster behaves when confronted with situations that may occur during cluster operation.

## 106.1. SUMMARY OF CLUSTER PROPERTIES AND OPTIONS

[Table 106.1, “Cluster Properties”](#) summaries the Pacemaker cluster properties, showing the default values of the properties and the possible values you can set for those properties.

There are additional cluster properties that determine fencing behavior. For information on these properties, see [Advanced fencing configuration options](#).



### NOTE

In addition to the properties described in this table, there are additional cluster properties that are exposed by the cluster software. For these properties, it is recommended that you not change their values from their defaults.

**Table 106.1. Cluster Properties**

Option	Default	Description
<b>batch-limit</b>	0	The number of resource actions that the cluster is allowed to execute in parallel. The “correct” value will depend on the speed and load of your network and cluster nodes. The default value of 0 means that the cluster will dynamically impose a limit when any node has a high CPU load.
<b>migration-limit</b>	-1 (unlimited)	The number of migration jobs that the cluster is allowed to execute in parallel on a node.
<b>no-quorum-policy</b>	stop	What to do when the cluster does not have quorum. Allowed values:  * ignore - continue all resource management  * freeze - continue resource management, but do not recover resources from nodes not in the affected partition  * stop - stop all resources in the affected cluster partition  * suicide - fence all nodes in the affected cluster partition
<b>symmetric-cluster</b>	true	Indicates whether resources can run on any node by default.

Option	Default	Description
<b>cluster-delay</b>	60s	Round trip delay over the network (excluding action execution). The "correct" value will depend on the speed and load of your network and cluster nodes.
<b>stop-orphan-resources</b>	true	Indicates whether deleted resources should be stopped.
<b>stop-orphan-actions</b>	true	Indicates whether deleted actions should be canceled.
<b>start-failure-is-fatal</b>	true	<p>Indicates whether a failure to start a resource on a particular node prevents further start attempts on that node. When set to <b>false</b>, the cluster will decide whether to try starting on the same node again based on the resource's current failure count and migration threshold. For information on setting the <b>migration-threshold</b> option for a resource, see <a href="#">Configuring resource meta options</a>.</p> <p>Setting <b>start-failure-is-fatal</b> to <b>false</b> incurs the risk that this will allow one faulty node that is unable to start a resource to hold up all dependent actions. This is why <b>start-failure-is-fatal</b> defaults to true. The risk of setting <b>start-failure-is-fatal=false</b> can be mitigated by setting a low migration threshold so that other actions can proceed after that many failures.</p>
<b>pe-error-series-max</b>	-1 (all)	The number of scheduler inputs resulting in ERRORs to save. Used when reporting problems.
<b>pe-warn-series-max</b>	-1 (all)	The number of scheduler inputs resulting in WARNINGS to save. Used when reporting problems.
<b>pe-input-series-max</b>	-1 (all)	The number of "normal" scheduler inputs to save. Used when reporting problems.
<b>cluster-infrastructure</b>		The messaging stack on which Pacemaker is currently running. Used for informational and diagnostic purposes; not user-configurable.

Option	Default	Description
<b>dc-version</b>		Version of Pacemaker on the cluster's Designated Controller (DC). Used for diagnostic purposes; not user-configurable.
<b>cluster-recheck-interval</b>	15 minutes	Polling interval for time-based changes to options, resource parameters and constraints. Allowed values: Zero disables polling, positive values are an interval in seconds (unless other SI units are specified, such as 5min). Note that this value is the maximum time between checks; if a cluster event occurs sooner than the time specified by this value, the check will be done sooner.
<b>maintenance-mode</b>	false	Maintenance Mode tells the cluster to go to a "hands off" mode, and not start or stop any services until told otherwise. When maintenance mode is completed, the cluster does a sanity check of the current state of any services, and then stops or starts any that need it.
<b>shutdown-escalation</b>	20min	The time after which to give up trying to shut down gracefully and just exit. Advanced use only.
<b>stop-all-resources</b>	false	Should the cluster stop all resources.
<b>enable-acl</b>	false	Indicates whether the cluster can use access control lists, as set with the <b>pcs acl</b> command.
<b>placement-strategy</b>	<b>default</b>	Indicates whether and how the cluster will take utilization attributes into account when determining resource placement on cluster nodes.

## 106.2. SETTING AND REMOVING CLUSTER PROPERTIES

To set the value of a cluster property, use the following **pcs** command.

```
pcs property set property=value
```

For example, to set the value of **symmetric-cluster** to **false**, use the following command.

```
pcs property set symmetric-cluster=false
```

You can remove a cluster property from the configuration with the following command.

**pcs property unset *property***

Alternately, you can remove a cluster property from a configuration by leaving the value field of the **pcs property set** command blank. This restores that property to its default value. For example, if you have previously set the **symmetric-cluster** property to **false**, the following command removes the value you have set from the configuration and restores the value of **symmetric-cluster** to **true**, which is its default value.

**# pcs property set symmetric-cluster=**

## 106.3. QUERYING CLUSTER PROPERTY SETTINGS

In most cases, when you use the **pcs** command to display values of the various cluster components, you can use **pcs list** or **pcs show** interchangeably. In the following examples, **pcs list** is the format used to display an entire list of all settings for more than one property, while **pcs show** is the format used to display the values of a specific property.

To display the values of the property settings that have been set for the cluster, use the following **pcs** command.

**pcs property list**

To display all of the values of the property settings for the cluster, including the default values of the property settings that have not been explicitly set, use the following command.

**pcs property list --all**

To display the current value of a specific cluster property, use the following command.

**pcs property show *property***

For example, to display the current value of the **cluster-infrastructure** property, execute the following command:

**# pcs property show cluster-infrastructure**

Cluster Properties:

cluster-infrastructure: cman

For informational purposes, you can display a list of all of the default values for the properties, whether they have been set to a value other than the default or not, by using the following command.

**pcs property [list|show] --defaults**

# CHAPTER 107. CONFIGURING A VIRTUAL DOMAIN AS A RESOURCE

You can configure a virtual domain that is managed by the **libvirt** virtualization framework as a cluster resource with the **pcs resource create** command, specifying **VirtualDomain** as the resource type.

When configuring a virtual domain as a resource, take the following considerations into account:

- A virtual domain should be stopped before you configure it as a cluster resource.
- Once a virtual domain is a cluster resource, it should not be started, stopped, or migrated except through the cluster tools.
- Do not configure a virtual domain that you have configured as a cluster resource to start when its host boots.
- All nodes allowed to run a virtual domain must have access to the necessary configuration files and storage devices for that virtual domain.

If you want the cluster to manage services within the virtual domain itself, you can configure the virtual domain as a guest node.

## 107.1. VIRTUAL DOMAIN RESOURCE OPTIONS

[Table 107.1, “Resource Options for Virtual Domain Resources”](#) describes the resource options you can configure for a **VirtualDomain** resource.

**Table 107.1. Resource Options for Virtual Domain Resources**

Field	Default	Description
<b>config</b>		(required) Absolute path to the <b>libvirt</b> configuration file for this virtual domain.
<b>hypervisor</b>	System dependent	Hypervisor URI to connect to. You can determine the system’s default URI by running the <b>virsh --quiet uri</b> command.
<b>force_stop</b>	<b>0</b>	Always forcefully shut down (“destroy”) the domain on stop. The default behavior is to resort to a forceful shutdown only after a graceful shutdown attempt has failed. You should set this to <b>true</b> only if your virtual domain (or your virtualization back end) does not support graceful shutdown.

Field	Default	Description
<b>migration_transport</b>	System dependent	Transport used to connect to the remote hypervisor while migrating. If this parameter is omitted, the resource will use <b>libvirt</b> 's default transport to connect to the remote hypervisor.
<b>migration_network_suffix</b>		Use a dedicated migration network. The migration URI is composed by adding this parameter's value to the end of the node name. If the node name is a fully qualified domain name (FQDN), insert the suffix immediately prior to the first period (.) in the FQDN. Ensure that this composed host name is locally resolvable and the associated IP address is reachable through the favored network.
<b>monitor_scripts</b>		To additionally monitor services within the virtual domain, add this parameter with a list of scripts to monitor. <i>Note:</i> When monitor scripts are used, the <b>start</b> and <b>migrate_from</b> operations will complete only when all monitor scripts have completed successfully. Be sure to set the timeout of these operations to accommodate this delay
<b>autoSet_utilization_cpu</b>	<b>true</b>	If set to <b>true</b> , the agent will detect the number of <b>domainU</b> 's <b>vCPUs</b> from <b>virsh</b> , and put it into the CPU utilization of the resource when the monitor is executed.
<b>autoSet_utilization_hv_memory</b>	<b>true</b>	If set to <b>true</b> , the agent will detect the number of <b>Max memory</b> from <b>virsh</b> , and put it into the <b>hv_memory</b> utilization of the source when the monitor is executed.
<b>migrateport</b>	random highport	This port will be used in the <b>qemu</b> migrate URI. If unset, the port will be a random highport.

Field	Default	Description
<b>snapshot</b>		Path to the snapshot directory where the virtual machine image will be stored. When this parameter is set, the virtual machine's RAM state will be saved to a file in the snapshot directory when stopped. If on start a state file is present for the domain, the domain will be restored to the same state it was in right before it stopped last. This option is incompatible with the <b>force_stop</b> option.

In addition to the **VirtualDomain** resource options, you can configure the **allow-migrate** metadata option to allow live migration of the resource to another node. When this option is set to **true**, the resource can be migrated without loss of state. When this option is set to **false**, which is the default state, the virtual domain will be shut down on the first node and then restarted on the second node when it is moved from one node to the other.

## 107.2. CREATING THE VIRTUAL DOMAIN RESOURCE

Use the following procedure to create a **VirtualDomain** resource in a cluster for a virtual machine you have previously created:

1. To create the **VirtualDomain** resource agent for the management of the virtual machine, Pacemaker requires the virtual machine's **xml** configuration file to be dumped to a file on disk. For example, if you created a virtual machine named **guest1**, dump the **xml** file to a file somewhere on one of the cluster nodes that will be allowed to run the guest. You can use a file name of your choosing; this example uses **/etc/pacemaker/guest1.xml**.

```
virsh dumpxml guest1 > /etc/pacemaker/guest1.xml
```

2. Copy the virtual machine's **xml** configuration file to all of the other cluster nodes that will be allowed to run the guest, in the same location on each node.
  3. Ensure that all of the nodes allowed to run the virtual domain have access to the necessary storage devices for that virtual domain.
  4. Separately test that the virtual domain can start and stop on each node that will run the virtual domain.
  5. If it is running, shut down the guest node. Pacemaker will start the node when it is configured in the cluster. The virtual machine should not be configured to start automatically when the host boots.
  6. Configure the **VirtualDomain** resource with the **pcs resource create** command. For example, the following command configures a **VirtualDomain** resource named **VM**. Since the **allow-migrate** option is set to **true** a **pcs move VM nodeX** command would be done as a live migration.
- In this example **migration\_transport** is set to **ssh**. Note that for SSH migration to work properly, keyless logging must work between nodes.

```
pcs resource create VM VirtualDomain config=/etc/pacemaker/guest1.xml
migration_transport=ssh meta allow-migrate=true
```

# CHAPTER 108. CLUSTER QUORUM

A Red Hat Enterprise Linux High Availability Add-On cluster uses the **votequorum** service, in conjunction with fencing, to avoid split brain situations. A number of votes is assigned to each system in the cluster, and cluster operations are allowed to proceed only when a majority of votes is present. The service must be loaded into all nodes or none; if it is loaded into a subset of cluster nodes, the results will be unpredictable. For information on the configuration and operation of the **votequorum** service, see the **votequorum**(5) man page.

## 108.1. CONFIGURING QUORUM OPTIONS

There are some special features of quorum configuration that you can set when you create a cluster with the **pcs cluster setup** command. Table 108.1, “Quorum Options” summarizes these options.

Table 108.1. Quorum Options

Option	Description
<b>auto_tie_breaker</b>	<p>When enabled, the cluster can suffer up to 50% of the nodes failing at the same time, in a deterministic fashion. The cluster partition, or the set of nodes that are still in contact with the <b>nodeid</b> configured in <b>auto_tie_breaker_node</b> (or lowest <b>nodeid</b> if not set), will remain quorate. The other nodes will be inquorate.</p> <p>The <b>auto_tie_breaker</b> option is principally used for clusters with an even number of nodes, as it allows the cluster to continue operation with an even split. For more complex failures, such as multiple, uneven splits, it is recommended that you use a quorum device, as described in <a href="#">Quorum devices</a>.</p> <p>The <b>auto_tie_breaker</b> option is incompatible with quorum devices.</p>
<b>wait_for_all</b>	<p>When enabled, the cluster will be quorate for the first time only after all nodes have been visible at least once at the same time.</p> <p>The <b>wait_for_all</b> option is primarily used for two-node clusters and for even-node clusters using the quorum device <b>lms</b> (last man standing) algorithm.</p> <p>The <b>wait_for_all</b> option is automatically enabled when a cluster has two nodes, does not use a quorum device, and <b>auto_tie_breaker</b> is disabled. You can override this by explicitly setting <b>wait_for_all</b> to 0.</p>
<b>last_man_standing</b>	<p>When enabled, the cluster can dynamically recalculate <b>expected_votes</b> and quorum under specific circumstances. You must enable <b>wait_for_all</b> when you enable this option. The <b>last_man_standing</b> option is incompatible with quorum devices.</p>

Option	Description
<b>last_man_standing_window</b>	The time, in milliseconds, to wait before recalculating <b>expected_votes</b> and quorum after a cluster loses nodes.

For further information about configuring and using these options, see the **votequorum**(5) man page.

## 108.2. MODIFYING QUORUM OPTIONS

You can modify general quorum options for your cluster with the **pcs quorum update** command. Executing this command requires that the cluster be stopped. For information on the quorum options, see the **votequorum**(5) man page.

The format of the **pcs quorum update** command is as follows.

```
pcs quorum update [auto_tie_breaker=[0|1]] [last_man_standing=[0|1]] [last_man_standing_window=[time-in-ms] [wait_for_all=[0|1]]]
```

The following series of commands modifies the **wait\_for\_all** quorum option and displays the updated status of the option. Note that the system does not allow you to execute this command while the cluster is running.

```
[root@node1:~]# pcs quorum update wait_for_all=1
Checking corosync is not running on nodes...
Error: node1: corosync is running
Error: node2: corosync is running
```

```
[root@node1:~]# pcs cluster stop --all
node2: Stopping Cluster (pacemaker)...
node1: Stopping Cluster (pacemaker)...
node1: Stopping Cluster (corosync)...
node2: Stopping Cluster (corosync)...
```

```
[root@node1:~]# pcs quorum update wait_for_all=1
Checking corosync is not running on nodes...
node2: corosync is not running
node1: corosync is not running
Sending updated corosync.conf to nodes...
node1: Succeeded
node2: Succeeded
```

```
[root@node1:~]# pcs quorum config
Options:
 wait_for_all: 1
```

## 108.3. DISPLAYING QUORUM CONFIGURATION AND STATUS

Once a cluster is running, you can enter the following cluster quorum commands.

The following command shows the quorum configuration.

**pcs quorum [config]**

The following command shows the quorum runtime status.

**pcs quorum status**

## 108.4. RUNNING INQUORATE CLUSTERS

If you take nodes out of a cluster for a long period of time and the loss of those nodes would cause quorum loss, you can change the value of the **expected\_votes** parameter for the live cluster with the **pcs quorum expected-votes** command. This allows the cluster to continue operation when it does not have quorum.



### WARNING

Changing the expected votes in a live cluster should be done with extreme caution. If less than 50% of the cluster is running because you have manually changed the expected votes, then the other nodes in the cluster could be started separately and run cluster services, causing data corruption and other unexpected results. If you change this value, you should ensure that the **wait\_for\_all** parameter is enabled.

The following command sets the expected votes in the live cluster to the specified value. This affects the live cluster only and does not change the configuration file; the value of **expected\_votes** is reset to the value in the configuration file in the event of a reload.

**pcs quorum expected-votes votes**

In a situation in which you know that the cluster is inquorate but you want the cluster to proceed with resource management, you can use the **pcs quorum unlock** command to prevent the cluster from waiting for all nodes when establishing quorum.



### NOTE

This command should be used with extreme caution. Before issuing this command, it is imperative that you ensure that nodes that are not currently in the cluster are switched off and have no access to shared resources.

**# pcs quorum unlock**

## 108.5. QUORUM DEVICES

You can allow a cluster to sustain more node failures than standard quorum rules allows by configuring a separate quorum device which acts as a third-party arbitration device for the cluster. A quorum device is recommended for clusters with an even number of nodes and highly recommended for two-node clusters.

You must take the following into account when configuring a quorum device.

- It is recommended that a quorum device be run on a different physical network at the same site as the cluster that uses the quorum device. Ideally, the quorum device host should be in a separate rack than the main cluster, or at least on a separate PSU and not on the same network segment as the corosync ring or rings.
- You cannot use more than one quorum device in a cluster at the same time.
- Although you cannot use more than one quorum device in a cluster at the same time, a single quorum device may be used by several clusters at the same time. Each cluster using that quorum device can use different algorithms and quorum options, as those are stored on the cluster nodes themselves. For example, a single quorum device can be used by one cluster with an **ffsplit** (fifty/fifty split) algorithm and by a second cluster with an **lms** (last man standing) algorithm.
- A quorum device should not be run on an existing cluster node.

### 108.5.1. Installing quorum device packages

Configuring a quorum device for a cluster requires that you install the following packages:

- Install **corosync-qdevice** on the nodes of an existing cluster.

```
[root@node1:~]# yum install corosync-qdevice
[root@node2:~]# yum install corosync-qdevice
```

- Install **pcs** and **corosync-qnetd** on the quorum device host.

```
[root@qdevice:~]# yum install pcs corosync-qnetd
```

- Start the **pcsd** service and enable **pcsd** at system start on the quorum device host.

```
[root@qdevice:~]# systemctl start pcsd.service
[root@qdevice:~]# systemctl enable pcsd.service
```

### 108.5.2. Configuring a quorum device

The following procedure configures a quorum device and adds it to the cluster. In this example:

- The node used for a quorum device is **qdevice**.
- The quorum device model is **net**, which is currently the only supported model. The **net** model supports the following algorithms:
  - ffsplit**: fifty-fifty split. This provides exactly one vote to the partition with the highest number of active nodes.
  - lms**: last-man-standing. If the node is the only one left in the cluster that can see the **qnetd** server, then it returns a vote.



## WARNING

The LMS algorithm allows the cluster to remain quorate even with only one remaining node, but it also means that the voting power of the quorum device is great since it is the same as `number_of_nodes - 1`. Losing connection with the quorum device means losing `number_of_nodes - 1` votes, which means that only a cluster with all nodes active can remain quorate (by overvoting the quorum device); any other cluster becomes inquorate.

For more detailed information on the implementation of these algorithms, see the **`corosync-qdevice(8)`** man page.

- The cluster nodes are **node1** and **node2**.

The following procedure configures a quorum device and adds that quorum device to a cluster.

1. On the node that you will use to host your quorum device, configure the quorum device with the following command. This command configures and starts the quorum device model **net** and configures the device to start on boot.

```
[root@qdevice:~]# pcs qdevice setup model net --enable --start
Quorum device 'net' initialized
quorum device enabled
Starting quorum device...
quorum device started
```

After configuring the quorum device, you can check its status. This should show that the **`corosync-qnetd`** daemon is running and, at this point, there are no clients connected to it. The **--full** command option provides detailed output.

```
[root@qdevice:~]# pcs qdevice status net --full
QNNetd address: *:5403
TLS: Supported (client certificate required)
Connected clients: 0
Connected clusters: 0
Maximum send/receive size: 32768/32768 bytes
```

2. Enable the ports on the firewall needed by the **`pcsd`** daemon and the **net** quorum device by enabling the **high-availability** service on **firewalld** with following commands.

```
[root@qdevice:~]# firewall-cmd --permanent --add-service=high-availability
[root@qdevice:~]# firewall-cmd --add-service=high-availability
```

3. From one of the nodes in the existing cluster, authenticate user **hacluster** on the node that is hosting the quorum device. This allows **pcs** on the cluster to connect to **pcs** on the **qdevice** host, but does not allow **pcs** on the **qdevice** host to connect to **pcs** on the cluster.

```
[root@node1:~]# pcs host auth qdevice
Username: hacluster
```

>Password:  
qdevice: Authorized

4. Add the quorum device to the cluster.

Before adding the quorum device, you can check the current configuration and status for the quorum device for later comparison. The output for these commands indicates that the cluster is not yet using a quorum device, and the **Qdevice** membership status for each node is **NR** (Not Registered).

[root@node1:~]# **pcs quorum config**  
Options:

[root@node1:~]# **pcs quorum status**

Quorum information

-----  
Date: Wed Jun 29 13:15:36 2016  
Quorum provider: corosync\_votequorum  
Nodes: 2  
Node ID: 1  
Ring ID: 1/8272  
Quorate: Yes

Votequorum information

-----  
Expected votes: 2  
Highest expected: 2  
Total votes: 2  
Quorum: 1  
Flags: 2Node Quorate

Membership information

-----  
Nodeid Votes Qdevice Name  
1 1 NR node1 (local)  
2 1 NR node2

The following command adds the quorum device that you have previously created to the cluster. You cannot use more than one quorum device in a cluster at the same time. However, one quorum device can be used by several clusters at the same time. This example command configures the quorum device to use the **ffsplit** algorithm. For information on the configuration options for the quorum device, see the **corosync-qdevice(8)** man page.

[root@node1:~]# **pcs quorum device add model net host=qdevice algorithm=ffsplit**  
Setting up qdevice certificates on nodes...  
node2: Succeeded  
node1: Succeeded  
Enabling corosync-qdevice...  
node1: corosync-qdevice enabled  
node2: corosync-qdevice enabled  
Sending updated corosync.conf to nodes...  
node1: Succeeded  
node2: Succeeded  
Corosync configuration reloaded

```
Starting corosync-qdevice...
node1: corosync-qdevice started
node2: corosync-qdevice started
```

5. Check the configuration status of the quorum device.

From the cluster side, you can execute the following commands to see how the configuration has changed.

The **pcs quorum config** shows the quorum device that has been configured.

```
[root@node1:~]# pcs quorum config
Options:
Device:
 Model: net
 algorithm: ffsplit
 host: qdevice
```

The **pcs quorum status** command shows the quorum runtime status, indicating that the quorum device is in use. The meanings of the **Qdevice** membership information status values for each cluster node are as follows:

- **A/NA** – The quorum device is alive or not alive, indicating whether there is a heartbeat between **qdevice** and **corosync**. This should always indicate that the quorum device is alive.
- **V/NV** – **V** is set when the quorum device has given a vote to a node. In this example, both nodes are set to **V** since they can communicate with each other. If the cluster were to split into two single-node clusters, one of the nodes would be set to **V** and the other node would be set to **NV**.
- **MW/NMW** – The quorum device **master\_wins** flag is set or not set. By default the flag is not set and the value is **NMW** (Not Master Wins). For information on the **master\_wins** flag see the **votequorum\_qdevice\_master\_wins(3)** man page.

```
[root@node1:~]# pcs quorum status
Quorum information

Date: Wed Jun 29 13:17:02 2016
Quorum provider: corosync_votequorum
Nodes: 2
Node ID: 1
Ring ID: 1/8272
Quorate: Yes

Votequorum information

Expected votes: 3
Highest expected: 3
Total votes: 3
Quorum: 2
Flags: Quorate Qdevice

Membership information

Nodeid Votes Qdevice Name
```

```

1 1 A,V,NMW node1 (local)
2 1 A,V,NMW node2
0 1 Qdevice

```

The **pcs quorum device status** shows the quorum device runtime status.

```

[root@node1:~]# pcs quorum device status
Qdevice information

Model: Net
Node ID: 1
Configured node list:
 0 Node ID = 1
 1 Node ID = 2
Membership node list: 1, 2

Qdevice-net information

Cluster name: mycluster
QNetd host: qdevice:5403
Algorithm: ffsplit
Tie-breaker: Node with lowest node ID
State: Connected

```

From the quorum device side, you can execute the following status command, which shows the status of the **corosync-qnetd** daemon.

```

[root@qdevice:~]# pcs qdevice status net --full
QNetd address: *:5403
TLS: Supported (client certificate required)
Connected clients: 2
Connected clusters: 1
Maximum send/receive size: 32768/32768 bytes
Cluster "mycluster":
 Algorithm: ffsplit
 Tie-breaker: Node with lowest node ID
 Node ID 2:
 Client address: ::ffff:192.168.122.122:50028
 HB interval: 8000ms
 Configured node list: 1, 2
 Ring ID: 1.2050
 Membership node list: 1, 2
 TLS active: Yes (client certificate verified)
 Vote: ACK (ACK)
 Node ID 1:
 Client address: ::ffff:192.168.122.121:48786
 HB interval: 8000ms
 Configured node list: 1, 2
 Ring ID: 1.2050
 Membership node list: 1, 2
 TLS active: Yes (client certificate verified)
 Vote: ACK (ACK)

```

### 108.5.3. Managing the Quorum Device Service

PCS provides the ability to manage the quorum device service on the local host (**corosync-qnetd**), as shown in the following example commands. Note that these commands affect only the **corosync-qnetd** service.

```
[root@qdevice:~]# pcs qdevice start net
[root@qdevice:~]# pcs qdevice stop net
[root@qdevice:~]# pcs qdevice enable net
[root@qdevice:~]# pcs qdevice disable net
[root@qdevice:~]# pcs qdevice kill net
```

#### 108.5.4. Managing the quorum device settings in a cluster

The following sections describe the PCS commands that you can use to manage the quorum device settings in a cluster.

##### 108.5.4.1. Changing quorum device settings

You can change the setting of a quorum device with the **pcs quorum device update** command.



##### WARNING

To change the **host** option of quorum device model **net**, use the **pcs quorum device remove** and the **pcs quorum device add** commands to set up the configuration properly, unless the old and the new host are the same machine.

The following command changes the quorum device algorithm to **lms**.

```
[root@node1:~]# pcs quorum device update model algorithm=lms
Sending updated corosync.conf to nodes...
node1: Succeeded
node2: Succeeded
Corosync configuration reloaded
Reloading qdevice configuration on nodes...
node1: corosync-qdevice stopped
node2: corosync-qdevice stopped
node1: corosync-qdevice started
node2: corosync-qdevice started
```

##### 108.5.4.2. Removing a quorum device

Use the following command to remove a quorum device configured on a cluster node.

```
[root@node1:~]# pcs quorum device remove
Sending updated corosync.conf to nodes...
node1: Succeeded
node2: Succeeded
Corosync configuration reloaded
Disabling corosync-qdevice...
```

```
node1: corosync-qdevice disabled
node2: corosync-qdevice disabled
Stopping corosync-qdevice...
node1: corosync-qdevice stopped
node2: corosync-qdevice stopped
Removing qdevice certificates from nodes...
node1: Succeeded
node2: Succeeded
```

After you have removed a quorum device, you should see the following error message when displaying the quorum device status.

```
[root@node1:~]# pcs quorum device status
Error: Unable to get quorum status: corosync-qdevice-tool: Can't connect to QDevice socket (is
QDevice running?): No such file or directory
```

#### 108.5.4.3. Destroying a quorum device

To disable and stop a quorum device on the quorum device host and delete all of its configuration files, use the following command.

```
[root@qdevice:~]# pcs qdevice destroy net
Stopping quorum device...
quorum device stopped
quorum device disabled
Quorum device 'net' configuration files removed
```

# CHAPTER 109. INTEGRATING NON-COROSYNC NODES INTO A CLUSTER: THE PACEMAKER\_REMOTE SERVICE

The **pacemaker\_remote** service allows nodes not running **corosync** to integrate into the cluster and have the cluster manage their resources just as if they were real cluster nodes.

Among the capabilities that the **pacemaker\_remote** service provides are the following:

- The **pacemaker\_remote** service allows you to scale beyond the Red Hat support limit of 32 nodes for RHEL 8.1.
- The **pacemaker\_remote** service allows you to manage a virtual environment as a cluster resource and also to manage individual services within the virtual environment as cluster resources.

The following terms are used to describe the **pacemaker\_remote** service.

- *cluster node* – A node running the High Availability services (**pacemaker** and **corosync**).
- *remote node* – A node running **pacemaker\_remote** to remotely integrate into the cluster without requiring **corosync** cluster membership. A remote node is configured as a cluster resource that uses the **ocf:pacemaker:remote** resource agent.
- *guest node* – A virtual guest node running the **pacemaker\_remote** service. The virtual guest resource is managed by the cluster; it is both started by the cluster and integrated into the cluster as a remote node.
- *pacemaker\_remote* – A service daemon capable of performing remote application management within remote nodes and KVM guest nodes in a Pacemaker cluster environment. This service is an enhanced version of Pacemaker’s local executor daemon (**pacemaker-execd**) that is capable of managing resources remotely on a node not running corosync.

A Pacemaker cluster running the **pacemaker\_remote** service has the following characteristics.

- Remote nodes and guest nodes run the **pacemaker\_remote** service (with very little configuration required on the virtual machine side).
- The cluster stack (**pacemaker** and **corosync**), running on the cluster nodes, connects to the **pacemaker\_remote** service on the remote nodes, allowing them to integrate into the cluster.
- The cluster stack (**pacemaker** and **corosync**), running on the cluster nodes, launches the guest nodes and immediately connects to the **pacemaker\_remote** service on the guest nodes, allowing them to integrate into the cluster.

The key difference between the cluster nodes and the remote and guest nodes that the cluster nodes manage is that the remote and guest nodes are not running the cluster stack. This means the remote and guest nodes have the following limitations:

- they do not take place in quorum
- they do not execute fencing device actions
- they are not eligible to be the cluster’s Designated Controller (DC)
- they do not themselves run the full range of **pcs** commands

On the other hand, remote nodes and guest nodes are not bound to the scalability limits associated with the cluster stack.

Other than these noted limitations, the remote and guest nodes behave just like cluster nodes in respect to resource management, and the remote and guest nodes can themselves be fenced. The cluster is fully capable of managing and monitoring resources on each remote and guest node: You can build constraints against them, put them in standby, or perform any other action you perform on cluster nodes with the **pcs** commands. Remote and guest nodes appear in cluster status output just as cluster nodes do.

## 109.1. HOST AND GUEST AUTHENTICATION OF PACEMAKER\_REMOTE NODES

The connection between cluster nodes and pacemaker\_remote is secured using Transport Layer Security (TLS) with pre-shared key (PSK) encryption and authentication over TCP (using port 3121 by default). This means both the cluster node and the node running **pacemaker\_remote** must share the same private key. By default this key must be placed at **/etc/pacemaker/authkey** on both cluster nodes and remote nodes.

The **pcs cluster node add-guest** command sets up the **authkey** for guest nodes and the **pcs cluster node add-remote** command sets up the **authkey** for remote nodes.

## 109.2. CONFIGURING KVM GUEST NODES

A Pacemaker guest node is a virtual guest node running the **pacemaker\_remote** service. The virtual guest node is managed by the cluster.

### 109.2.1. Guest node resource options

When configuring a virtual machine to act as a guest node, you create a **VirtualDomain** resource, which manages the virtual machine. For descriptions of the options you can set for a **VirtualDomain** resource, see [Table 107.1, “Resource Options for Virtual Domain Resources”](#).

In addition to the **VirtualDomain** resource options, metadata options define the resource as a guest node and define the connection parameters. You set these resource options with the **pcs cluster node add-guest** command. [Table 109.1, “Metadata Options for Configuring KVM Resources as Remote Nodes”](#) describes these metadata options.

**Table 109.1. Metadata Options for Configuring KVM Resources as Remote Nodes**

Field	Default	Description
<b>remote-node</b>	<none>	The name of the guest node this resource defines. This both enables the resource as a guest node and defines the unique name used to identify the guest node. <b>WARNING:</b> This value cannot overlap with any resource or node IDs.

Field	Default	Description
<b>remote-port</b>	3121	Configures a custom port to use for the guest connection to <b>pacemaker_remote</b>
<b>remote-addr</b>	The address provided in the <b>pcs host auth</b> command	The IP address or host name to connect to
<b>remote-connect-timeout</b>	60s	Amount of time before a pending guest connection will time out

### 109.2.2. Integrating a virtual machine as a guest node

The following procedure is a high-level summary overview of the steps to perform to have Pacemaker launch a virtual machine and to integrate that machine as a guest node, using **libvirt** and KVM virtual guests.

1. Configure the **VirtualDomain** resources.
2. Enter the following commands on every virtual machine to install **pacemaker\_remote** packages, start the **pcsd** service and enable it to run on startup, and allow TCP port 3121 through the firewall.

```
yum install pacemaker-remote resource-agents pcs
systemctl start pcasd.service
systemctl enable pcasd.service
firewall-cmd --add-port 3121/tcp --permanent
firewall-cmd --add-port 2224/tcp --permanent
firewall-cmd --reload
```

3. Give each virtual machine a static network address and unique host name, which should be known to all nodes. For information on setting a static IP address for the guest virtual machine, see the *Virtualization Deployment and Administration Guide*.
4. If you have not already done so, authenticate **pcs** to the node you will be integrating as a guest node.

```
pcs host auth nodename
```

5. Use the following command to convert an existing **VirtualDomain** resource into a guest node. This command must be run on a cluster node and not on the guest node which is being added. In addition to converting the resource, this command copies the **/etc/pacemaker/authkey** to the guest node and starts and enables the **pacemaker\_remote** daemon on the guest node. The node name for the guest node, which you can define arbitrarily, can differ from the host name for the node.

```
pcs cluster node add-guest nodename resource_id [options]
```

6. After creating the **VirtualDomain** resource, you can treat the guest node just as you would treat any other node in the cluster. For example, you can create a resource and place a resource constraint on the resource to run on the guest node as in the following commands, which are run

from a cluster node. You can include guest nodes in groups, which allows you to group a storage device, file system, and VM.

```
pcs resource create webserver apache configfile=/etc/httpd/conf/httpd.conf op
monitor interval=30s
pcs constraint location webserver prefers nodename
```

## 109.3. CONFIGURING PACEMAKER REMOTE NODES

A remote node is defined as a cluster resource with **ocf:pacemaker:remote** as the resource agent. You create this resource with the **pcs cluster node add-remote** command.

### 109.3.1. Remote node resource options

[Table 109.2, “Resource Options for Remote Nodes”](#) describes the resource options you can configure for a **remote** resource.

Table 109.2. Resource Options for Remote Nodes

Field	Default	Description
<b>reconnect_interval</b>	0	Time in seconds to wait before attempting to reconnect to a remote node after an active connection to the remote node has been severed. This wait is recurring. If reconnect fails after the wait period, a new reconnect attempt will be made after observing the wait time. When this option is in use, Pacemaker will keep attempting to reach out and connect to the remote node indefinitely after each wait interval.
<b>server</b>	Address specified with <b>pcs host auth</b> command	Server to connect to. This can be an IP address or host name.
<b>port</b>		TCP port to connect to.

### 109.3.2. Remote node configuration overview

This section provides a high-level summary overview of the steps to perform to configure a Pacemaker Remote node and to integrate that node into an existing Pacemaker cluster environment.

1. On the node that you will be configuring as a remote node, allow cluster-related services through the local firewall.

```
firewall-cmd --permanent --add-service=high-availability
success
firewall-cmd --reload
success
```



### NOTE

If you are using **iptables** directly, or some other firewall solution besides **firewalld**, simply open the following ports: TCP ports 2224 and 3121.

2. Install the **pacemaker\_remote** daemon on the remote node.

```
yum install -y pacemaker-remote resource-agents pcs
```

3. Start and enable **pcsd** on the remote node.

```
systemctl start pcsd.service
systemctl enable pcsd.service
```

4. If you have not already done so, authenticate **pcs** to the node you will be adding as a remote node.

```
pcs host auth remote1
```

5. Add the remote node resource to the cluster with the following command. This command also syncs all relevant configuration files to the new node, starts the node, and configures it to start **pacemaker\_remote** on boot. This command must be run on a cluster node and not on the remote node which is being added.

```
pcs cluster node add-remote remote1
```

6. After adding the **remote** resource to the cluster, you can treat the remote node just as you would treat any other node in the cluster. For example, you can create a resource and place a resource constraint on the resource to run on the remote node as in the following commands, which are run from a cluster node.

```
pcs resource create webserver apache configfile=/etc/httpd/conf/httpd.conf op
monitor interval=30s
pcs constraint location webserver prefers remote1
```



### WARNING

Never involve a remote node connection resource in a resource group, colocation constraint, or order constraint.

7. Configure fencing resources for the remote node. Remote nodes are fenced the same way as cluster nodes. Configure fencing resources for use with remote nodes the same as you would with cluster nodes. Note, however, that remote nodes can never initiate a fencing action. Only cluster nodes are capable of actually executing a fencing operation against another node.

## 109.4. CHANGING THE DEFAULT PORT LOCATION

If you need to change the default port location for either Pacemaker or **pacemaker\_remote**, you can set the **PCMK\_remote\_port** environment variable that affects both of these daemons. This environment variable can be enabled by placing it in the **/etc/sysconfig/pacemaker** file as follows.

```
====# Pacemaker Remote
...
#
Specify a custom port for Pacemaker Remote connections
PCMK_remote_port=3121
```

When changing the default port used by a particular guest node or remote node, the **PCMK\_remote\_port** variable must be set in that node's **/etc/sysconfig/pacemaker** file, and the cluster resource creating the guest node or remote node connection must also be configured with the same port number (using the **remote-port** metadata option for guest nodes, or the **port** option for remote nodes).

## 109.5. UPGRADING SYSTEMS WITH PACEMAKER\_REMOTE NODES

If the **pacemaker\_remote** service is stopped on an active Pacemaker Remote node, the cluster will gracefully migrate resources off the node before stopping the node. This allows you to perform software upgrades and other routine maintenance procedures without removing the node from the cluster. Once **pacemaker\_remote** is shut down, however, the cluster will immediately try to reconnect. If **pacemaker\_remote** is not restarted within the resource's monitor timeout, the cluster will consider the monitor operation as failed.

If you wish to avoid monitor failures when the **pacemaker\_remote** service is stopped on an active Pacemaker Remote node, you can use the following procedure to take the node out of the cluster before performing any system administration that might stop **pacemaker\_remote**

1. Stop the node's connection resource with the **pcs resource disable resourcename**, which will move all services off the node. For guest nodes, this will also stop the VM, so the VM must be started outside the cluster (for example, using **virsh**) to perform any maintenance.
2. Perform the required maintenance.
3. When ready to return the node to the cluster, re-enable the resource with the **pcs resource enable**.

# CHAPTER 110. PERFORMING CLUSTER MAINTENANCE

In order to perform maintenance on the nodes of your cluster, you may need to stop or move the resources and services running on that cluster. Or you may need to stop the cluster software while leaving the services untouched. Pacemaker provides a variety of methods for performing system maintenance.

- If you need to stop a node in a cluster while continuing to provide the services running on that cluster on another node, you can put the cluster node in standby mode. A node that is in standby mode is no longer able to host resources. Any resource currently active on the node will be moved to another node, or stopped if no other node is eligible to run the resource. For information on standby mode, see [Putting a node into standby mode](#).
- If you need to move an individual resource off the node on which it is currently running without stopping that resource, you can use the **pcs resource move** command to move the resource to a different node. For information on the **pcs resource move** command, see [Manually moving cluster resources](#).

When you execute the **pcs resource move** command, this adds a constraint to the resource to prevent it from running on the node on which it is currently running. When you are ready to move the resource back, you can execute the **pcs resource clear** or the **pcs constraint delete** command to remove the constraint. This does not necessarily move the resources back to the original node, however, since where the resources can run at that point depends on how you have configured your resources initially. You can relocate a resource to its preferred node with the **pcs resource relocate run** command, as described in [Moving a resource to its preferred node](#).

- If you need to stop a running resource entirely and prevent the cluster from starting it again, you can use the **pcs resource disable** command. For information on the **pcs resource disable** command, see [Enabling, disabling, and banning cluster resources](#).
- If you want to prevent Pacemaker from taking any action for a resource (for example, if you want to disable recovery actions while performing maintenance on the resource, or if you need to reload the `/etc/sysconfig/pacemaker` settings), use the **pcs resource unmanage** command, as described in [Setting a resource to unmanaged mode](#). Pacemaker Remote connection resources should never be unmanaged.
- If you need to put the cluster in a state where no services will be started or stopped, you can set the **maintenance-mode** cluster property. Putting the cluster into maintenance mode automatically unmanages all resources. For information on putting the cluster in maintenance mode, see [Putting a cluster in maintenance mode](#).
- If you need to update the packages that make up the RHEL High Availability and Resilient Storage Add-Ons, you can update the packages on one node at a time or on the entire cluster as a whole, as summarized in [Updating a Red Hat Enterprise Linux high availability cluster](#).
- If you need to perform maintenance on a Pacemaker remote node, you can remove that node from the cluster by disabling the remote node resource, as described in [Upgrading remote nodes and guest nodes](#).

## 110.1. PUTTING A NODE INTO STANDBY MODE

When a cluster node is in standby mode, the node is no longer able to host resources. Any resources currently active on the node will be moved to another node.

The following command puts the specified node into standby mode. If you specify the **--all**, this command puts all nodes into standby mode.

You can use this command when updating a resource's packages. You can also use this command when testing a configuration, to simulate recovery without actually shutting down a node.

```
pcs node standby node | --all
```

The following command removes the specified node from standby mode. After running this command, the specified node is then able to host resources. If you specify the **--all**, this command removes all nodes from standby mode.

```
pcs node unstandby node | --all
```

Note that when you execute the **pcs node standby** command, this prevents resources from running on the indicated node. When you execute the **pcs node unstandby** command, this allows resources to run on the indicated node. This does not necessarily move the resources back to the indicated node; where the resources can run at that point depends on how you have configured your resources initially.

## 110.2. MANUALLY MOVING CLUSTER RESOURCES

You can override the cluster and force resources to move from their current location. There are two occasions when you would want to do this:

- When a node is under maintenance, and you need to move all resources running on that node to a different node
- When individually specified resources needs to be moved

To move all resources running on a node to a different node, you put the node in standby mode.

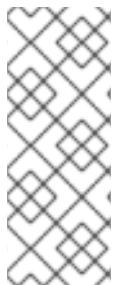
You can move individually specified resources in either of the following ways.

- You can use the **pcs resource move** command to move a resource off a node on which it is currently running.
- You can use the **pcs resource relocate run** command to move a resource to its preferred node, as determined by current cluster status, constraints, location of resources and other settings.

### 110.2.1. Moving a resource from its current node

To move a resource off the node on which it is currently running, use the following command, specifying the *resource\_id* of the resource as defined. Specify the **destination\_node** if you want to indicate on which node to run the resource that you are moving.

```
pcs resource move resource_id [destination_node] [--master] [lifetime=lifetime]
```



#### NOTE

When you execute the **pcs resource move** command, this adds a constraint to the resource to prevent it from running on the node on which it is currently running. You can execute the **pcs resource clear** or the **pcs constraint delete** command to remove the constraint. This does not necessarily move the resources back to the original node; where the resources can run at that point depends on how you have configured your resources initially.

If you specify the **--master** parameter of the **pcs resource move** command, the scope of the constraint is limited to the master role and you must specify *master\_id* rather than *resource\_id*.

You can optionally configure a **lifetime** parameter for the **pcs resource move** command to indicate a period of time the constraint should remain. You specify the units of a **lifetime** parameter according to the format defined in ISO 8601, which requires that you specify the unit as a capital letter such as Y (for years), M (for months), W (for weeks), D (for days), H (for hours), M (for minutes), and S (for seconds).

To distinguish a unit of minutes(M) from a unit of months(M), you must specify PT before indicating the value in minutes. For example, a **lifetime** parameter of 5M indicates an interval of five months, while a **lifetime** parameter of PT5M indicates an interval of five minutes.

The **lifetime** parameter is checked at intervals defined by the **cluster-recheck-interval** cluster property. By default this value is 15 minutes. If your configuration requires that you check this parameter more frequently, you can reset this value with the following command.

```
pcs property set cluster-recheck-interval=value
```

You can optionally configure a **--wait[=n]** parameter for the **pcs resource move** command to indicate the number of seconds to wait for the resource to start on the destination node before returning 0 if the resource is started or 1 if the resource has not yet started. If you do not specify n, the default resource timeout will be used.

The following command moves the resource **resource1** to node **example-node2** and prevents it from moving back to the node on which it was originally running for one hour and thirty minutes.

```
pcs resource move resource1 example-node2 lifetime=PT1H30M
```

The following command moves the resource **resource1** to node **example-node2** and prevents it from moving back to the node on which it was originally running for thirty minutes.

```
pcs resource move resource1 example-node2 lifetime=PT30M
```

### 110.2.2. Moving a resource to its preferred node

After a resource has moved, either due to a failover or to an administrator manually moving the node, it will not necessarily move back to its original node even after the circumstances that caused the failover have been corrected. To relocate resources to their preferred node, use the following command. A preferred node is determined by the current cluster status, constraints, resource location, and other settings and may change over time.

```
pcs resource relocate run [resource1] [resource2] ...
```

If you do not specify any resources, all resource are relocated to their preferred nodes.

This command calculates the preferred node for each resource while ignoring resource stickiness. After calculating the preferred node, it creates location constraints which will cause the resources to move to their preferred nodes. Once the resources have been moved, the constraints are deleted automatically. To remove all constraints created by the **pcs resource relocate run** command, you can enter the **pcs resource relocate clear** command. To display the current status of resources and their optimal node ignoring resource stickiness, enter the **pcs resource relocate show** command.

## 110.3. DISABLING, ENABLING, AND BANNING CLUSTER RESOURCES

In addition to the **pcs resource move** and **pcs resource relocate** commands, there are a variety of other commands you can use to control the behavior of cluster resources.

## Disabling a cluster resource

You can manually stop a running resource and prevent the cluster from starting it again with the following command. Depending on the rest of the configuration (constraints, options, failures, and so on), the resource may remain started. If you specify the **--wait** option, **pcs** will wait up to 'n' seconds for the resource to stop and then return 0 if the resource is stopped or 1 if the resource has not stopped. If 'n' is not specified it defaults to 60 minutes.

```
pcs resource disable resource_id [--wait[=n]]
```

As of Red Hat Enterprise Linux 8.2, you can specify that a resource be disabled only if disabling the resource would not have an effect on other resources. Ensuring that this would be the case can be impossible to do by hand when complex resource relations are set up.

- The **pcs resource disable --simulate** command shows the effects of disabling a resource while not changing the cluster configuration.
- The **pcs resource disable --safe** command disables a resource only if no other resources would be affected in any way, such as being migrated from one node to another. The **pcs resource safe-disable** command is an alias for the **pcs resource disable --safe** command.
- The **pcs resource disable --safe --no-strict** command disables a resource only if no other resources would be stopped or demoted

## Enabling a cluster resource

Use the following command to allow the cluster to start a resource. Depending on the rest of the configuration, the resource may remain stopped. If you specify the **--wait** option, **pcs** will wait up to 'n' seconds for the resource to start and then return 0 if the resource is started or 1 if the resource has not started. If 'n' is not specified it defaults to 60 minutes.

```
pcs resource enable resource_id [--wait[=n]]
```

## Preventing a resource from running on a particular node

Use the following command to prevent a resource from running on a specified node, or on the current node if no node is specified.

```
pcs resource ban resource_id [node] [--master] [lifetime=lifetime] [--wait[=n]]
```

Note that when you execute the **pcs resource ban** command, this adds a **-INFINITY** location constraint to the resource to prevent it from running on the indicated node. You can execute the **pcs resource clear** or the **pcs constraint delete** command to remove the constraint. This does not necessarily move the resources back to the indicated node; where the resources can run at that point depends on how you have configured your resources initially.

If you specify the **--master** parameter of the **pcs resource ban** command, the scope of the constraint is limited to the master role and you must specify *master\_id* rather than *resource\_id*.

You can optionally configure a **lifetime** parameter for the **pcs resource ban** command to indicate a period of time the constraint should remain.

You can optionally configure a **--wait[=n]** parameter for the **pcs resource ban** command to indicate the number of seconds to wait for the resource to start on the destination node before returning 0 if the resource is started or 1 if the resource has not yet started. If you do not specify n, the default

resource timeout will be used.

### Forcing a resource to start on the current node

Use the **debug-start** parameter of the **pcs resource** command to force a specified resource to start on the current node, ignoring the cluster recommendations and printing the output from starting the resource. This is mainly used for debugging resources; starting resources on a cluster is (almost) always done by Pacemaker and not directly with a **pcs** command. If your resource is not starting, it is usually due to either a misconfiguration of the resource (which you debug in the system log), constraints that prevent the resource from starting, or the resource being disabled. You can use this command to test resource configuration, but it should not normally be used to start resources in a cluster.

The format of the **debug-start** command is as follows.

```
pcs resource debug-start resource_id
```

## 110.4. SETTING A RESOURCE TO UNMANAGED MODE

When a resource is in **unmanaged** mode, the resource is still in the configuration but Pacemaker does not manage the resource.

The following command sets the indicated resources to **unmanaged** mode.

```
pcs resource unmanage resource1 [resource2] ...
```

The following command sets resources to **managed** mode, which is the default state.

```
pcs resource manage resource1 [resource2] ...
```

You can specify the name of a resource group with the **pcs resource manage** or **pcs resource unmanage** command. The command will act on all of the resources in the group, so that you can set all of the resources in a group to **managed** or **unmanaged** mode with a single command and then manage the contained resources individually.

## 110.5. PUTTING A CLUSTER IN MAINTENANCE MODE

When a cluster is in maintenance mode, the cluster does not start or stop any services until told otherwise. When maintenance mode is completed, the cluster does a sanity check of the current state of any services, and then stops or starts any that need it.

To put a cluster in maintenance mode, use the following command to set the **maintenance-mode** cluster property to **true**.

```
pcs property set maintenance-mode=true
```

To remove a cluster from maintenance mode, use the following command to set the **maintenance-mode** cluster property to **false**.

```
pcs property set maintenance-mode=false
```

You can remove a cluster property from the configuration with the following command.

```
pcs property unset property
```

Alternately, you can remove a cluster property from a configuration by leaving the value field of the **pcs property set** command blank. This restores that property to its default value. For example, if you have previously set the **symmetric-cluster** property to **false**, the following command removes the value you have set from the configuration and restores the value of **symmetric-cluster** to **true**, which is its default value.

```
pcs property set symmetric-cluster=
```

## 110.6. UPDATING A RHEL HIGH AVAILABILITY CLUSTER

Updating packages that make up the RHEL High Availability and Resilient Storage Add-Ons, either individually or as a whole, can be done in one of two general ways:

- *Rolling Updates*: Remove one node at a time from service, update its software, then integrate it back into the cluster. This allows the cluster to continue providing service and managing resources while each node is updated.
- *Entire Cluster Update*: Stop the entire cluster, apply updates to all nodes, then start the cluster back up.



### WARNING

It is critical that when performing software update procedures for Red Hat Enterprise Linux High Availability and Resilient Storage clusters, you ensure that any node that will undergo updates is not an active member of the cluster before those updates are initiated.

For a full description of each of these methods and the procedures to follow for the updates, see [Recommended Practices for Applying Software Updates to a RHEL High Availability or Resilient Storage Cluster](#).

## 110.7. UPGRADING REMOTE NODES AND GUEST NODES

If the **pacemaker\_remote** service is stopped on an active remote node or guest node, the cluster will gracefully migrate resources off the node before stopping the node. This allows you to perform software upgrades and other routine maintenance procedures without removing the node from the cluster. Once **pacemaker\_remote** is shut down, however, the cluster will immediately try to reconnect. If **pacemaker\_remote** is not restarted within the resource's monitor timeout, the cluster will consider the monitor operation as failed.

If you wish to avoid monitor failures when the **pacemaker\_remote** service is stopped on an active Pacemaker Remote node, you can use the following procedure to take the node out of the cluster before performing any system administration that might stop **pacemaker\_remote**

1. Stop the node's connection resource with the **pcs resource disable resourcename**, which will move all services off the node. For guest nodes, this will also stop the VM, so the VM must be started outside the cluster (for example, using **virsh**) to perform any maintenance.
2. Perform the required maintenance.

3. When ready to return the node to the cluster, re-enable the resource with the **pcs resource enable**.

## CHAPTER 111. CONFIGURING AND MANAGING LOGICAL VOLUMES

## CHAPTER 112. LOGICAL VOLUMES

Volume management creates a layer of abstraction over physical storage, allowing you to create logical storage volumes. This provides much greater flexibility in a number of ways than using physical storage directly. In addition, the hardware storage configuration is hidden from the software so it can be resized and moved without stopping applications or unmounting file systems. This can reduce operational costs.

Logical volumes provide the following advantages over using physical storage directly:

- **Flexible capacity**

When using logical volumes, file systems can extend across multiple disks, since you can aggregate disks and partitions into a single logical volume.

- **Resizeable storage pools**

You can extend logical volumes or reduce logical volumes in size with simple software commands, without reformatting and repartitioning the underlying disk devices.

- **Online data relocation**

To deploy newer, faster, or more resilient storage subsystems, you can move data while your system is active. Data can be rearranged on disks while the disks are in use. For example, you can empty a hot-swappable disk before removing it.

- **Convenient device naming**

Logical storage volumes can be managed in user-defined and custom named groups.

- **Disk striping**

You can create a logical volume that stripes data across two or more disks. This can dramatically increase throughput.

- **Mirroring volumes**

Logical volumes provide a convenient way to configure a mirror for your data.

- **Volume snapshots**

Using logical volumes, you can take device snapshots for consistent backups or to test the effect of changes without affecting the real data.

- **Thin volumes**

Logical volumes can be thinly provisioned. This allows you to create logical volumes that are larger than the available extents.

- **Cache volumes**

A cache logical volume uses a small logical volume consisting of fast block devices (such as SSD drives) to improve the performance of a larger and slower logical volume by storing the frequently used blocks on the smaller, faster logical volume.

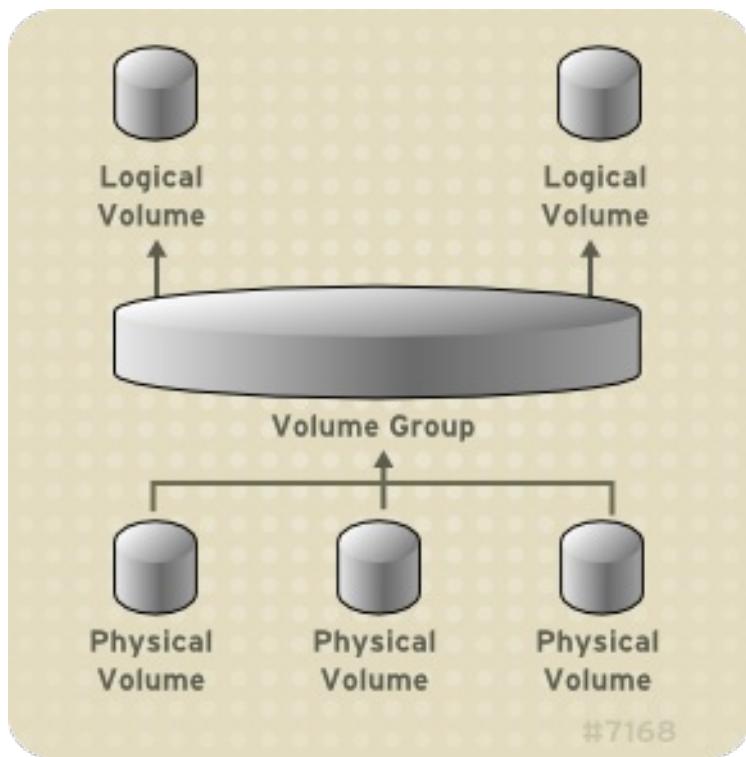
### 112.1. LVM ARCHITECTURE OVERVIEW

The underlying physical storage unit of an LVM logical volume is a block device such as a partition or whole disk. This device is initialized as an LVM *physical volume* (PV).

To create an LVM logical volume, the physical volumes are combined into a *volume group* (VG). This creates a pool of disk space out of which LVM logical volumes (LVs) can be allocated. This process is analogous to the way in which disks are divided into partitions. A logical volume is used by file systems and applications (such as databases).

Figure 112.1, “LVM logical volume components” shows the components of a simple LVM logical volume:

Figure 112.1. LVM logical volume components



## 112.2. PHYSICAL VOLUMES

The underlying physical storage unit of an LVM logical volume is a block device such as a partition or whole disk. To use the device for an LVM logical volume, the device must be initialized as a physical volume (PV). Initializing a block device as a physical volume places a label near the start of the device.

By default, the LVM label is placed in the second 512-byte sector. You can overwrite this default by placing the label on any of the first 4 sectors when you create the physical volume. This allows LVM volumes to co-exist with other users of these sectors, if necessary.

An LVM label provides correct identification and device ordering for a physical device, since devices can come up in any order when the system is booted. An LVM label remains persistent across reboots and throughout a cluster.

The LVM label identifies the device as an LVM physical volume. It contains a random unique identifier (the UUID) for the physical volume. It also stores the size of the block device in bytes, and it records where the LVM metadata will be stored on the device.

The LVM metadata contains the configuration details of the LVM volume groups on your system. By default, an identical copy of the metadata is maintained in every metadata area in every physical volume within the volume group. LVM metadata is small and stored as ASCII.

Currently LVM allows you to store 0, 1 or 2 identical copies of its metadata on each physical volume. The default is 1 copy. Once you configure the number of metadata copies on the physical volume, you cannot change that number at a later time. The first copy is stored at the start of the device, shortly after the label. If there is a second copy, it is placed at the end of the device. If you accidentally overwrite the area at the beginning of your disk by writing to a different disk than you intend, a second copy of the metadata at the end of the device will allow you to recover the metadata.

### 112.2.1. LVM physical volume layout

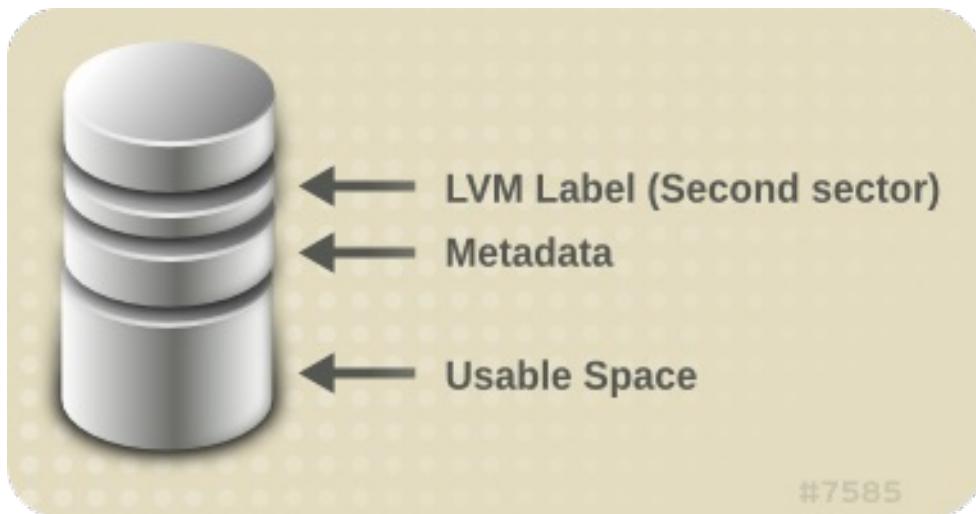
Figure 112.2, “Physical volume layout” shows the layout of an LVM physical volume. The LVM label is on the second sector, followed by the metadata area, followed by the usable space on the device.



### NOTE

In the Linux kernel (and throughout this document), sectors are considered to be 512 bytes in size.

Figure 112.2. Physical volume layout



#### 112.2.2. Multiple partitions on a disk

LVM allows you to create physical volumes out of disk partitions. Red Hat recommends that you create a single partition that covers the whole disk to label as an LVM physical volume for the following reasons:

- Administrative convenience

It is easier to keep track of the hardware in a system if each real disk only appears once. This becomes particularly true if a disk fails. In addition, multiple physical volumes on a single disk may cause a kernel warning about unknown partition types at boot.

- Striping performance

LVM cannot tell that two physical volumes are on the same physical disk. If you create a striped logical volume when two physical volumes are on the same physical disk, the stripes could be on different partitions on the same disk. This would result in a decrease in performance rather than an increase.

Although it is not recommended, there may be specific circumstances when you will need to divide a disk into separate LVM physical volumes. For example, on a system with few disks it may be necessary to move data around partitions when you are migrating an existing system to LVM volumes. Additionally, if you have a very large disk and want to have more than one volume group for administrative purposes then it is necessary to partition the disk. If you do have a disk with more than one partition and both of those partitions are in the same volume group, take care to specify which partitions are to be included in a logical volume when creating striped volumes.

## 112.3. VOLUME GROUPS

Physical volumes are combined into volume groups (VGs). This creates a pool of disk space out of which logical volumes can be allocated.

Within a volume group, the disk space available for allocation is divided into units of a fixed-size called extents. An extent is the smallest unit of space that can be allocated. Within a physical volume, extents are referred to as physical extents.

A logical volume is allocated into logical extents of the same size as the physical extents. The extent size is thus the same for all logical volumes in the volume group. The volume group maps the logical extents to physical extents.

## 112.4. LVM LOGICAL VOLUMES

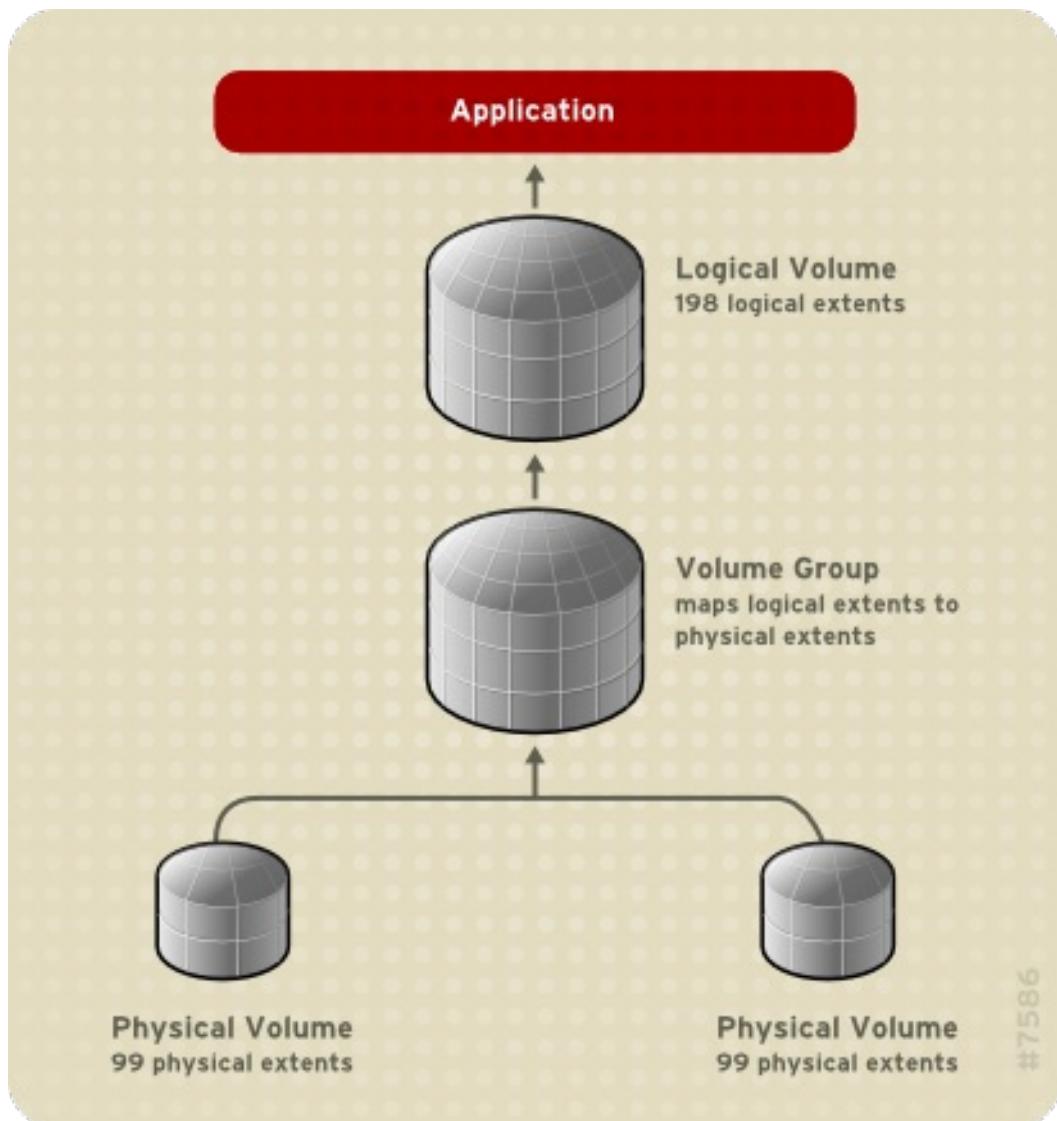
In LVM, a volume group is divided up into logical volumes. The following sections describe the different types of logical volumes.

### 112.4.1. Linear Volumes

A linear volume aggregates space from one or more physical volumes into one logical volume. For example, if you have two 60GB disks, you can create a 120GB logical volume. The physical storage is concatenated.

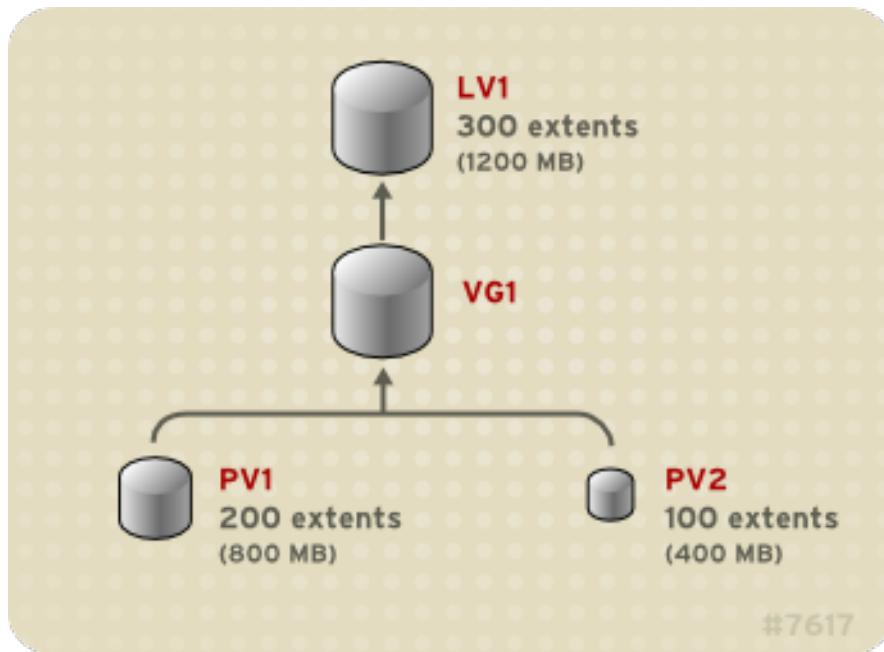
Creating a linear volume assigns a range of physical extents to an area of a logical volume in order. For example, as shown in [Figure 112.3, “Extent Mapping”](#) logical extents 1 to 99 could map to one physical volume and logical extents 100 to 198 could map to a second physical volume. From the point of view of the application, there is one device that is 198 extents in size.

Figure 112.3. Extent Mapping



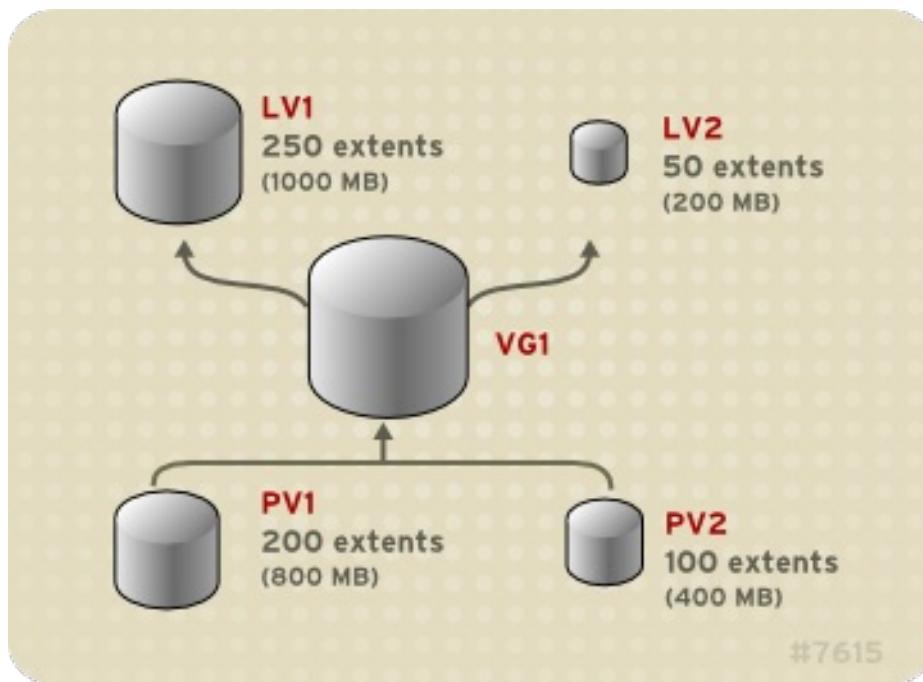
The physical volumes that make up a logical volume do not have to be the same size. Figure 112.4, “[Linear volume with unequal physical volumes](#)” shows volume group **VG1** with a physical extent size of 4MB. This volume group includes 2 physical volumes named **PV1** and **PV2**. The physical volumes are divided into 4MB units, since that is the extent size. In this example, **PV1** is 200 extents in size (800MB) and **PV2** is 100 extents in size (400MB). You can create a linear volume any size between 1 and 300 extents (4MB to 1200MB). In this example, the linear volume named **LV1** is 300 extents in size.

Figure 112.4. Linear volume with unequal physical volumes



You can configure more than one linear logical volume of whatever size you require from the pool of physical extents. [Figure 112.5, “Multiple logical volumes”](#) shows the same volume group as in [Figure 112.4, “Linear volume with unequal physical volumes”](#), but in this case two logical volumes have been carved out of the volume group: **LV1**, which is 250 extents in size (1000MB) and **LV2** which is 50 extents in size (200MB).

Figure 112.5. Multiple logical volumes



#### 112.4.2. Striped Logical Volumes

When you write data to an LVM logical volume, the file system lays the data out across the underlying physical volumes. You can control the way the data is written to the physical volumes by creating a striped logical volume. For large sequential reads and writes, this can improve the efficiency of the data I/O.

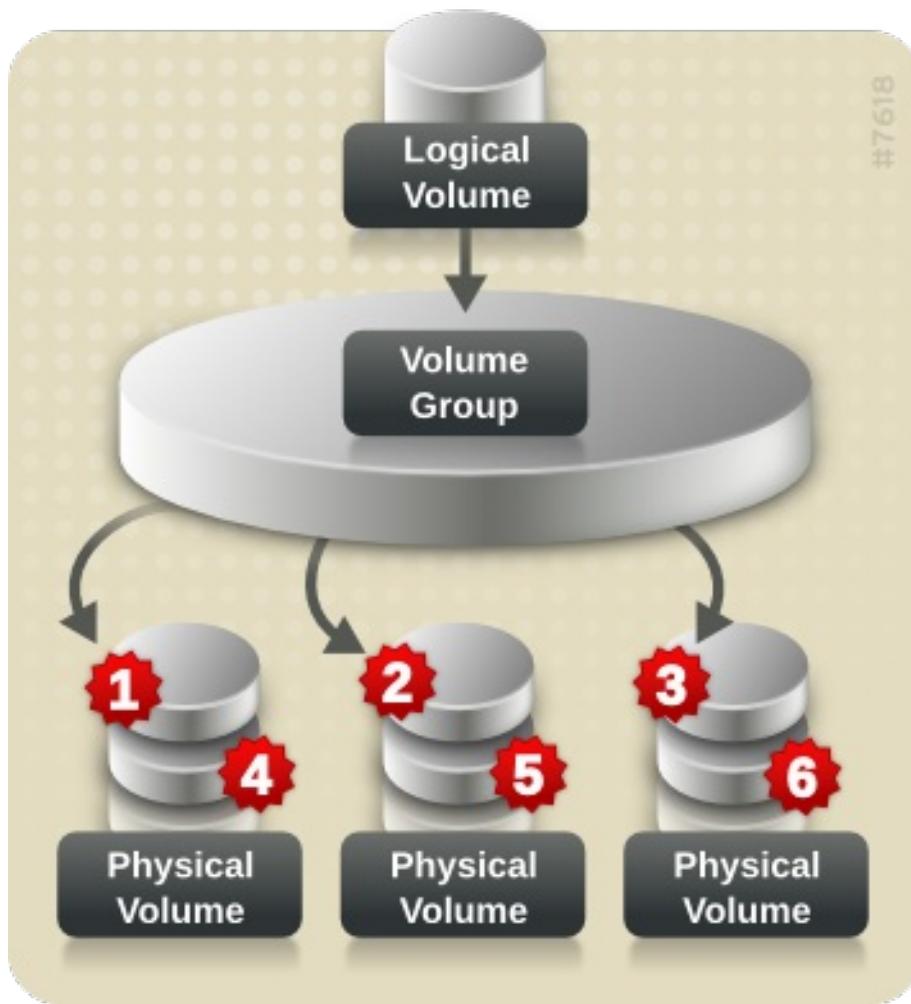
Striping enhances performance by writing data to a predetermined number of physical volumes in round-robin fashion. With striping, I/O can be done in parallel. In some situations, this can result in near-linear performance gain for each additional physical volume in the stripe.

The following illustration shows data being striped across three physical volumes. In this figure:

- the first stripe of data is written to the first physical volume
- the second stripe of data is written to the second physical volume
- the third stripe of data is written to the third physical volume
- the fourth stripe of data is written to the first physical volume

In a striped logical volume, the size of the stripe cannot exceed the size of an extent.

**Figure 112.6. Striping data across three PVs**



Striped logical volumes can be extended by concatenating another set of devices onto the end of the first set. In order to extend a striped logical volume, however, there must be enough free space on the set of underlying physical volumes that make up the volume group to support the stripe. For example, if you have a two-way stripe that uses up an entire volume group, adding a single physical volume to the volume group will not enable you to extend the stripe. Instead, you must add at least two physical volumes to the volume group.

### 112.4.3. RAID logical volumes

LVM supports RAID0/1/4/5/6/10. An LVM RAID volume has the following characteristics:

- RAID logical volumes created and managed by means of LVM leverage the MD kernel drivers.
- RAID1 images can be temporarily split from the array and merged back into the array later.
- LVM RAID volumes support snapshots.



#### NOTE

RAID logical volumes are not cluster-aware. While RAID logical volumes can be created and activated exclusively on one machine, they cannot be activated simultaneously on more than one machine.

#### 112.4.4. Thinly-provisioned logical volumes (thin volumes)

Logical volumes can be thinly provisioned. This allows you to create logical volumes that are larger than the available extents. Using thin provisioning, you can manage a storage pool of free space, known as a thin pool, which can be allocated to an arbitrary number of devices when needed by applications. You can then create devices that can be bound to the thin pool for later allocation when an application actually writes to the logical volume. The thin pool can be expanded dynamically when needed for cost-effective allocation of storage space.



#### NOTE

Thin volumes are not supported across the nodes in a cluster. The thin pool and all its thin volumes must be exclusively activated on only one cluster node.

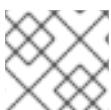
By using thin provisioning, a storage administrator can overcommit the physical storage, often avoiding the need to purchase additional storage. For example, if ten users each request a 100GB file system for their application, the storage administrator can create what appears to be a 100GB file system for each user but which is backed by less actual storage that is used only when needed. When using thin provisioning, it is important that the storage administrator monitor the storage pool and add more capacity if it starts to become full.

To make sure that all available space can be used, LVM supports data discard. This allows for re-use of the space that was formerly used by a discarded file or other block range.

Thin volumes provide support for a new implementation of copy-on-write (COW) snapshot logical volumes, which allow many virtual devices to share the same data in the thin pool.

#### 112.4.5. Snapshot Volumes

The LVM snapshot feature provides the ability to create virtual images of a device at a particular instant without causing a service interruption. When a change is made to the original device (the origin) after a snapshot is taken, the snapshot feature makes a copy of the changed data area as it was prior to the change so that it can reconstruct the state of the device.



#### NOTE

LVM supports thinly-provisioned snapshots.

Because a snapshot copies only the data areas that change after the snapshot is created, the snapshot feature requires a minimal amount of storage. For example, with a rarely updated origin, 3-5 % of the origin's capacity is sufficient to maintain the snapshot.



## NOTE

Snapshot copies of a file system are virtual copies, not an actual media backup for a file system. Snapshots do not provide a substitute for a backup procedure.

The size of the snapshot governs the amount of space set aside for storing the changes to the origin volume. For example, if you made a snapshot and then completely overwrote the origin the snapshot would have to be at least as big as the origin volume to hold the changes. You need to dimension a snapshot according to the expected level of change. So for example a short-lived snapshot of a read-mostly volume, such as `/usr`, would need less space than a long-lived snapshot of a volume that sees a greater number of writes, such as `/home`.

If a snapshot runs full, the snapshot becomes invalid, since it can no longer track changes on the origin volume. You should regularly monitor the size of the snapshot. Snapshots are fully resizable, however, so if you have the storage capacity you can increase the size of the snapshot volume to prevent it from getting dropped. Conversely, if you find that the snapshot volume is larger than you need, you can reduce the size of the volume to free up space that is needed by other logical volumes.

When you create a snapshot file system, full read and write access to the origin stays possible. If a chunk on a snapshot is changed, that chunk is marked and never gets copied from the original volume.

There are several uses for the snapshot feature:

- Most typically, a snapshot is taken when you need to perform a backup on a logical volume without halting the live system that is continuously updating the data.
- You can execute the **fsck** command on a snapshot file system to check the file system integrity and determine whether the original file system requires file system repair.
- Because the snapshot is read/write, you can test applications against production data by taking a snapshot and running tests against the snapshot, leaving the real data untouched.
- You can create LVM volumes for use with Red Hat Virtualization. LVM snapshots can be used to create snapshots of virtual guest images. These snapshots can provide a convenient way to modify existing guests or create new guests with minimal additional storage.

You can use the **--merge** option of the **lvconvert** command to merge a snapshot into its origin volume. One use for this feature is to perform system rollback if you have lost data or files or otherwise need to restore your system to a previous state. After you merge the snapshot volume, the resulting logical volume will have the origin volume's name, minor number, and UUID and the merged snapshot is removed.

### 112.4.6. Thinly-provisioned snapshot volumes

Red Hat Enterprise Linux provides support for thinly-provisioned snapshot volumes. Thin snapshot volumes allow many virtual devices to be stored on the same data volume. This simplifies administration and allows for the sharing of data between snapshot volumes.

As for all LVM snapshot volumes, as well as all thin volumes, thin snapshot volumes are not supported across the nodes in a cluster. The snapshot volume must be exclusively activated on only one cluster node.

Thin snapshot volumes provide the following benefits:

- A thin snapshot volume can reduce disk usage when there are multiple snapshots of the same origin volume.

- If there are multiple snapshots of the same origin, then a write to the origin will cause one COW operation to preserve the data. Increasing the number of snapshots of the origin should yield no major slowdown.
- Thin snapshot volumes can be used as a logical volume origin for another snapshot. This allows for an arbitrary depth of recursive snapshots (snapshots of snapshots of snapshots...).
- A snapshot of a thin logical volume also creates a thin logical volume. This consumes no data space until a COW operation is required, or until the snapshot itself is written.
- A thin snapshot volume does not need to be activated with its origin, so a user may have only the origin active while there are many inactive snapshot volumes of the origin.
- When you delete the origin of a thinly-provisioned snapshot volume, each snapshot of that origin volume becomes an independent thinly-provisioned volume. This means that instead of merging a snapshot with its origin volume, you may choose to delete the origin volume and then create a new thinly-provisioned snapshot using that independent volume as the origin volume for the new snapshot.

Although there are many advantages to using thin snapshot volumes, there are some use cases for which the older LVM snapshot volume feature may be more appropriate to your needs:

- You cannot change the chunk size of a thin pool. If the thin pool has a large chunk size (for example, 1MB) and you require a short-living snapshot for which a chunk size that large is not efficient, you may elect to use the older snapshot feature.
- You cannot limit the size of a thin snapshot volume; the snapshot will use all of the space in the thin pool, if necessary. This may not be appropriate for your needs.

In general, you should consider the specific requirements of your site when deciding which snapshot format to use.

#### 112.4.7. Cache Volumes

LVM supports the use of fast block devices (such as SSD drives) as write-back or write-through caches for larger slower block devices. Users can create cache logical volumes to improve the performance of their existing logical volumes or create new cache logical volumes composed of a small and fast device coupled with a large and slow device.

# CHAPTER 113. CONFIGURING LVM LOGICAL VOLUMES

The following procedures provide examples of basic LVM administration tasks.

## 113.1. USING CLI COMMANDS

The following sections describe some general operational features of LVM CLI commands.

### Specifying units in a command line argument

When sizes are required in a command line argument, units can always be specified explicitly. If you do not specify a unit, then a default is assumed, usually KB or MB. LVM CLI commands do not accept fractions.

When specifying units in a command line argument, LVM is case-insensitive; specifying M or m is equivalent, for example, and powers of 2 (multiples of 1024) are used. However, when specifying the **--units** argument in a command, lower-case indicates that units are in multiples of 1024 while upper-case indicates that units are in multiples of 1000.

### Specifying volume groups and logical volumes

Note the following when specifying volume groups or logical volumes in an LVM CLI command.

- Where commands take volume group or logical volume names as arguments, the full path name is optional. A logical volume called **lvol0** in a volume group called **vg0** can be specified as **vg0/lvol0**.
- Where a list of volume groups is required but is left empty, a list of all volume groups will be substituted.
- Where a list of logical volumes is required but a volume group is given, a list of all the logical volumes in that volume group will be substituted. For example, the **lvdisplay vg0** command will display all the logical volumes in volume group **vg0**.

### Increasing output verbosity

All LVM commands accept a **-v** argument, which can be entered multiple times to increase the output verbosity. The following examples shows the default output of the **lvcreate** command.

```
lvcreate -L 50MB new_vg
Rounding up size to full physical extent 52.00 MB
Logical volume "lvol0" created
```

The following command shows the output of the **lvcreate** command with the **-v** argument.

```
lvcreate -v -L 50MB new_vg
Rounding up size to full physical extent 52.00 MB
Archiving volume group "new_vg" metadata (seqno 1).
Creating logical volume lvol0
Creating volume group backup "/etc/lvm/backup/new_vg" (seqno 2).
Activating logical volume new_vg/lvol0.
activation/volume_list configuration setting not defined: Checking only host tags for new_vg/lvol0.
Creating new_vg-lvol0
Loading table for new_vg-lvol0 (253:0).
Resuming new_vg-lvol0 (253:0).
Wiping known signatures on logical volume "new_vg/lvol0"
Initializing 4.00 KiB of logical volume "new_vg/lvol0" with value 0.
Logical volume "lvol0" created
```

The **-vv**, **-vvv** and the **-vvvv** arguments display increasingly more details about the command execution. The **-vvvv** argument provides the maximum amount of information at this time. The following example shows the first few lines of output for the **lvcreate** command with the **-vvvv** argument specified.

```
lvcreate -vvvv -L 50MB new_vg
#lvmcmdline.c:913 Processing: lvcreate -vvvv -L 50MB new_vg
#lvmcmdline.c:916 O_DIRECT will be used
#config/config.c:864 Setting global/locking_type to 1
#locking/locking.c:138 File-based locking selected.
#config/config.c:841 Setting global/locking_dir to /var/lock/lvm
#activate/activate.c:358 Getting target version for linear
#ioctl/libdm-iface.c:1569 dm version OF [16384]
#ioctl/libdm-iface.c:1569 dm versions OF [16384]
#activate/activate.c:358 Getting target version for striped
#ioctl/libdm-iface.c:1569 dm versions OF [16384]
#config/config.c:864 Setting activation/mirror_region_size to 512
...
...
```

### Displaying help for LVM CLI commands

You can display help for any of the LVM CLI commands with the **--help** argument of the command.

```
commandname --help
```

To display the man page for a command, execute the **man** command:

```
man commandname
```

The **man lvm** command provides general online information about LVM.

## 113.2. CREATING AN LVM LOGICAL VOLUME ON THREE DISKS

This example procedure creates an LVM logical volume called **mylv** that consists of the disks at **/dev/sda1**, **/dev/sdb1**, and **/dev/sdc1**.

1. To use disks in a volume group, label them as LVM physical volumes with the **pvcreate** command.



### WARNING

This command destroys any data on **/dev/sda1**, **/dev/sdb1**, and **/dev/sdc1**.

```
pvcreate /dev/sda1 /dev/sdb1 /dev/sdc1
Physical volume "/dev/sda1" successfully created
Physical volume "/dev/sdb1" successfully created
Physical volume "/dev/sdc1" successfully created
```

2. Create the a volume group that consists of the LVM physical volumes you have created. The following command creates the volume group **myvg**.

```
vgcreate myvg /dev/sda1 /dev/sdb1 /dev/sdc1
Volume group "myvg" successfully created
```

You can use the **vgs** command to display the attributes of the new volume group.

```
vgs
VG #PV #LV #SN Attr VSize VFree
myvg 3 0 0 wz--n- 51.45G 51.45G
```

3. Create the logical volume from the volume group you have created. The following command creates the logical volume **mylv** from the volume group **myvg**. This example creates a logical volume that uses 2 gigabytes of the volume group.

```
lvcreate -L 2G -n mylv myvg
Logical volume "mylv" created
```

4. Create a file system on the logical volume. The following command creates an **ext4** file system on the logical volume.

```
mkfs.ext4 /dev/myvg/mylv
mke2fs 1.44.3 (10-July-2018)
Creating filesystem with 524288 4k blocks and 131072 inodes
Filesystem UUID: 616da032-8a48-4cd7-8705-bd94b7a1c8c4
Superblock backups stored on blocks:
32768, 98304, 163840, 229376, 294912
```

```
Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
```

The following commands mount the logical volume and report the file system disk space usage.

```
mount /dev/myvg/mylv /mnt
df
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/mapper/myvg-my lv 1998672 6144 1871288 1% /mnt
```

### 113.3. CREATING A RAID0 (STRIPED) LOGICAL VOLUME

A RAID0 logical volume spreads logical volume data across multiple data subvolumes in units of stripe size.

The format for the command to create a RAID0 volume is as follows.

```
lvcreate --type raid0[_meta] --stripes Stripes --stripesize StripeSize VolumeGroup
[PhysicalVolumePath ...]
```

Table 113.1. RAID0 Command Creation parameters

Parameter	Description
<b>--type raid0[_meta]</b>	Specifying <b>raid0</b> creates a RAID0 volume without metadata volumes. Specifying <b>raid0_meta</b> creates a RAID0 volume with metadata volumes. Because RAID0 is non-resilient, it does not have to store any mirrored data blocks as RAID1/10 or calculate and store any parity blocks as RAID4/5/6 do. Hence, it does not need metadata volumes to keep state about resynchronization progress of mirrored or parity blocks. Metadata volumes become mandatory on a conversion from RAID0 to RAID4/5/6/10, however, and specifying <b>raid0_meta</b> preallocates those metadata volumes to prevent a respective allocation failure.
<b>--stripes <i>Stripes</i></b>	Specifies the number of devices to spread the logical volume across.
<b>--stripesize <i>StripeSize</i></b>	Specifies the size of each stripe in kilobytes. This is the amount of data that is written to one device before moving to the next device.
<b>VolumeGroup</b>	Specifies the volume group to use.
<b>PhysicalVolumePath ...</b>	Specifies the devices to use. If this is not specified, LVM will choose the number of devices specified by the <i>Stripes</i> option, one for each stripe.

This example procedure creates an LVM RAID0 logical volume called **mylv** that stripes data across the disks at **/dev/sda1**, **/dev/sdb1**, and **/dev/sdc1**.

1. Label the disks you will use in the volume group as LVM physical volumes with the **pvccreate** command.



#### WARNING

This command destroys any data on **/dev/sda1**, **/dev/sdb1**, and **/dev/sdc1**.

```
pvccreate /dev/sda1 /dev/sdb1 /dev/sdc1
Physical volume "/dev/sda1" successfully created
Physical volume "/dev/sdb1" successfully created
Physical volume "/dev/sdc1" successfully created
```

2. Create the volume group **myvg**. The following command creates the volume group **myvg**.

```
vgcreate myvg /dev/sda1 /dev/sdb1 /dev/sdc1
Volume group "myvg" successfully created
```

You can use the **vgs** command to display the attributes of the new volume group.

```
vgs
VG #PV #LV #SN Attr VSize VFree
myvg 3 0 0 wz--n- 51.45G 51.45G
```

3. Create a RAID0 logical volume from the volume group you have created. The following command creates the RAID0 volume **mylv** from the volume group **myvg**. This example creates a logical volume that is 2 gigabytes in size, with three stripes and a stripe size of 4 kilobytes.

```
lvcreate --type raid0 -L 2G --stripes 3 --stripesize 4 -n mylv myvg
Rounding size 2.00 GiB (512 extents) up to stripe boundary size 2.00 GiB(513 extents).
Logical volume "mylv" created.
```

4. Create a file system on the RAID0 logical volume. The following command creates an **ext4** file system on the logical volume.

```
mkfs.ext4 /dev/myvg/mylv
mke2fs 1.44.3 (10-July-2018)
Creating filesystem with 525312 4k blocks and 131376 inodes
Filesystem UUID: 9d4c0704-6028-450a-8b0a-8875358c0511
Superblock backups stored on blocks:
 32768, 98304, 163840, 229376, 294912

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
```

The following commands mount the logical volume and report the file system disk space usage.

```
mount /dev/myvg/mylv /mnt
df
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/mapper/myvg-my lv 2002684 6168 1875072 1% /mnt
```

## 113.4. RENAMING LVM LOGICAL VOLUMES

This procedure renames an existing logical volume using the command-line LVM interface.

### Procedure

1. If the logical volume is currently mounted, unmount the volume.
2. If the logical volume exists in a clustered environment, deactivate the logical volume on all nodes where it is active. Use the following command on each such node:

```
[root@node-n]# lvchange --activate n vg-name/lv-name
```

3. Use the **lvrename** utility to rename an existing logical volume:

```
lvrename vg-name original-lv-name new-lv-name
```

Optionally, you can specify the full paths to the devices:

```
lvrename /dev/vg-name/original-lv-name /dev/vg-name/new-lv-name
```

## Additional resources

- The **lvrename(8)** man page

## 113.5. REMOVING A DISK FROM A LOGICAL VOLUME

These example procedures show how you can remove a disk from an existing logical volume, either to replace the disk or to use the disk as part of a different volume. In order to remove a disk, you must first move the extents on the LVM physical volume to a different disk or set of disks.

### 113.5.1. Moving extents to existing physical volumes

In this example, the logical volume is distributed across four physical volumes in the volume group **myvg**.

```
pvs -o+pv_used
PV VG Fmt Attr PSize PFree Used
/dev/sda1 myvg lvm2 a- 17.15G 12.15G 5.00G
/dev/sdb1 myvg lvm2 a- 17.15G 12.15G 5.00G
/dev/sdc1 myvg lvm2 a- 17.15G 12.15G 5.00G
/dev/sdd1 myvg lvm2 a- 17.15G 2.15G 15.00G
```

This example moves the extents off of **/dev/sdb1** so that it can be removed from the volume group.

1. If there are enough free extents on the other physical volumes in the volume group, you can execute the **pvmove** command on the device you want to remove with no other options and the extents will be distributed to the other devices.  
In a cluster, the **pvmove** command can move only logical volume that are active exclusively on a single node.

```
pvmove /dev/sdb1
/dev/sdb1: Moved: 2.0%
...
/dev/sdb1: Moved: 79.2%
...
/dev/sdb1: Moved: 100.0%
```

After the **pvmove** command has finished executing, the distribution of extents is as follows:

```
pvs -o+pv_used
PV VG Fmt Attr PSize PFree Used
/dev/sda1 myvg lvm2 a- 17.15G 7.15G 10.00G
/dev/sdb1 myvg lvm2 a- 17.15G 17.15G 0
/dev/sdc1 myvg lvm2 a- 17.15G 12.15G 5.00G
/dev/sdd1 myvg lvm2 a- 17.15G 2.15G 15.00G
```

2. Use the **vgreduce** command to remove the physical volume **/dev/sdb1** from the volume group.

```
vgreduce myvg /dev/sdb1
Removed "/dev/sdb1" from volume group "myvg"
pvs
PV VG Fmt Attr PSize PFree
/dev/sda1 myvg lvm2 a- 17.15G 7.15G
/dev/sdb1 lvm2 -- 17.15G 17.15G
/dev/sdc1 myvg lvm2 a- 17.15G 12.15G
/dev/sdd1 myvg lvm2 a- 17.15G 2.15G
```

The disk can now be physically removed or allocated to other users.

### 113.5.2. Moving Extents to a New Disk

In this example, the logical volume is distributed across three physical volumes in the volume group **myvg** as follows:

```
pvs -o+pv_used
PV VG Fmt Attr PSize PFree Used
/dev/sda1 myvg lvm2 a- 17.15G 7.15G 10.00G
/dev/sdb1 myvg lvm2 a- 17.15G 15.15G 2.00G
/dev/sdc1 myvg lvm2 a- 17.15G 15.15G 2.00G
```

This example procedure moves the extents of **/dev/sdb1** to a new device, **/dev/sdd1**.

1. Create a new physical volume from **/dev/sdd1**.

```
pvcreate /dev/sdd1
Physical volume "/dev/sdd1" successfully created
```

2. Add the new physical volume **/dev/sdd1** to the existing volume group **myvg**.

```
vgextend myvg /dev/sdd1
Volume group "myvg" successfully extended
pvs -o+pv_used
PV VG Fmt Attr PSize PFree Used
/dev/sda1 myvg lvm2 a- 17.15G 7.15G 10.00G
/dev/sdb1 myvg lvm2 a- 17.15G 15.15G 2.00G
/dev/sdc1 myvg lvm2 a- 17.15G 15.15G 2.00G
/dev/sdd1 myvg lvm2 a- 17.15G 17.15G 0
```

3. Use the **pvmove** command to move the data from **/dev/sdb1** to **/dev/sdd1**.

```
pvmove /dev/sdb1 /dev/sdd1
/dev/sdb1: Moved: 10.0%
...
/dev/sdb1: Moved: 79.7%
...
/dev/sdb1: Moved: 100.0%
pvs -o+pv_used
PV VG Fmt Attr PSize PFree Used
/dev/sda1 myvg lvm2 a- 17.15G 7.15G 10.00G
```

```
/dev/sdb1 myvg lvm2 a- 17.15G 17.15G 0
/dev/sdc1 myvg lvm2 a- 17.15G 15.15G 2.00G
/dev/sdd1 myvg lvm2 a- 17.15G 15.15G 2.00G
```

4. After you have moved the data off **/dev/sdb1**, you can remove it from the volume group.

```
vgreduce myvg /dev/sdb1
Removed "/dev/sdb1" from volume group "myvg"
```

You can now reallocate the disk to another volume group or remove the disk from the system.

## 113.6. CONFIGURING PERSISTENT DEVICE NUMBERS

Major and minor device numbers are allocated dynamically at module load. Some applications work best if the block device is always activated with the same device (major and minor) number. You can specify these with the **lvcreate** and the **lvchange** commands by using the following arguments:

```
--persistent y --major major --minor minor
```

Use a large minor number to be sure that it has not already been allocated to another device dynamically.

If you are exporting a file system using NFS, specifying the **fsid** parameter in the exports file may avoid the need to set a persistent device number within LVM.

## 113.7. SPECIFYING LVM EXTENT SIZE

When physical volumes are used to create a volume group, its disk space is divided into 4MB extents, by default. This extent is the minimum amount by which the logical volume may be increased or decreased in size. Large numbers of extents will have no impact on I/O performance of the logical volume.

You can specify the extent size with the **-s** option to the **vgcreate** command if the default extent size is not suitable. You can put limits on the number of physical or logical volumes the volume group can have by using the **-p** and **-l** arguments of the **vgcreate** command.

## 113.8. MANAGING LVM LOGICAL VOLUMES USING RHEL SYSTEM ROLES

This section describes how to apply the **storage** role to perform the following tasks:

- Create an LVM logical volume in a volume group consisting of multiple disks.
- Create an ext4 file system with a given label on the logical volume.
- Persistently mount the ext4 file system.

### Prerequisites

- An Ansible playbook including the **storage** role

For information on how to apply an Ansible playbook, see [Applying a role](#).

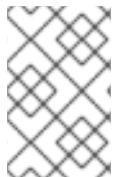
### 113.8.1. Example Ansible playbook to manage logical volumes

This section provides an example Ansible playbook. This playbook applies the **storage** role to create a LVM logical volume called **mylv** in a volume group called **myvg**. The volume group consists of the following disks:

- **/dev/sda**
- **/dev/sdb**
- **/dev/sdc**

The playbook creates an ext4 file system on the logical volume, and persistently mounts the file system.

```
- hosts: all
vars:
 storage_pools:
 - name: myvg
 disks:
 - sda
 - sdb
 - sdc
 volumes:
 - name: mylv
 size: 2G
 fs_type: ext4
 mount_point: /mnt
roles:
 - rhel-system-roles.storage
```



#### NOTE

If a volume group called **myvg** already exists, the logical volume is added to it.

If a volume group called **myvg** does not exist, it is created.

### 113.8.2. Additional resources

- For more information about the **storage** role, see [Managing local storage using RHEL System Roles](#).

## 113.9. REMOVING LVM LOGICAL VOLUMES

This procedure removes an existing logical volume using the command-line LVM interface.

The following commands remove the logical volume **/dev/vg-name/lv-name** from the volume group **vg-name**.

#### Procedure

1. If the logical volume is currently mounted, unmount the volume.
2. If the logical volume exists in a clustered environment, deactivate the logical volume on all nodes where it is active. Use the following command on each such node:

```
[root@node-n]# lvchange --activate n vg-name/lv-name
```

3. Remove the logical volume using the **lvremove** utility:

```
lvremove /dev/vg-name/lv-name
Do you really want to remove active logical volume "/lv-name"? [y/n]: y
Logical volume "/lv-name" successfully removed
```



### NOTE

In this case, the logical volume has not been deactivated. If you explicitly deactivated the logical volume before removing it, you would not see the prompt verifying whether you want to remove an active logical volume.

## Additional resources

- The **lvremove(8)** man page

# CHAPTER 114. MODIFYING THE SIZE OF A LOGICAL VOLUME

After you have created a logical volume, you can modify the size of the volume.

## 114.1. GROWING LOGICAL VOLUMES

To increase the size of a logical volume, use the **lvextend** command.

When you extend the logical volume, you can indicate how much you want to extend the volume, or how large you want it to be after you extend it.

The following command extends the logical volume **/dev/myvg/homevol** to 12 gigabytes.

```
lvextend -L12G /dev/myvg/homevol
lvextend -- extending logical volume "/dev/myvg/homevol" to 12 GB
lvextend -- doing automatic backup of volume group "myvg"
lvextend -- logical volume "/dev/myvg/homevol" successfully extended
```

The following command adds another gigabyte to the logical volume **/dev/myvg/homevol**.

```
lvextend -L+1G /dev/myvg/homevol
lvextend -- extending logical volume "/dev/myvg/homevol" to 13 GB
lvextend -- doing automatic backup of volume group "myvg"
lvextend -- logical volume "/dev/myvg/homevol" successfully extended
```

As with the **lvcreate** command, you can use the **-l** argument of the **lvextend** command to specify the number of extents by which to increase the size of the logical volume. You can also use this argument to specify a percentage of the volume group, or a percentage of the remaining free space in the volume group. The following command extends the logical volume called **testlv** to fill all of the unallocated space in the volume group **myvg**.

```
lvextend -l +100%FREE /dev/myvg/testlv
Extending logical volume testlv to 68.59 GB
Logical volume testlv successfully resized
```

After you have extended the logical volume it is necessary to increase the file system size to match.

By default, most file system resizing tools will increase the size of the file system to be the size of the underlying logical volume so you do not need to worry about specifying the same size for each of the two commands.

## 114.2. GROWING A FILE SYSTEM ON A LOGICAL VOLUME

To grow a file system on a logical volume, perform the following steps:

1. Determine whether there is sufficient unallocated space in the existing volume group to extend the logical volume. If not, perform the following procedure:
  - a. Create a new physical volume with the **pvcreate** command.
  - b. Use the **vgextend** command to extend the volume group that contains the logical volume with the file system you are growing to include the new physical volume.

2. Once the volume group is large enough to include the larger file system, extend the logical volume with the **lvresize** command.
3. Resize the file system on the logical volume.

Note that you can use the **-r** option of the **lvresize** command to extend the logical volume and resize the underlying file system with a single command

### 114.3. SHRINKING LOGICAL VOLUMES

You can reduce the size of a logical volume with the **lvreduce** command.



#### NOTE

Shrinking is not supported on a GFS2 or XFS file system, so you cannot reduce the size of a logical volume that contains a GFS2 or XFS file system.

If the logical volume you are reducing contains a file system, to prevent data loss you must ensure that the file system is not using the space in the logical volume that is being reduced. For this reason, it is recommended that you use the **--resizesfs** option of the **lvreduce** command when the logical volume contains a file system. When you use this option, the **lvreduce** command attempts to reduce the file system before shrinking the logical volume. If shrinking the file system fails, as can occur if the file system is full or the file system does not support shrinking, then the **lvreduce** command will fail and not attempt to shrink the logical volume.



#### WARNING

In most cases, the **lvreduce** command warns about possible data loss and asks for a confirmation. However, you should not rely on these confirmation prompts to prevent data loss because in some cases you will not see these prompts, such as when the logical volume is inactive or the **--resizesfs** option is not used.

Note that using the **--test** option of the **lvreduce** command does not indicate where the operation is safe, as this option does not check the file system or test the file system resize.

The following command shrinks the logical volume **lvol1** in volume group **vg00** to be 64 megabytes. In this example, **lvol1** contains a file system, which this command resizes together with the logical volume. This example shows the output to the command.

```
lvreduce --resizesfs -L 64M vg00/lvol1
fsck from util-linux 2.23.2
/dev/mapper/vg00-lvol1: clean, 11/25688 files, 8896/102400 blocks
resize2fs 1.42.9 (28-Dec-2013)
Resizing the filesystem on /dev/mapper/vg00-lvol1 to 65536 (1k) blocks.
The filesystem on /dev/mapper/vg00-lvol1 is now 65536 blocks long.
```

Size of logical volume vg00/lvol1 changed from 100.00 MiB (25 extents) to 64.00 MiB (16 extents).
Logical volume vg00/lvol1 successfully resized.

Specifying the `-` sign before the resize value indicates that the value will be subtracted from the logical volume's actual size. The following example shows the command you would use if, instead of shrinking a logical volume to an absolute size of 64 megabytes, you wanted to shrink the volume by a value 64 megabytes.

```
lvreduce --resizesfs -L -64M vg00/lvol1
```

## 114.4. EXTENDING A STRIPED LOGICAL VOLUME

In order to increase the size of a striped logical volume, there must be enough free space on the underlying physical volumes that make up the volume group to support the stripe. For example, if you have a two-way stripe that that uses up an entire volume group, adding a single physical volume to the volume group will not enable you to extend the stripe. Instead, you must add at least two physical volumes to the volume group.

For example, consider a volume group **vg** that consists of two underlying physical volumes, as displayed with the following **vgs** command.

```
vgs
VG #PV #LV #SN Attr VSize VFree
vg 2 0 0 wz--n- 271.31G 271.31G
```

You can create a stripe using the entire amount of space in the volume group.

```
lvcreate -n stripe1 -L 271.31G -i 2 vg
Using default stripesize 64.00 KB
Rounding up size to full physical extent 271.31 GB
Logical volume "stripe1" created
lvs -a -o +devices
LV VG Attr LSize Origin Snap% Move Log Copy% Devices
stripe1 vg -wi-a- 271.31G /dev/sda1(0),/dev/sdb1(0)
```

Note that the volume group now has no more free space.

```
vgs
VG #PV #LV #SN Attr VSize VFree
vg 2 1 0 wz--n- 271.31G 0
```

The following command adds another physical volume to the volume group, which then has 135 gigabytes of additional space.

```
vgextend vg /dev/sdc1
Volume group "vg" successfully extended
vgs
VG #PV #LV #SN Attr VSize VFree
vg 3 1 0 wz--n- 406.97G 135.66G
```

At this point you cannot extend the striped logical volume to the full size of the volume group, because two underlying devices are needed in order to stripe the data.

```
lvextend vg/stripe1 -L 406G
Using stripesize of last segment 64.00 KB
Extending logical volume stripe1 to 406.00 GB
```

Insufficient suitable allocatable extents for logical volume stripe1: 34480  
more required

To extend the striped logical volume, add another physical volume and then extend the logical volume. In this example, having added two physical volumes to the volume group we can extend the logical volume to the full size of the volume group.

```
vgextend vg /dev/sdd1
Volume group "vg" successfully extended
vgs
VG #PV #LV #SN Attr VSize VFree
vg 4 1 0 wz--n- 542.62G 271.31G
lvextend vg/stripe1 -L 542G
Using stripesize of last segment 64.00 KB
Extending logical volume stripe1 to 542.00 GB
Logical volume stripe1 successfully resized
```

If you do not have enough underlying physical devices to extend the striped logical volume, it is possible to extend the volume anyway if it does not matter that the extension is not striped, which may result in uneven performance. When adding space to the logical volume, the default operation is to use the same striping parameters of the last segment of the existing logical volume, but you can override those parameters. The following example extends the existing striped logical volume to use the remaining free space after the initial **lvextend** command fails.

```
lvextend vg/stripe1 -L 406G
Using stripesize of last segment 64.00 KB
Extending logical volume stripe1 to 406.00 GB
Insufficient suitable allocatable extents for logical volume stripe1: 34480
more required
lvextend -i1 -l+100%FREE vg/stripe1
```

# CHAPTER 115. MANAGING LVM PHYSICAL VOLUMES

There are a variety of commands and procedures you can use to manage LVM physical volumes.

## 115.1. SCANNING FOR BLOCK DEVICES TO USE AS PHYSICAL VOLUMES

You can scan for block devices that may be used as physical volumes with the **lvmdiskscan** command, as shown in the following example.

```
lvmdiskscan
/dev/ram0 [16.00 MB]
/dev/sda [17.15 GB]
/dev/root [13.69 GB]
/dev/ram [16.00 MB]
/dev/sda1 [17.14 GB] LVM physical volume
/dev/VolGroup00/LogVol01 [512.00 MB]
/dev/ram2 [16.00 MB]
/dev/new_vg/lvol0 [52.00 MB]
/dev/ram3 [16.00 MB]
/dev/pkl_new_vg/sparkie_lv [7.14 GB]
/dev/ram4 [16.00 MB]
/dev/ram5 [16.00 MB]
/dev/ram6 [16.00 MB]
/dev/ram7 [16.00 MB]
/dev/ram8 [16.00 MB]
/dev/ram9 [16.00 MB]
/dev/ram10 [16.00 MB]
/dev/ram11 [16.00 MB]
/dev/ram12 [16.00 MB]
/dev/ram13 [16.00 MB]
/dev/ram14 [16.00 MB]
/dev/ram15 [16.00 MB]
/dev/sdb [17.15 GB]
/dev/sdb1 [17.14 GB] LVM physical volume
/dev/sdc [17.15 GB]
/dev/sdc1 [17.14 GB] LVM physical volume
/dev/sdd [17.15 GB]
/dev/sdd1 [17.14 GB] LVM physical volume
7 disks
17 partitions
0 LVM physical volume whole disks
4 LVM physical volumes
```

## 115.2. SETTING THE PARTITION TYPE FOR A PHYSICAL VOLUME

If you are using a whole disk device for your physical volume, the disk must have no partition table. For DOS disk partitions, the partition id should be set to 0x8e using the **fdisk** or **cfdisk** command or an equivalent. For whole disk devices only the partition table must be erased, which will effectively destroy all data on that disk. You can remove an existing partition table by zeroing the first sector with the following command:

```
dd if=/dev/zero of=PhysicalVolume bs=512 count=1
```

## 115.3. RESIZING AN LVM PHYSICAL VOLUME

If you need to change the size of an underlying block device for any reason, use the **pvresize** command to update LVM with the new size. You can execute this command while LVM is using the physical volume.

## 115.4. REMOVING PHYSICAL VOLUMES

If a device is no longer required for use by LVM, you can remove the LVM label with the **pvremove** command. Executing the **pvremove** command zeroes the LVM metadata on an empty physical volume.

If the physical volume you want to remove is currently part of a volume group, you must remove it from the volume group with the **vgreduce** command.

```
pvremove /dev/ram15
Labels on physical volume "/dev/ram15" successfully wiped
```

## 115.5. ADDING PHYSICAL VOLUMES TO A VOLUME GROUP

To add additional physical volumes to an existing volume group, use the **vgextend** command. The **vgextend** command increases a volume group's capacity by adding one or more free physical volumes.

The following command adds the physical volume **/dev/sdf1** to the volume group **vg1**.

```
vgextend vg1 /dev/sdf1
```

## 115.6. REMOVING PHYSICAL VOLUMES FROM A VOLUME GROUP

To remove unused physical volumes from a volume group, use the **vgreduce** command. The **vgreduce** command shrinks a volume group's capacity by removing one or more empty physical volumes. This frees those physical volumes to be used in different volume groups or to be removed from the system.

Before removing a physical volume from a volume group, you can make sure that the physical volume is not used by any logical volumes by using the **pvdisplay** command.

```
pvdisplay /dev/hda1

-- Physical volume --
PV Name /dev/hda1
VG Name myvg
PV Size 1.95 GB / NOT usable 4 MB [LVM: 122 KB]
PV# 1
PV Status available
Allocatable yes (but full)
Cur LV 1
PE Size (KByte) 4096
Total PE 499
Free PE 0
Allocated PE 499
PV UUID Sd44tK-9IRw-SrMC-MOkn-76iP-iftz-OVSen7
```

If the physical volume is still being used you will have to migrate the data to another physical volume using the **pvmove** command. Then use the **vgreduce** command to remove the physical volume.

The following command removes the physical volume **/dev/hda1** from the volume group **my\_volume\_group**.

```
vgreduce my_volume_group /dev/hda1
```

If a logical volume contains a physical volume that fails, you cannot use that logical volume. To remove missing physical volumes from a volume group, you can use the **--removemissing** parameter of the **vgreduce** command, if there are no logical volumes that are allocated on the missing physical volumes.

If the physical volume that fails contains a mirror image of a logical volume of a **mirror** segment type, you can remove that image from the mirror with the **vgreduce --removemissing --mirroronly --force** command. This removes only the logical volumes that are mirror images from the physical volume.

# CHAPTER 116. DISPLAYING LVM COMPONENTS

LVM provides a variety of ways to display the LVM components, as well as to customize the display. This section summarizes the usage of the basic LVM display commands.

## 116.1. DISPLAYING LVM INFORMATION WITH THE LVM COMMAND

The **lvm** command provides several built-in options that you can use to display information about LVM support and configuration.

- **lvm devtypes**  
Displays the recognized built-in block device types
- **lvm formats**  
Displays recognized metadata formats.
- **lvm help**  
Displays LVM help text.
- **lvm segtypes**  
Displays recognized logical volume segment types.
- **lvm tags**  
Displays any tags defined on this host.
- **lvm version**  
Displays the current version information.

## 116.2. DISPLAYING PHYSICAL VOLUMES

There are three commands you can use to display properties of LVM physical volumes: **pvs**, **pvdisplay**, and **pvscan**.

The **pvs** command provides physical volume information in a configurable form, displaying one line per physical volume. The **pvs** command provides a great deal of format control, and is useful for scripting.

The **pvdisplay** command provides a verbose multi-line output for each physical volume. It displays physical properties (size, extents, volume group, and so on) in a fixed format.

The following example shows the output of the **pvdisplay** command for a single physical volume.

```
pvdisplay
--- Physical volume ---
PV Name /dev/sdc1
VG Name new_vg
PV Size 17.14 GB / not usable 3.40 MB
Allocatable yes
PE Size (KByte) 4096
Total PE 4388
Free PE 4375
Allocated PE 13
PV UUID Joqlch-yWSj-kuEn-ldwM-01S9-XO8M-mcpsVe
```

The **pvscan** command scans all supported LVM block devices in the system for physical volumes.

The following command shows all physical devices found:

```
pvscan
PV /dev/sdb2 VG vg0 lvm2 [964.00 MB / 0 free]
PV /dev/sdc1 VG vg0 lvm2 [964.00 MB / 428.00 MB free]
PV /dev/sdc2 lvm2 [964.84 MB]
Total: 3 [2.83 GB] / in use: 2 [1.88 GB] / in no VG: 1 [964.84 MB]
```

You can define a filter in the **lvm.conf** file so that this command will avoid scanning specific physical volumes.

### 116.3. DISPLAYING VOLUME GROUPS

There are two commands you can use to display properties of LVM volume groups: **vgs** and **vgdisplay**. The **vgscan** command, which scans all supported LVM block devices in the system for volume groups, can also be used to display the existing volume groups.

The **vgs** command provides volume group information in a configurable form, displaying one line per volume group. The **vgs** command provides a great deal of format control, and is useful for scripting.

The **vgdisplay** command displays volume group properties (such as size, extents, number of physical volumes, and so on) in a fixed form. The following example shows the output of the **vgdisplay** command for the volume group **new\_vg**. If you do not specify a volume group, all existing volume groups are displayed.

```
vgdisplay new_vg
--- Volume group ---
VG Name new_vg
System ID
Format lvm2
Metadata Areas 3
Metadata Sequence No 11
VG Access read/write
VG Status resizable
MAX LV 0
Cur LV 1
Open LV 0
Max PV 0
Cur PV 3
Act PV 3
VG Size 51.42 GB
PE Size 4.00 MB
Total PE 13164
Alloc PE / Size 13 / 52.00 MB
Free PE / Size 13151 / 51.37 GB
VG UUID jxQJ0a-ZKk0-OpMO-0118-nlwO-wwqd-fD5D32
```

The following example shows the output of the **vgscan** command.

```
vgscan
Reading all physical volumes. This may take a while...
Found volume group "new_vg" using metadata type lvm2
Found volume group "officevg" using metadata type lvm2
```

## 116.4. DISPLAYING LOGICAL VOLUMES

There are three commands you can use to display properties of LVM logical volumes: **lvs**, **lvdisplay**, and **lvscan**.

The **lvs** command provides logical volume information in a configurable form, displaying one line per logical volume. The **lvs** command provides a great deal of format control, and is useful for scripting.

The **lvdisplay** command displays logical volume properties (such as size, layout, and mapping) in a fixed format.

The following command shows the attributes of **lvol2** in **vg00**. If snapshot logical volumes have been created for this original logical volume, this command shows a list of all snapshot logical volumes and their status (active or inactive) as well.

```
lvdisplay -v /dev/vg00/lvol2
```

The **lvscan** command scans for all logical volumes in the system and lists them, as in the following example.

```
lvscan
ACTIVE '/dev/vg0/gfslv' [1.46 GB] inherit
```

## CHAPTER 117. CUSTOMIZED REPORTING FOR LVM

LVM provides a wide range of configuration and command line options to produce customized reports and to filter the report's output. For a full description of LVM reporting features and capabilities, see the **lvmreport(7)** man page.

You can produce concise and customizable reports of LVM objects with the **pvs**, **lvs**, and **vgs** commands. The reports that these commands generate include one line of output for each object. Each line contains an ordered list of fields of properties related to the object. There are five ways to select the objects to be reported: by physical volume, volume group, logical volume, physical volume segment, and logical volume segment.

You can report information about physical volumes, volume groups, logical volumes, physical volume segments, and logical volume segments all at once with the **lvm fullreport** command. For information on this command and its capabilities, see the **lvm-fullreport(8)** man page.

LVM supports log reports, which contain a log of operations, messages, and per-object status with complete object identification collected during LVM command execution. For further information about the LVM log report, see the **lvmreport(7)** man page.

### 117.1. CONTROLLING THE FORMAT OF THE LVM DISPLAY

Whether you use the **pvs**, **lvs**, or **vgs** command determines the default set of fields displayed and the sort order. You can control the output of these commands with the following arguments:

- You can change what fields are displayed to something other than the default by using the **-o** argument. For example, the following command displays only the physical volume name and size.

```
pvs -o pv_name,pv_size
PV PSize
/dev/sdb1 17.14G
/dev/sdc1 17.14G
/dev/sdd1 17.14G
```

- You can append a field to the output with the plus sign (+), which is used in combination with the **-o** argument.

The following example displays the UUID of the physical volume in addition to the default fields.

```
pvs -o +pv_uuid
PV VG Fmt Attr PSize PFree PV UUID
/dev/sdb1 new_vg lvm2 a- 17.14G 17.14G onFF2w-1fLC-ughJ-D9eB-M7iv-6XqA-dqGeXY
/dev/sdc1 new_vg lvm2 a- 17.14G 17.09G Joqlch-yWSj-kuEn-IdwM-01S9-X08M-mcpsVe
/dev/sdd1 new_vg lvm2 a- 17.14G 17.14G yfvZK-Cf31-j75k-dECm-0RZ3-0dGW-UqkCS
```

- Adding the **-v** argument to a command includes some extra fields. For example, the **pvs -v** command will display the **DevSize** and **PV UUID** fields in addition to the default fields.

```
pvs -v
Scanning for physical volume names
PV VG Fmt Attr PSize PFree DevSize PV UUID
/dev/sdb1 new_vg lvm2 a- 17.14G 17.14G 17.14G onFF2w-1fLC-ughJ-D9eB-M7iv-6XqA-dqGeXY
/dev/sdc1 new_vg lvm2 a- 17.14G 17.09G 17.14G Joqlch-yWSj-kuEn-IdwM-01S9-XO8M-
```

```
mcpsVe
/dev/sdd1 new_vg lvm2 a- 17.14G 17.14G 17.14G yfvfZK-Cf31-j75k-dECm-0RZ3-0dGW-
tUqkCS
```

- The **--noheadings** argument suppresses the headings line. This can be useful for writing scripts. The following example uses the **--noheadings** argument in combination with the **pv\_name** argument, which will generate a list of all physical volumes.

```
pvs --noheadings -o pv_name
/dev/sdb1
/dev/sdc1
/dev/sdd1
```

- The **--separator separator** argument uses *separator* to separate each field. The following example separates the default output fields of the **pvs** command with an equals sign (=).

```
pvs --separator =
PV=VG=Fmt=Attr=PSize=PFree
/dev/sdb1=new_vg=lvm2=a-=17.14G=17.14G
/dev/sdc1=new_vg=lvm2=a-=17.14G=17.09G
/dev/sdd1=new_vg=lvm2=a-=17.14G=17.14G
```

To keep the fields aligned when using the **separator** argument, use the **separator** argument in conjunction with the **--aligned** argument.

```
pvs --separator = --aligned
PV =VG =Fmt =Attr=PSize =PFree
/dev/sdb1 =new_vg=lvm2=a- =17.14G=17.14G
/dev/sdc1 =new_vg=lvm2=a- =17.14G=17.09G
/dev/sdd1 =new_vg=lvm2=a- =17.14G=17.14G
```

You can use the **-P** argument of the **lvs** or **vgs** command to display information about a failed volume that would otherwise not appear in the output.

For a full listing of display arguments, see the **pvs(8)**, **vgs(8)** and **lvs(8)** man pages.

Volume group fields can be mixed with either physical volume (and physical volume segment) fields or with logical volume (and logical volume segment) fields, but physical volume and logical volume fields cannot be mixed. For example, the following command will display one line of output for each physical volume.

```
vgs -o +pv_name
VG #PV #LV #SN Attr VSize VFree PV
new_vg 3 1 0 wz--n- 51.42G 51.37G /dev/sdc1
new_vg 3 1 0 wz--n- 51.42G 51.37G /dev/sdd1
new_vg 3 1 0 wz--n- 51.42G 51.37G /dev/sdb1
```

## 117.2. LVM OBJECT DISPLAY FIELDS

This section provides a series of tables that list the information you can display about the LVM objects with the **pvs**, **vgs**, and **lvs** commands.

For convenience, a field name prefix can be dropped if it matches the default for the command. For example, with the **pvs** command, **name** means **pv\_name**, but with the **vgs** command, **name** is interpreted as **vg\_name**.

Executing the following command is the equivalent of executing **pvs -o pv\_free**.

```
pvs -o free
PFree
17.14G
17.09G
17.14G
```



### NOTE

The number of characters in the attribute fields in **pvs**, **vgs**, and **lvs** output may increase in later releases. The existing character fields will not change position, but new fields may be added to the end. You should take this into account when writing scripts that search for particular attribute characters, searching for the character based on its relative position to the beginning of the field, but not for its relative position to the end of the field. For example, to search for the character **p** in the ninth bit of the **lv\_attr** field, you could search for the string "**^/.....p/**", but you should not search for the string "**/\*p\$/**".

[Table 117.1, "The pvs Command Display Fields"](#) lists the display arguments of the **pvs** command, along with the field name as it appears in the header display and a description of the field.

Table 117.1. The pvs Command Display Fields

Argument	Header	Description
<b>dev_size</b>	DevSize	The size of the underlying device on which the physical volume was created
<b>pe_start</b>	1st PE	Offset to the start of the first physical extent in the underlying device
<b>pv_attr</b>	Attr	Status of the physical volume: (a)llocatable or e(x)ported.
<b>pv_fmt</b>	Fmt	The metadata format of the physical volume ( <b>lvm2</b> or <b>lvm1</b> )
<b>pv_free</b>	PFree	The free space remaining on the physical volume
<b>pv_name</b>	PV	The physical volume name
<b>pv_pe_alloc_count</b>	Alloc	Number of used physical extents
<b>pv_pe_count</b>	PE	Number of physical extents
<b>pvseg_size</b>	SSize	The segment size of the physical volume
<b>pvseg_start</b>	Start	The starting physical extent of the physical volume segment

Argument	Header	Description
<b>pv_size</b>	PSize	The size of the physical volume
<b>pv_tags</b>	PV Tags	LVM tags attached to the physical volume
<b>pv_used</b>	Used	The amount of space currently used on the physical volume
<b>pv_uuid</b>	PV UUID	The UUID of the physical volume

The **pvs** command displays the following fields by default: **pv\_name**, **vg\_name**, **pv\_fmt**, **pv\_attr**, **pv\_size**, **pv\_free**. The display is sorted by **pv\_name**.

```
pvs
PV VG Fmt Attr PSize PFree
/dev/sdb1 new_vg lvm2 a- 17.14G 17.14G
/dev/sdc1 new_vg lvm2 a- 17.14G 17.09G
/dev/sdd1 new_vg lvm2 a- 17.14G 17.13G
```

Using the **-v** argument with the **pvs** command adds the following fields to the default display: **dev\_size**, **pv\_uuid**.

```
pvs -v
Scanning for physical volume names
PV VG Fmt Attr PSize PFree DevSize PV UUID
/dev/sdb1 new_vg lvm2 a- 17.14G 17.14G 17.14G onFF2w-1fLC-ughJ-D9eB-M7iv-6XqA-
dqGeXY
/dev/sdc1 new_vg lvm2 a- 17.14G 17.09G 17.14G Joqlch-yWSj-kuEn-ldwM-01S9-XO8M-mcpsVe
/dev/sdd1 new_vg lvm2 a- 17.14G 17.13G 17.14G yfvfZK-Cf31-j75k-dECm-0RZ3-0dGW-tUqkCS
```

You can use the **--segments** argument of the **pvs** command to display information about each physical volume segment. A segment is a group of extents. A segment view can be useful if you want to see whether your logical volume is fragmented.

The **pvs --segments** command displays the following fields by default: **pv\_name**, **vg\_name**, **pv\_fmt**, **pv\_attr**, **pv\_size**, **pv\_free**, **pvseg\_start**, **pvseg\_size**. The display is sorted by **pv\_name** and **pvseg\_size** within the physical volume.

```
pvs --segments
PV VG Fmt Attr PSize PFree Start SSize
/dev/hda2 VolGroup00 lvm2 a- 37.16G 32.00M 0 1172
/dev/hda2 VolGroup00 lvm2 a- 37.16G 32.00M 1172 16
/dev/hda2 VolGroup00 lvm2 a- 37.16G 32.00M 1188 1
/dev/sda1 vg lvm2 a- 17.14G 16.75G 0 26
/dev/sda1 vg lvm2 a- 17.14G 16.75G 26 24
/dev/sda1 vg lvm2 a- 17.14G 16.75G 50 26
/dev/sda1 vg lvm2 a- 17.14G 16.75G 76 24
/dev/sda1 vg lvm2 a- 17.14G 16.75G 100 26
/dev/sda1 vg lvm2 a- 17.14G 16.75G 126 24
/dev/sda1 vg lvm2 a- 17.14G 16.75G 150 22
/dev/sda1 vg lvm2 a- 17.14G 16.75G 172 4217
/dev/sdb1 vg lvm2 a- 17.14G 17.14G 0 4389
```

```
/dev/sdc1 vg lvm2 a- 17.14G 17.14G 0 4389
/dev/sdd1 vg lvm2 a- 17.14G 17.14G 0 4389
/dev/sde1 vg lvm2 a- 17.14G 17.14G 0 4389
/dev/sdf1 vg lvm2 a- 17.14G 17.14G 0 4389
/dev/sdg1 vg lvm2 a- 17.14G 17.14G 0 4389
```

You can use the **pvs -a** command to see devices detected by LVM that have not been initialized as LVM physical volumes.

```
pvs -a
PV VG Fmt Attr PSize PFree
/dev/VolGroup00/LogVol01 -- 0 0
/dev/new_vg/lvol0 -- 0 0
/dev/ram -- 0 0
/dev/ram0 -- 0 0
/dev/ram2 -- 0 0
/dev/ram3 -- 0 0
/dev/ram4 -- 0 0
/dev/ram5 -- 0 0
/dev/ram6 -- 0 0
/dev/root -- 0 0
/dev/sda -- 0 0
/dev/sdb -- 0 0
/dev/sdb1 new_vg lvm2 a- 17.14G 17.14G
/dev/sdc -- 0 0
/dev/sdc1 new_vg lvm2 a- 17.14G 17.09G
/dev/sdd -- 0 0
/dev/sdd1 new_vg lvm2 a- 17.14G 17.14G
```

[Table 117.2, “vgs Display Fields”](#) lists the display arguments of the **vgs** command, along with the field name as it appears in the header display and a description of the field.

**Table 117.2. vgs Display Fields**

Argument	Header	Description
<b>lv_count</b>	#LV	The number of logical volumes the volume group contains
<b>max_lv</b>	MaxLV	The maximum number of logical volumes allowed in the volume group (0 if unlimited)
<b>max_pv</b>	MaxPV	The maximum number of physical volumes allowed in the volume group (0 if unlimited)
<b>pv_count</b>	#PV	The number of physical volumes that define the volume group
<b>snap_count</b>	#SN	The number of snapshots the volume group contains
<b>vg_attr</b>	Attr	Status of the volume group: (w)ritable, (r)eadonly, resi(z)eable, e(x)ported, (p)artial and (c)lustered.

Argument	Header	Description
<b>vg_extent_count</b>	#Ext	The number of physical extents in the volume group
<b>vg_extent_size</b>	Ext	The size of the physical extents in the volume group
<b>vg_fmt</b>	Fmt	The metadata format of the volume group ( <b>lvm2</b> or <b>lvm1</b> )
<b>vg_free</b>	VFree	Size of the free space remaining in the volume group
<b>vg_free_count</b>	Free	Number of free physical extents in the volume group
<b>vg_name</b>	VG	The volume group name
<b>vg_seqno</b>	Seq	Number representing the revision of the volume group
<b>vg_size</b>	VSize	The size of the volume group
<b>vg_sysid</b>	SYS ID	LVM1 System ID
<b>vg_tags</b>	VG Tags	LVM tags attached to the volume group
<b>vg_uuid</b>	VG UUID	The UUID of the volume group

The **vgs** command displays the following fields by default: **vg\_name**, **pv\_count**, **lv\_count**, **snap\_count**, **vg\_attr**, **vg\_size**, **vg\_free**. The display is sorted by **vg\_name**.

```
vgs
VG #PV #LV #SN Attr VSize VFree
new_vg 3 1 1 wz--n- 51.42G 51.36G
```

Using the **-v** argument with the **vgs** command adds the following fields to the default display: **vg\_extent\_size**, **vg\_uuid**.

```
vgs -v
Finding all volume groups
Finding volume group "new_vg"
VG Attr Ext #PV #LV #SN VSize VFree VG UUID
new_vg wz--n- 4.00M 3 1 1 51.42G 51.36G jxQJ0a-ZKk0-OpMO-0118-nlwO-wwqd-fD5D32
```

[Table 117.3, “lvs Display Fields”](#) lists the display arguments of the **lvs** command, along with the field name as it appears in the header display and a description of the field.



## NOTE

In later releases of Red Hat Enterprise Linux, the output of the **lvs** command may differ, with additional fields in the output. The order of the fields, however, will remain the same and any additional fields will appear at the end of the display.

Table 117.3. lvs Display Fields

Argument	Header	Description
* <b>chunksize</b>	Chunk	Unit size in a snapshot volume
* <b>chunk_size</b>		
<b>copy_percent</b>	Copy%	The synchronization percentage of a mirrored logical volume; also used when physical extents are being moved with the <b>pv_move</b> command
<b>devices</b>	Devices	The underlying devices that make up the logical volume: the physical volumes, logical volumes, and start physical extents and logical extents
<b>lv_ancestors</b>	Ancestors	For thin pool snapshots, the ancestors of the logical volume
<b>lv_descendants</b>	Descendants	For thin pool snapshots, the descendants of the logical volume
<b>lv_attr</b>	Attr	<p>The status of the logical volume. The logical volume attribute bits are as follows:</p> <ul style="list-style-type: none"> <li>* Bit 1: Volume type: (m)irrored, (M)irrored without initial sync, (o)rigin, (O)rigin with merging snapshot, (r)aid, <sup>o</sup>aid without initial sync, (s)napshot, merging (S)napshot, (p)vmove, (v)irtual, mirror or raid (i)mage, mirror or raid (l)image out-of-sync, mirror (l)og device, under (c)onversion, thin (V)olume, (t)hin pool, (T)hin pool data, raid or thin pool m(e)tadata or pool metadata spare,</li> <li>* Bit 2: Permissions: (w)riteable, (r)ead-only, <sup>o</sup>ead-only activation of non-read-only volume</li> <li>* Bit 3: Allocation policy: (a)nywhere, (c)ontiguous, (i)nherited, c(l)ing, (n)ormal. This is capitalized if the volume is currently locked against allocation changes, for example while executing the <b>pvmove</b> command.</li> <li>* Bit 4: fixed (m)inor</li> <li>* Bit 5: State: (a)ctive, (s)uspended, (l)invalid snapshot, invalid (S)suspended snapshot, snapshot (m)erge failed, suspended snapshot (M)erge failed, mapped (d)evice present without tables, mapped device present with (i)nactive table</li> <li>* Bit 6: device (o)pen</li> <li>* Bit 7: Target type: (m)irror, (r)aid, (s)napshot, (t)hin, (u)nknown, (v)irtual. This groups logical volumes related to the same kernel target together. So, for example, mirror images, mirror logs as well as mirrors themselves appear as (m) if they use the original device-mapper mirror kernel</li> </ul>

Argument	Header	Description
		<p>driver, whereas the raid equivalents using the md raid kernel driver appear as (r). Snapshots using the original device-mapper driver appear as (s), whereas snapshots of thin volumes using the thin provisioning driver appear as (t).</p> <p>* Bit 8: Newly-allocated data blocks are overwritten with blocks of (z)eroes before use.</p> <p>* Bit 9: Volume Health: (p)artial, (r)efresh needed, (m)ismatches exist, (w)ritemostly. (p)artial signifies that one or more of the Physical Volumes this Logical Volume uses is missing from the system. (r)efresh signifies that one or more of the Physical Volumes this RAID Logical Volume uses had suffered a write error. The write error could be due to a temporary failure of that Physical Volume or an indication that it is failing. The device should be refreshed or replaced. (m)ismatches signifies that the RAID logical volume has portions of the array that are not coherent. Inconsistencies are discovered by initiating a <b>check</b> operation on a RAID logical volume. (The scrubbing operations, <b>check</b> and <b>repair</b>, can be performed on a RAID Logical Volume by means of the <b>lvchange</b> command.) (w)ritemostly signifies the devices in a RAID 1 logical volume that have been marked write-mostly.</p> <p>* Bit 10: s(k)ip activation: this volume is flagged to be skipped during activation.</p>
<b>lv_kernel_major</b>	KMaj	Actual major device number of the logical volume (-1 if inactive)
<b>lv_kernel_minor</b>	KMIN	Actual minor device number of the logical volume (-1 if inactive)
<b>lv_major</b>	Maj	The persistent major device number of the logical volume (-1 if not specified)
<b>lv_minor</b>	Min	The persistent minor device number of the logical volume (-1 if not specified)
<b>lv_name</b>	LV	The name of the logical volume
<b>lv_size</b>	LSize	The size of the logical volume
<b>lv_tags</b>	LV Tags	LVM tags attached to the logical volume
<b>lv_uuid</b>	LV UUID	The UUID of the logical volume.
<b>mirror_log</b>	Log	Device on which the mirror log resides
<b>modules</b>	Modules	Corresponding kernel device-mapper target necessary to use this logical volume

Argument	Header	Description
<b>move_pv</b>	Move	Source physical volume of a temporary logical volume created with the <b>pvmove</b> command
<b>origin</b>	Origin	The origin device of a snapshot volume
<b>* regionsize</b>	Region	The unit size of a mirrored logical volume
<b>* region_size</b>		
<b>seg_count</b>	#Seg	The number of segments in the logical volume
<b>seg_size</b>	SSize	The size of the segments in the logical volume
<b>seg_start</b>	Start	Offset of the segment in the logical volume
<b>seg_tags</b>	Seg Tags	LVM tags attached to the segments of the logical volume
<b>segtype</b>	Type	The segment type of a logical volume (for example: mirror, striped, linear)
<b>snap_percent</b>	Snap%	Current percentage of a snapshot volume that is in use
<b>stripes</b>	#Str	Number of stripes or mirrors in a logical volume
<b>* stripesize</b>	Stripe	Unit size of the stripe in a striped logical volume
<b>* stripe_size</b>		

The **lvs** command provides the following display by default. The default display is sorted by **vg\_name** and **lv\_name** within the volume group.

```
lvs
 LV VG Attr LSize Pool Origin Data% Meta% Move Log Cpy%Sync Convert
 origin VG owi-a-s--- 1.00g
 snap VG swi-a-s--- 100.00m origin 0.00
```

A common use of the **lvs** command is to append **devices** to the command to display the underlying devices that make up the logical volume. This example also specifies the **-a** option to display the internal volumes that are components of the logical volumes, such as RAID mirrors, enclosed in brackets. This example includes a RAID volume, a striped volume, and a thinly-pooled volume.

```
lvs -a -o +devices
 LV VG Attr LSize Pool Origin Data% Meta% Move Log Cpy%Sync Convert
 Devices
 raid1 VG rwi-a-r--- 1.00g
 raid1_rimage_0(0),raid1_rimage_1(0)
 [raid1_rimage_0] VG iwi-aor--- 1.00g
 /dev/sde1(7041)
```

[raid1_rimage_1] VG	iwi-aor---	1.00g	/dev/sdf1(7041)
[raid1_rmeta_0] VG	ewi-aor---	4.00m	/dev/sde1(7040)
[raid1_rmeta_1] VG	ewi-aor---	4.00m	/dev/sdf1(7040)
stripe1 VG	-wi-a----	99.95g	/dev/sde1(0),/dev/sdf1(0)
stripe1 VG	-wi-a----	99.95g	/dev/sdd1(0)
stripe1 VG	-wi-a----	99.95g	/dev/sdc1(0)
[lvol0_pmspare] rhel_host-083 ewi-----	4.00m		/dev/vda2(0)
pool00 rhel_host-083 twi-aotz--	<4.79g	72.90 54.69	
pool00_tdata(0)			
[pool00_tdata] rhel_host-083 Twi-ao----	<4.79g		/dev/vda2(1)
[pool00_tmeta] rhel_host-083 ewi-ao----	4.00m		/dev/vda2(1226)
root rhel_host-083 Vwi-aotz--	<4.79g	pool00 72.90	
swap rhel_host-083 -wi-ao----	820.00m		/dev/vda2(1227)

Using the **-v** argument with the **lvs** command adds the following fields to the default display:

**seg\_count**, **lv\_major**, **lv\_minor**, **lv\_kernel\_major**, **lv\_kernel\_minor**, **lv\_uuid**.

```
lvs -v
Finding all logical volumes
LV VG #Seg Attr LSize Maj Min KMaj KMin Origin Snap% Move Copy% Log Convert LV
UUID
lvol0 new_vg 1 owi-a- 52.00M -1 -1 253 3 LBy1Tz-sr23-OjsI-LT03-
nHLC-y8XW-EhCI78
newvgsnap1 new_vg 1 swi-a- 8.00M -1 -1 253 5 lvol0 0.20 1ye1OU-1clu-
o79k-20h2-ZGF0-qCJm-CfbSlx
```

You can use the **--segments** argument of the **lvs** command to display information with default columns that emphasize the segment information. When you use the **segments** argument, the **seg** prefix is optional. The **lvs --segments** command displays the following fields by default: **lv\_name**, **vg\_name**, **lv\_attr**, **stripes**, **segtype**, **seg\_size**. The default display is sorted by **vg\_name**, **lv\_name** within the volume group, and **seg\_start** within the logical volume. If the logical volumes were fragmented, the output from this command would show that.

```
lvs --segments
LV VG Attr #Str Type SSize
LogVol00 VolGroup00 -wi-ao 1 linear 36.62G
LogVol01 VolGroup00 -wi-ao 1 linear 512.00M
lv vg -wi-a- 1 linear 104.00M
lv vg -wi-a- 1 linear 104.00M
lv vg -wi-a- 1 linear 104.00M
lv vg -wi-a- 1 linear 88.00M
```

Using the **-v** argument with the **lvs --segments** command adds the following fields to the default display: **seg\_start**, **stripesize**, **chunksize**.

```
lvs -v --segments
Finding all logical volumes
LV VG Attr Start SSize #Str Type Stripe Chunk
lvol0 new_vg owi-a- 0 52.00M 1 linear 0 0
newvgsnap1 new_vg swi-a- 0 8.00M 1 linear 0 8.00K
```

The following example shows the default output of the **lvs** command on a system with one logical volume configured, followed by the default output of the **lvs** command with the **segments** argument specified.

```
lvs
LV VG Attr LSize Origin Snap% Move Log Copy%
lvol0 new_vg -wi-a- 52.00M
lvs --segments
LV VG Attr #Str Type SSize
lvol0 new_vg -wi-a- 1 linear 52.00M
```

### 117.3. SORTING LVM REPORTS

Normally the entire output of the **lvs**, **vgs**, or **pvs** command has to be generated and stored internally before it can be sorted and columns aligned correctly. You can specify the **--unbuffered** argument to display unsorted output as soon as it is generated.

To specify an alternative ordered list of columns to sort on, use the **-O** argument of any of the reporting commands. It is not necessary to include these fields within the output itself.

The following example shows the output of the **pvs** command that displays the physical volume name, size, and free space.

```
pvs -o pv_name,pv_size,pv_free
PV PSize PFree
/dev/sdb1 17.14G 17.14G
/dev/sdc1 17.14G 17.09G
/dev/sdd1 17.14G 17.14G
```

The following example shows the same output, sorted by the free space field.

```
pvs -o pv_name,pv_size,pv_free -O pv_free
PV PSize PFree
/dev/sdc1 17.14G 17.09G
/dev/sdd1 17.14G 17.14G
/dev/sdb1 17.14G 17.14G
```

The following example shows that you do not need to display the field on which you are sorting.

```
pvs -o pv_name,pv_size -O pv_free
PV PSize
/dev/sdc1 17.14G
/dev/sdd1 17.14G
/dev/sdb1 17.14G
```

To display a reverse sort, precede a field you specify after the **-O** argument with the **-** character.

```
pvs -o pv_name,pv_size,pv_free -O -pv_free
PV PSize PFree
/dev/sdd1 17.14G 17.14G
/dev/sdb1 17.14G 17.14G
/dev/sdc1 17.14G 17.09G
```

### 117.4. SPECIFYING THE UNITS FOR AN LVM REPORT DISPLAY

To specify the units for the LVM report display, use the **--units** argument of the report command. You can specify (b)ytes, (k)ilobytes, (m)egabytes, (g)igabytes, (t)erabytes, (e)xabytes, (p)etabytes, and

(h)uman-readable. The default display is human-readable. You can override the default by setting the **units** parameter in the **global** section of the **/etc/lvm/lvm.conf** file.

The following example specifies the output of the **pvs** command in megabytes rather than the default gigabytes.

```
pvs --units m
PV VG Fmt Attr PSize PFree
/dev/sda1 lvm2 -- 17555.40M 17555.40M
/dev/sdb1 new_vg lvm2 a- 17552.00M 17552.00M
/dev/sdc1 new_vg lvm2 a- 17552.00M 17500.00M
/dev/sdd1 new_vg lvm2 a- 17552.00M 17552.00M
```

By default, units are displayed in powers of 2 (multiples of 1024). You can specify that units be displayed in multiples of 1000 by capitalizing the unit specification (B, K, M, G, T, H).

The following command displays the output as a multiple of 1024, the default behavior.

```
pvs
PV VG Fmt Attr PSize PFree
/dev/sdb1 new_vg lvm2 a- 17.14G 17.14G
/dev/sdc1 new_vg lvm2 a- 17.14G 17.09G
/dev/sdd1 new_vg lvm2 a- 17.14G 17.14G
```

The following command displays the output as a multiple of 1000.

```
pvs --units G
PV VG Fmt Attr PSize PFree
/dev/sdb1 new_vg lvm2 a- 18.40G 18.40G
/dev/sdc1 new_vg lvm2 a- 18.40G 18.35G
/dev/sdd1 new_vg lvm2 a- 18.40G 18.40G
```

You can also specify (s)ectors (defined as 512 bytes) or custom units.

The following example displays the output of the **pvs** command as a number of sectors.

```
pvs --units s
PV VG Fmt Attr PSize PFree
/dev/sdb1 new_vg lvm2 a- 35946496S 35946496S
/dev/sdc1 new_vg lvm2 a- 35946496S 35840000S
/dev/sdd1 new_vg lvm2 a- 35946496S 35946496S
```

The following example displays the output of the **pvs** command in units of 4 MB.

```
pvs --units 4m
PV VG Fmt Attr PSize PFree
/dev/sdb1 new_vg lvm2 a- 4388.00U 4388.00U
/dev/sdc1 new_vg lvm2 a- 4388.00U 4375.00U
/dev/sdd1 new_vg lvm2 a- 4388.00U 4388.00U
```

## 117.5. DISPLAYING LVM COMMAND OUTPUT IN JSON FORMAT

You can use the **--reportformat** option of the LVM display commands to display the output in JSON format.

The following example shows the output of the **lvs** in standard default format.

```
lvs
 LV VG Attr LSize Pool Origin Data% Meta% Move Log Cpy%Sync Convert
my_raid my_vg Rwi-a-r--- 12.00m 100.00
root rhel_host-075 -wi-ao---- 6.67g
swap rhel_host-075 -wi-ao---- 820.00m
```

The following command shows the output of the same LVM configuration when you specify JSON format.

```
lvs --reportformat json
{
 "report": [
 {
 "lv": [
 {"lv_name": "my_raid", "vg_name": "my_vg", "lv_attr": "Rwi-a-r---", "lv_size": "12.00m", "pool_lv": "", "origin": "", "data_percent": "", "metadata_percent": "", "move_pv": "", "mirror_log": "", "copy_percent": "100.00", "convert_lv": ""},
 {"lv_name": "root", "vg_name": "rhel_host-075", "lv_attr": "-wi-ao----", "lv_size": "6.67g", "pool_lv": "", "origin": "", "data_percent": "", "metadata_percent": "", "move_pv": "", "mirror_log": "", "copy_percent": "", "convert_lv": ""},
 {"lv_name": "swap", "vg_name": "rhel_host-075", "lv_attr": "-wi-ao----", "lv_size": "820.00m", "pool_lv": "", "origin": "", "data_percent": "", "metadata_percent": "", "move_pv": "", "mirror_log": "", "copy_percent": "", "convert_lv": ""}
]
 }
]
}
```

You can also set the report format as a configuration option in the **/etc/lvm/lvm.conf** file, using the **output\_format** setting. The **--reportformat** setting of the command line, however, takes precedence over this setting.

## 117.6. DISPLAYING THE LVM COMMAND LOG

Both report-oriented and processing-oriented LVM commands can report the command log if this is enabled with the **log/report\_command\_log** configuration setting. You can determine the set of fields to display and to sort by for this report.

The following examples configures LVM to generate a complete log report for LVM commands. In this example, you can see that both logical volumes **lvol0** and **lvol1** were successfully processed, as was the volume group **VG** that contains the volumes.

```
lvmconfig --type full log/command_log_selection
command_log_selection="all"

lvs
Logical Volume
=====
LV LSize Cpy%Sync
lvol1 4.00m 100.00
```

```
lvol0 4.00m
```

```
Command Log
```

```
=====
```

Seq	LogType	Context	ObjType	ObjName	ObjGrp	Msg	Errno	RetCode
1	status	processing	lv	lvol0	vg	success	0	1
2	status	processing	lv	lvol1	vg	success	0	1
3	status	processing	vg		vg	success	0	1

```
lvchange -an vg/lvol1
```

```
Command Log
```

```
=====
```

Seq	LogType	Context	ObjType	ObjName	ObjGrp	Msg	Errno	RetCode
1	status	processing	lv	lvol1	vg	success	0	1
2	status	processing	vg		vg	success	0	1

For further information on configuring LVM reports and command logs, see the **lvmreport** man page.

## CHAPTER 118. CONFIGURING RAID LOGICAL VOLUMES

LVM supports RAID0/1/4/5/6/10.



### NOTE

RAID logical volumes are not cluster-aware. While RAID logical volumes can be created and activated exclusively on one machine, they cannot be activated simultaneously on more than one machine.

To create a RAID logical volume, you specify a raid type as the **--type** argument of the **lvcreate** command. [Table 118.1, “RAID Segment Types”](#) describes the possible RAID segment types. After you have created a RAID logical volume with LVM, you can activate, change, remove, display, and use the volume just as you would any other LVM logical volume.

**Table 118.1. RAID Segment Types**

Segment type	Description
<b>raid1</b>	RAID1 mirroring. This is the default value for the <b>--type</b> argument of the <b>lvcreate</b> command when you specify the <b>-m</b> but you do not specify striping.
<b>raid4</b>	RAID4 dedicated parity disk
<b>raid5</b>	Same as <b>raid5_ls</b>
<b>raid5_la</b>	<ul style="list-style-type: none"> <li>* RAID5 left asymmetric.</li> <li>* Rotating parity 0 with data continuation</li> </ul>
<b>raid5_ra</b>	<ul style="list-style-type: none"> <li>* RAID5 right asymmetric.</li> <li>* Rotating parity N with data continuation</li> </ul>
<b>raid5_ls</b>	<ul style="list-style-type: none"> <li>* RAID5 left symmetric.</li> <li>* Rotating parity 0 with data restart</li> </ul>
<b>raid5_rs</b>	<ul style="list-style-type: none"> <li>* RAID5 right symmetric.</li> <li>* Rotating parity N with data restart</li> </ul>
<b>raid6</b>	Same as <b>raid6_zr</b>
<b>raid6_zr</b>	<ul style="list-style-type: none"> <li>* RAID6 zero restart</li> <li>* Rotating parity zero (left-to-right) with data restart</li> </ul>
<b>raid6_nr</b>	<ul style="list-style-type: none"> <li>* RAID6 N restart</li> <li>* Rotating parity N (left-to-right) with data restart</li> </ul>

Segment type	Description
<b>raid6_nc</b>	<ul style="list-style-type: none"> <li>* RAID6 N continue</li> <li>* Rotating parity N (left-to-right) with data continuation</li> </ul>
<b>raid10</b>	<ul style="list-style-type: none"> <li>* Striped mirrors. This is the default value for the <b>--type</b> argument of the <b>lvcreate</b> command if you specify the <b>-m</b> and you specify a number of stripes that is greater than 1.</li> <li>* Striping of mirror sets</li> </ul>
<b>raid0/raid0_meta</b>	Striping. RAID0 spreads logical volume data across multiple data subvolumes in units of stripe size. This is used to increase performance. Logical volume data will be lost if any of the data subvolumes fail.

For most users, specifying one of the five available primary types (**raid1**, **raid4**, **raid5**, **raid6**, **raid10**) should be sufficient. For more information on the different algorithms used by RAID 5/6, see chapter four of the *Common RAID Disk Data Format Specification* at [http://www.snia.org/sites/default/files/SNIA\\_DDF\\_Technical\\_Position\\_v2.0.pdf](http://www.snia.org/sites/default/files/SNIA_DDF_Technical_Position_v2.0.pdf).

When you create a RAID logical volume, LVM creates a metadata subvolume that is one extent in size for every data or parity subvolume in the array. For example, creating a 2-way RAID1 array results in two metadata subvolumes (**lv\_rmeta\_0** and **lv\_rmeta\_1**) and two data subvolumes (**lv\_rimage\_0** and **lv\_rimage\_1**). Similarly, creating a 3-way stripe (plus 1 implicit parity device) RAID4 results in 4 metadata subvolumes (**lv\_rmeta\_0**, **lv\_rmeta\_1**, **lv\_rmeta\_2**, and **lv\_rmeta\_3**) and 4 data subvolumes (**lv\_rimage\_0**, **lv\_rimage\_1**, **lv\_rimage\_2**, and **lv\_rimage\_3**).



#### NOTE

You can generate commands to create logical volumes on RAID storage with the LVM RAID Calculator application. This application uses the information you input about your current or planned storage to generate these commands. The LVM RAID Calculator application can be found at <https://access.redhat.com/labs/lvmaidcalculator/>.

## 118.1. CREATING RAID LOGICAL VOLUMES

This section provides example commands that create different types of RAID logical volume.

You can create RAID1 arrays with different numbers of copies according to the value you specify for the **-m** argument. Similarly, you specify the number of stripes for a RAID 4/5/6 logical volume with the **-i** argument. You can also specify the stripe size with the **-l** argument.

The following command creates a 2-way RAID1 array named **my\_lv** in the volume group **my\_vg** that is one gigabyte in size.

```
lvcreate --type raid1 -m 1 -L 1G -n my_lv my_vg
```

The following command creates a RAID5 array (3 stripes + 1 implicit parity drive) named **my\_lv** in the volume group **my\_vg** that is one gigabyte in size. Note that you specify the number of stripes just as you do for an LVM striped volume; the correct number of parity drives is added automatically.

```
lvcreate --type raid5 -i 3 -L 1G -n my_lv my_vg
```

The following command creates a RAID6 array (3 stripes + 2 implicit parity drives) named **my\_lv** in the volume group **my\_vg** that is one gigabyte in size.

```
lvcreate --type raid6 -i 3 -L 1G -n my_lv my_vg
```

## 118.2. CREATING A RAID0 (STRIPED) LOGICAL VOLUME

A RAID0 logical volume spreads logical volume data across multiple data subvolumes in units of stripe size.

The format for the command to create a RAID0 volume is as follows.

```
lvcreate --type raid0[_meta] --stripes Stripes --stripesize StripeSize VolumeGroup
[PhysicalVolumePath ...]
```

Table 118.2. RAID0 Command Creation parameters

Parameter	Description
<b>--type raid0[_meta]</b>	Specifying <b>raid0</b> creates a RAID0 volume without metadata volumes. Specifying <b>raid0_meta</b> creates a RAID0 volume with metadata volumes. Because RAID0 is non-resilient, it does not have to store any mirrored data blocks as RAID1/10 or calculate and store any parity blocks as RAID4/5/6 do. Hence, it does not need metadata volumes to keep state about resynchronization progress of mirrored or parity blocks. Metadata volumes become mandatory on a conversion from RAID0 to RAID4/5/6/10, however, and specifying <b>raid0_meta</b> preallocates those metadata volumes to prevent a respective allocation failure.
<b>--stripes <i>Stripes</i></b>	Specifies the number of devices to spread the logical volume across.
<b>--stripesize <i>StripeSize</i></b>	Specifies the size of each stripe in kilobytes. This is the amount of data that is written to one device before moving to the next device.
<b><i>VolumeGroup</i></b>	Specifies the volume group to use.

Parameter	Description
<b>PhysicalVolumePath</b> ...	Specifies the devices to use. If this is not specified, LVM will choose the number of devices specified by the <i>Stripes</i> option, one for each stripe.

This example procedure creates an LVM RAID0 logical volume called **mylv** that stripes data across the disks at **/dev/sda1**, **/dev/sdb1**, and **/dev/sdc1**.

1. Label the disks you will use in the volume group as LVM physical volumes with the **pvcreate** command.



#### WARNING

This command destroys any data on **/dev/sda1**, **/dev/sdb1**, and **/dev/sdc1**.

```
pvcreate /dev/sda1 /dev/sdb1 /dev/sdc1
Physical volume "/dev/sda1" successfully created
Physical volume "/dev/sdb1" successfully created
Physical volume "/dev/sdc1" successfully created
```

2. Create the volume group **myvg**. The following command creates the volume group **myvg**.

```
vgcreate myvg /dev/sda1 /dev/sdb1 /dev/sdc1
Volume group "myvg" successfully created
```

You can use the **vgs** command to display the attributes of the new volume group.

```
vgs
VG #PV #LV #SN Attr VSize VFree
myvg 3 0 0 wz--n- 51.45G 51.45G
```

3. Create a RAID0 logical volume from the volume group you have created. The following command creates the RAID0 volume **mylv** from the volume group **myvg**. This example creates a logical volume that is 2 gigabytes in size, with three stripes and a stripe size of 4 kilobytes.

```
lvcreate --type raid0 -L 2G --stripes 3 --stripesize 4 -n mylv myvg
Rounding size 2.00 GiB (512 extents) up to stripe boundary size 2.00 GiB(513 extents).
Logical volume "mylv" created.
```

4. Create a file system on the RAID0 logical volume. The following command creates an **ext4** file system on the logical volume.

```
mkfs.ext4 /dev/myvg/mylv
```

```
mke2fs 1.44.3 (10-July-2018)
Creating filesystem with 525312 4k blocks and 131376 inodes
Filesystem UUID: 9d4c0704-6028-450a-8b0a-8875358c0511
Superblock backups stored on blocks:
 32768, 98304, 163840, 229376, 294912
```

```
Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
```

The following commands mount the logical volume and report the file system disk space usage.

```
mount /dev/myvg/mylv /mnt
df
Filesystem 1K-blocks Used Available Use% Mounted on
/dev/mapper/myvg-mylv 2002684 6168 1875072 1% /mnt
```

## 118.3. CONTROLLING THE RATE AT WHICH RAID VOLUMES ARE INITIALIZED

When you create RAID10 logical volumes, the background I/O required to initialize the logical volumes with a **sync** operation can crowd out other I/O operations to LVM devices, such as updates to volume group metadata, particularly when you are creating many RAID logical volumes. This can cause the other LVM operations to slow down.

You can control the rate at which a RAID logical volume is initialized by implementing recovery throttling. You control the rate at which **sync** operations are performed by setting the minimum and maximum I/O rate for those operations with the **--minrecoveryrate** and **--maxrecoveryrate** options of the **lvcreate** command. You specify these options as follows.

- **--maxrecoveryrate Rate[bBsSkMmGg]**

Sets the maximum recovery rate for a RAID logical volume so that it will not crowd out nominal I/O operations. The *Rate* is specified as an amount per second for each device in the array. If no suffix is given, then kiB/sec/device is assumed. Setting the recovery rate to 0 means it will be unbounded.

- **--minrecoveryrate Rate[bBsSkMmGg]**

Sets the minimum recovery rate for a RAID logical volume to ensure that I/O for **sync** operations achieves a minimum throughput, even when heavy nominal I/O is present. The *Rate* is specified as an amount per second for each device in the array. If no suffix is given, then kiB/sec/device is assumed.

The following command creates a 2-way RAID10 array with 3 stripes that is 10 gigabytes in size with a maximum recovery rate of 128 kiB/sec/device. The array is named **my\_lv** and is in the volume group **my\_vg**.

```
lvcreate --type raid10 -i 2 -m 1 -L 10G --maxrecoveryrate 128 -n my_lv my_vg
```

You can also specify minimum and maximum recovery rates for a RAID scrubbing operation.

## 118.4. CONVERTING A LINEAR DEVICE TO A RAID DEVICE

You can convert an existing linear logical volume to a RAID device by using the **--type** argument of the **lvconvert** command.

The following command converts the linear logical volume **my\_lv** in volume group **my\_vg** to a 2-way RAID1 array.

```
lvconvert --type raid1 -m 1 my_vg/my_lv
```

Since RAID logical volumes are composed of metadata and data subvolume pairs, when you convert a linear device to a RAID1 array, a new metadata subvolume is created and associated with the original logical volume on (one of) the same physical volumes that the linear volume is on. The additional images are added in metadata/data subvolume pairs. For example, if the original device is as follows:

```
lvs -a -o name,copy_percent,devices my_vg
 LV Copy% Devices
 my_lv /dev/sde1(0)
```

After conversion to a 2-way RAID1 array the device contains the following data and metadata subvolume pairs:

```
lvconvert --type raid1 -m 1 my_vg/my_lv
lvs -a -o name,copy_percent,devices my_vg
 LV Copy% Devices
 my_lv 6.25 my_lv_rimage_0(0),my_lv_rimage_1(0)
 [my_lv_rimage_0] /dev/sde1(0)
 [my_lv_rimage_1] /dev/sdf1(1)
 [my_lv_rmeta_0] /dev/sde1(256)
 [my_lv_rmeta_1] /dev/sdf1(0)
```

If the metadata image that pairs with the original logical volume cannot be placed on the same physical volume, the **lvconvert** will fail.

## 118.5. CONVERTING AN LVM RAID1 LOGICAL VOLUME TO AN LVM LINEAR LOGICAL VOLUME

You can convert an existing RAID1 LVM logical volume to an LVM linear logical volume with the **lvconvert** command by specifying the **-m0** argument. This removes all the RAID data subvolumes and all the RAID metadata subvolumes that make up the RAID array, leaving the top-level RAID1 image as the linear logical volume.

The following example displays an existing LVM RAID1 logical volume.

```
lvs -a -o name,copy_percent,devices my_vg
 LV Copy% Devices
 my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0)
 [my_lv_rimage_0] /dev/sde1(1)
 [my_lv_rimage_1] /dev/sdf1(1)
 [my_lv_rmeta_0] /dev/sde1(0)
 [my_lv_rmeta_1] /dev/sdf1(0)
```

The following command converts the LVM RAID1 logical volume **my\_vg/my\_lv** to an LVM linear device.

```
lvconvert -m0 my_vg/my_lv
lvs -a -o name,copy_percent,devices my_vg
```

```
LV Copy% Devices
my_lv /dev/sde1(1)
```

When you convert an LVM RAID1 logical volume to an LVM linear volume, you can specify which physical volumes to remove. The following example shows the layout of an LVM RAID1 logical volume made up of two images: **/dev/sda1** and **/dev/sdb1**. In this example, the **lvconvert** command specifies that you want to remove **/dev/sda1**, leaving **/dev/sdb1** as the physical volume that makes up the linear device.

```
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0)
[my_lv_rimage_0] /dev/sda1(1)
[my_lv_rimage_1] /dev/sdb1(1)
[my_lv_rmeta_0] /dev/sda1(0)
[my_lv_rmeta_1] /dev/sdb1(0)
lvconvert -m0 my_vg/my_lv /dev/sda1
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv /dev/sdb1(1)
```

## 118.6. CONVERTING A MIRRORED LVM DEVICE TO A RAID1 DEVICE

You can convert an existing mirrored LVM device with a segment type of **mirror** to a RAID1 LVM device with the **lvconvert** command by specifying the **--type raid1** argument. This renames the mirror subvolumes (**mimage**) to RAID subvolumes (**rimage**). In addition, the mirror log is removed and metadata subvolumes (**rmeta**) are created for the data subvolumes on the same physical volumes as the corresponding data subvolumes.

The following example shows the layout of a mirrored logical volume **my\_vg/my\_lv**.

```
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 15.20 my_lv_mimage_0(0),my_lv_mimage_1(0)
[my_lv_mimage_0] /dev/sde1(0)
[my_lv_mimage_1] /dev/sdf1(0)
[my_lv_mlog] /dev/sdd1(0)
```

The following command converts the mirrored logical volume **my\_vg/my\_lv** to a RAID1 logical volume.

```
lvconvert --type raid1 my_vg/my_lv
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0)
[my_lv_rimage_0] /dev/sde1(0)
[my_lv_rimage_1] /dev/sdf1(0)
[my_lv_rmeta_0] /dev/sde1(125)
[my_lv_rmeta_1] /dev/sdf1(125)
```

## 118.7. RESIZING A RAID LOGICAL VOLUME

You can resize a RAID logical volume in the following ways:

- You can increase the size of a RAID logical volume of any type with the **lvresize** or **lvextend** command. This does not change the number of RAID images. For striped RAID logical volumes the same stripe rounding constraints apply as when you create a striped RAID logical volume.
- You can reduce the size of a RAID logical volume of any type with the **lvresize** or **lvreduce** command. This does not change the number of RAID images. As with the **lvextend** command, the same stripe rounding constraints apply as when you create a striped RAID logical volume.
- You can change the number of stripes on a striped RAID logical volume (**raid4/5/6/10**) with the **-stripes N** parameter of the **lvconvert** command. This increases or reduces the size of the RAID logical volume by the capacity of the stripes added or removed. Note that **raid10** volumes are capable only of adding stripes. This capability is part of the RAID *reshaping* feature that allows you to change attributes of a RAID logical volume while keeping the same RAID level. For information on RAID reshaping and examples of using the **lvconvert** command to reshape a RAID logical volume, see the **lvmraid(7)** man page.

## 118.8. CHANGING THE NUMBER OF IMAGES IN AN EXISTING RAID1 DEVICE

You can change the number of images in an existing RAID1 array just as you can change the number of images in the earlier implementation of LVM mirroring. Use the **lvconvert** command to specify the number of additional metadata/data subvolume pairs to add or remove.

When you add images to a RAID1 device with the **lvconvert** command, you can specify the total number of images for the resulting device, or you can specify how many images to add to the device. You can also optionally specify on which physical volumes the new metadata/data image pairs will reside.

Metadata subvolumes (named **rmeta**) always exist on the same physical devices as their data subvolume counterparts (**rimage**). The metadata/data subvolume pairs will not be created on the same physical volumes as those from another metadata/data subvolume pair in the RAID array (unless you specify **--alloc anywhere**).

The format for the command to add images to a RAID1 volume is as follows:

```
lvconvert -m new_absolute_count vg/lv [removable_PVs]
lvconvert -m +num_additional_images vg/lv [removable_PVs]
```

For example, the following command displays the LVM device **my\_vg/my\_lv**, which is a 2-way RAID1 array:

```
lvs -a -o name,copy_percent,devices my_vg
 LV Copy% Devices
 my_lv 6.25 my_lv_rimage_0(0),my_lv_rimage_1(0)
 [my_lv_rimage_0] /dev/sde1(0)
 [my_lv_rimage_1] /dev/sdf1(1)
 [my_lv_rmeta_0] /dev/sde1(256)
 [my_lv_rmeta_1] /dev/sdf1(0)
```

The following command converts the 2-way RAID1 device **my\_vg/my\_lv** to a 3-way RAID1 device:

```
lvconvert -m 2 my_vg/my_lv
lvs -a -o name,copy_percent,devices my_vg
 LV Copy% Devices
 my_lv 6.25 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
 [my_lv_rimage_0] /dev/sde1(0)
```

```
[my_lv_rimage_1] /dev/sdf1(1)
[my_lv_rimage_2] /dev/sdg1(1)
[my_lv_rmeta_0] /dev/sde1(256)
[my_lv_rmeta_1] /dev/sdf1(0)
[my_lv_rmeta_2] /dev/sdg1(0)
```

When you add an image to a RAID1 array, you can specify which physical volumes to use for the image. The following command converts the 2-way RAID1 device **my\_vg/my\_lv** to a 3-way RAID1 device, specifying that the physical volume **/dev/sdd1** be used for the array:

```
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 56.00 my_lv_rimage_0(0),my_lv_rimage_1(0)
[my_lv_rimage_0] /dev/sda1(1)
[my_lv_rimage_1] /dev/sdb1(1)
[my_lv_rmeta_0] /dev/sda1(0)
[my_lv_rmeta_1] /dev/sdb1(0)
lvconvert -m 2 my_vg/my_lv /dev/sdd1
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 28.00 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
[my_lv_rimage_0] /dev/sda1(1)
[my_lv_rimage_1] /dev/sdb1(1)
[my_lv_rimage_2] /dev/sdd1(1)
[my_lv_rmeta_0] /dev/sda1(0)
[my_lv_rmeta_1] /dev/sdb1(0)
[my_lv_rmeta_2] /dev/sdd1(0)
```

To remove images from a RAID1 array, use the following command. When you remove images from a RAID1 device with the **lvconvert** command, you can specify the total number of images for the resulting device, or you can specify how many images to remove from the device. You can also optionally specify the physical volumes from which to remove the device.

```
lvconvert -m new_absolute_count vg/lv [removable_PVs]
lvconvert -m -num_fewer_images vg/lv [removable_PVs]
```

Additionally, when an image and its associated metadata subvolume volume are removed, any higher-numbered images will be shifted down to fill the slot. If you remove **lv\_rimage\_1** from a 3-way RAID1 array that consists of **lv\_rimage\_0**, **lv\_rimage\_1**, and **lv\_rimage\_2**, this results in a RAID1 array that consists of **lv\_rimage\_0** and **lv\_rimage\_1**. The subvolume **lv\_rimage\_2** will be renamed and take over the empty slot, becoming **lv\_rimage\_1**.

The following example shows the layout of a 3-way RAID1 logical volume **my\_vg/my\_lv**.

```
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
[my_lv_rimage_0] /dev/sde1(1)
[my_lv_rimage_1] /dev/sdf1(1)
[my_lv_rimage_2] /dev/sdg1(1)
[my_lv_rmeta_0] /dev/sde1(0)
[my_lv_rmeta_1] /dev/sdf1(0)
[my_lv_rmeta_2] /dev/sdg1(0)
```

The following command converts the 3-way RAID1 logical volume into a 2-way RAID1 logical volume.

```
lvconvert -m1 my_vg/my_lv
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0)
[my_lv_rimage_0] /dev/sde1(1)
[my_lv_rimage_1] /dev/sdf1(1)
[my_lv_rmeta_0] /dev/sde1(0)
[my_lv_rmeta_1] /dev/sdf1(0)
```

The following command converts the 3-way RAID1 logical volume into a 2-way RAID1 logical volume, specifying the physical volume that contains the image to remove as **/dev/sde1**.

```
lvconvert -m1 my_vg/my_lv /dev/sde1
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0)
[my_lv_rimage_0] /dev/sdf1(1)
[my_lv_rimage_1] /dev/sdg1(1)
[my_lv_rmeta_0] /dev/sdf1(0)
[my_lv_rmeta_1] /dev/sdg1(0)
```

## 118.9. SPLITTING OFF A RAID IMAGE AS A SEPARATE LOGICAL VOLUME

You can split off an image of a RAID logical volume to form a new logical volume.

The format of the command to split off a RAID image is as follows:

```
lvconvert --splitmirrors count -n splitname vg/lv [removable_PVs]
```

Just as when you are removing a RAID image from an existing RAID1 logical volume, when you remove a RAID data subvolume (and its associated metadata subvolume) from the middle of the device any higher numbered images will be shifted down to fill the slot. The index numbers on the logical volumes that make up a RAID array will thus be an unbroken sequence of integers.



### NOTE

You cannot split off a RAID image if the RAID1 array is not yet in sync.

The following example splits a 2-way RAID1 logical volume, **my\_lv**, into two linear logical volumes, **my\_lv** and **new**.

```
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 12.00 my_lv_rimage_0(0),my_lv_rimage_1(0)
[my_lv_rimage_0] /dev/sde1(1)
[my_lv_rimage_1] /dev/sdf1(1)
[my_lv_rmeta_0] /dev/sde1(0)
[my_lv_rmeta_1] /dev/sdf1(0)
lvconvert --splitmirror 1 -n new my_vg/my_lv
lvs -a -o name,copy_percent,devices my_vg
```

```
LV Copy% Devices
my_lv /dev/sde1(1)
new /dev/sdf1(1)
```

The following example splits a 3-way RAID1 logical volume, **my\_lv**, into a 2-way RAID1 logical volume, **my\_lv**, and a linear logical volume, **new**

```
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
[my_lv_rimage_0] /dev/sde1(1)
[my_lv_rimage_1] /dev/sdf1(1)
[my_lv_rimage_2] /dev/sdg1(1)
[my_lv_rmeta_0] /dev/sde1(0)
[my_lv_rmeta_1] /dev/sdf1(0)
[my_lv_rmeta_2] /dev/sdg1(0)
lvconvert --splitmirror 1 -n new my_vg/my_lv
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0)
[my_lv_rimage_0] /dev/sde1(1)
[my_lv_rimage_1] /dev/sdf1(1)
[my_lv_rmeta_0] /dev/sde1(0)
[my_lv_rmeta_1] /dev/sdf1(0)
new /dev/sdg1(1)
```

## 118.10. SPLITTING AND MERGING A RAID IMAGE

You can temporarily split off an image of a RAID1 array for read-only use while keeping track of any changes by using the **--trackchanges** argument in conjunction with the **--splitmirrors** argument of the **lvconvert** command. This allows you to merge the image back into the array at a later time while resyncing only those portions of the array that have changed since the image was split.

The format for the **lvconvert** command to split off a RAID image is as follows.

```
lvconvert --splitmirrors count --trackchanges vg/lv [removable_PVs]
```

When you split off a RAID image with the **--trackchanges** argument, you can specify which image to split but you cannot change the name of the volume being split. In addition, the resulting volumes have the following constraints.

- The new volume you create is read-only.
- You cannot resize the new volume.
- You cannot rename the remaining array.
- You cannot resize the remaining array.
- You can activate the new volume and the remaining array independently.

You can merge an image that was split off with the **--trackchanges** argument specified by executing a subsequent **lvconvert** command with the **--merge** argument. When you merge the image, only the portions of the array that have changed since the image was split are resynced.

The format for the **lvconvert** command to merge a RAID image is as follows.

```
lvconvert --merge raid_image
```

The following example creates a RAID1 logical volume and then splits off an image from that volume while tracking changes to the remaining array.

```
lvcreate --type raid1 -m 2 -L 1G -n my_lv my_vg
Logical volume "my_lv" created
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
[my_lv_rimage_0] /dev/sdb1(1)
[my_lv_rimage_1] /dev/sdc1(1)
[my_lv_rimage_2] /dev/sdd1(1)
[my_lv_rmeta_0] /dev/sdb1(0)
[my_lv_rmeta_1] /dev/sdc1(0)
[my_lv_rmeta_2] /dev/sdd1(0)
lvconvert --splitmirrors 1 --trackchanges my_vg/my_lv
my_lv_rimage_2 split from my_lv for read-only purposes.
Use 'lvconvert --merge my_vg/my_lv_rimage_2' to merge back into my_lv
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
[my_lv_rimage_0] /dev/sdb1(1)
[my_lv_rimage_1] /dev/sdc1(1)
my_lv_rimage_2 /dev/sdd1(1)
[my_lv_rmeta_0] /dev/sdb1(0)
[my_lv_rmeta_1] /dev/sdc1(0)
[my_lv_rmeta_2] /dev/sdd1(0)
```

The following example splits off an image from a RAID1 volume while tracking changes to the remaining array, then merges the volume back into the array.

```
lvconvert --splitmirrors 1 --trackchanges my_vg/my_lv
lv_rimage_1 split from my_lv for read-only purposes.
Use 'lvconvert --merge my_vg/my_lv_rimage_1' to merge back into my_lv
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0)
[my_lv_rimage_0] /dev/sdc1(1)
my_lv_rimage_1 /dev/sdd1(1)
[my_lv_rmeta_0] /dev/sdc1(0)
[my_lv_rmeta_1] /dev/sdd1(0)
lvconvert --merge my_vg/my_lv_rimage_1
my_vg/my_lv_rimage_1 successfully merged back into my_vg/my_lv
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0)
[my_lv_rimage_0] /dev/sdc1(1)
[my_lv_rimage_1] /dev/sdd1(1)
[my_lv_rmeta_0] /dev/sdc1(0)
[my_lv_rmeta_1] /dev/sdd1(0)
```

## 118.11. SETTING A RAID FAULT POLICY

LVM RAID handles device failures in an automatic fashion based on the preferences defined by the **raid\_fault\_policy** field in the **lvm.conf** file.

- If the **raid\_fault\_policy** field is set to **allocate**, the system will attempt to replace the failed device with a spare device from the volume group. If there is no available spare device, this will be reported to the system log.
- If the **raid\_fault\_policy** field is set to **warn**, the system will produce a warning and the log will indicate that a device has failed. This allows the user to determine the course of action to take.

As long as there are enough devices remaining to support usability, the RAID logical volume will continue to operate.

### 118.11.1. The allocate RAID Fault Policy

In the following example, the **raid\_fault\_policy** field has been set to **allocate** in the **lvm.conf** file. The RAID logical volume is laid out as follows.

```
lvs -a -o name,copy_percent,devices my_vg
 LV Copy% Devices
 my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
 [my_lv_rimage_0] /dev/sde1(1)
 [my_lv_rimage_1] /dev/sdf1(1)
 [my_lv_rimage_2] /dev/sdg1(1)
 [my_lv_rmeta_0] /dev/sde1(0)
 [my_lv_rmeta_1] /dev/sdf1(0)
 [my_lv_rmeta_2] /dev/sdg1(0)
```

If the **/dev/sde** device fails, the system log will display error messages.

```
grep lvm /var/log/messages
Jan 17 15:57:18 bp-01 lvm[8599]: Device #0 of raid1 array, my_vg-my_lv, has failed.
Jan 17 15:57:18 bp-01 lvm[8599]: /dev/sde1: read failed after 0 of 2048 at
250994294784: Input/output error
Jan 17 15:57:18 bp-01 lvm[8599]: /dev/sde1: read failed after 0 of 2048 at
250994376704: Input/output error
Jan 17 15:57:18 bp-01 lvm[8599]: /dev/sde1: read failed after 0 of 2048 at 0:
Input/output error
Jan 17 15:57:18 bp-01 lvm[8599]: /dev/sde1: read failed after 0 of 2048 at
4096: Input/output error
Jan 17 15:57:19 bp-01 lvm[8599]: Couldn't find device with uuid
3lugiV-3eSP-AFAR-sdrP-H20O-wM2M-qdMANy.
Jan 17 15:57:27 bp-01 lvm[8599]: raid1 array, my_vg-my_lv, is not in-sync.
Jan 17 15:57:36 bp-01 lvm[8599]: raid1 array, my_vg-my_lv, is now in-sync.
```

Since the **raid\_fault\_policy** field has been set to **allocate**, the failed device is replaced with a new device from the volume group.

```
lvs -a -o name,copy_percent,devices vg
Couldn't find device with uuid 3lugiV-3eSP-AFAR-sdrP-H20O-wM2M-qdMANy.
 LV Copy% Devices
 lv 100.00 lv_rimage_0(0),lv_rimage_1(0),lv_rimage_2(0)
 [lv_rimage_0] /dev/sdh1(1)
```

```
[lv_rimage_1] /dev/sdf1(1)
[lv_rimage_2] /dev/sdg1(1)
[lv_rmeta_0] /dev/sdh1(0)
[lv_rmeta_1] /dev/sdf1(0)
[lv_rmeta_2] /dev/sdg1(0)
```

Note that even though the failed device has been replaced, the display still indicates that LVM could not find the failed device. This is because, although the failed device has been removed from the RAID logical volume, the failed device has not yet been removed from the volume group. To remove the failed device from the volume group, you can execute **vgreduce --removemissing VG**.

If the **raid\_fault\_policy** has been set to **allocate** but there are no spare devices, the allocation will fail, leaving the logical volume as it is. If the allocation fails, you have the option of fixing the drive, then initiating recovery of the failed device with the **--refresh** option of the **lvchange** command. Alternately, you can replace the failed device.

### 118.11.2. The warn RAID Fault Policy

In the following example, the **raid\_fault\_policy** field has been set to **warn** in the **lvm.conf** file. The RAID logical volume is laid out as follows.

```
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
[my_lv_rimage_0] /dev/sdh1(1)
[my_lv_rimage_1] /dev/sdf1(1)
[my_lv_rimage_2] /dev/sdg1(1)
[my_lv_rmeta_0] /dev/sdh1(0)
[my_lv_rmeta_1] /dev/sdf1(0)
[my_lv_rmeta_2] /dev/sdg1(0)
```

If the **/dev/sdh** device fails, the system log will display error messages. In this case, however, LVM will not automatically attempt to repair the RAID device by replacing one of the images. Instead, if the device has failed you can replace the device with the **--repair** argument of the **lvconvert** command.

## 118.12. REPLACING A RAID DEVICE IN A LOGICAL VOLUME

You can replace a RAID device in a logical volume with the **lvconvert** command.

- If there has been no failure on a RAID device, use the **--replace** argument of the **lvconvert** command to replace the device.
- If the RAID device has failed, use the **--repair** argument of the **lvconvert** command to replace the failed device.

### 118.12.1. Replacing a RAID device that has not failed

To replace a RAID device in a logical volume, use the **--replace** argument of the **lvconvert** command. Note that this command will not work if the RAID device has failed.

The format for the **lvconvert --replace** command is as follows.

```
lvconvert --replace dev_to_remove vg/lv [possible_replacements]
```

The following example creates a RAID1 logical volume and then replaces a device in that volume.

```
lvcreate --type raid1 -m 2 -L 1G -n my_lv my_vg
Logical volume "my_lv" created
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
[my_lv_rimage_0] /dev/sdb1(1)
[my_lv_rimage_1] /dev/sdb2(1)
[my_lv_rimage_2] /dev/sdc1(1)
[my_lv_rmeta_0] /dev/sdb1(0)
[my_lv_rmeta_1] /dev/sdb2(0)
[my_lv_rmeta_2] /dev/sdc1(0)
lvconvert --replace /dev/sdb2 my_vg/my_lv
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 37.50 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
[my_lv_rimage_0] /dev/sdb1(1)
[my_lv_rimage_1] /dev/sdc2(1)
[my_lv_rimage_2] /dev/sdc1(1)
[my_lv_rmeta_0] /dev/sdb1(0)
[my_lv_rmeta_1] /dev/sdc2(0)
[my_lv_rmeta_2] /dev/sdc1(0)
```

The following example creates a RAID1 logical volume and then replaces a device in that volume, specifying which physical volume to use for the replacement.

```
lvcreate --type raid1 -m 1 -L 100 -n my_lv my_vg
Logical volume "my_lv" created
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0)
[my_lv_rimage_0] /dev/sda1(1)
[my_lv_rimage_1] /dev/sdb1(1)
[my_lv_rmeta_0] /dev/sda1(0)
[my_lv_rmeta_1] /dev/sdb1(0)
pvs
PV VG Fmt Attr PSize PFree
/dev/sda1 my_vg lvm2 a-- 1020.00m 916.00m
/dev/sdb1 my_vg lvm2 a-- 1020.00m 916.00m
/dev/sdc1 my_vg lvm2 a-- 1020.00m 1020.00m
/dev/sdd1 my_vg lvm2 a-- 1020.00m 1020.00m
lvconvert --replace /dev/sdb1 my_vg/my_lv /dev/sdd1
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 28.00 my_lv_rimage_0(0),my_lv_rimage_1(0)
[my_lv_rimage_0] /dev/sda1(1)
[my_lv_rimage_1] /dev/sdd1(1)
[my_lv_rmeta_0] /dev/sda1(0)
[my_lv_rmeta_1] /dev/sdd1(0)
```

You can replace more than one RAID device at a time by specifying multiple **replace** arguments, as in the following example.

```
lvcreate --type raid1 -m 2 -L 100 -n my_lv my_vg
```

```

Logical volume "my_lv" created
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
[my_lv_rimage_0] /dev/sda1(1)
[my_lv_rimage_1] /dev/sdb1(1)
[my_lv_rimage_2] /dev/sdc1(1)
[my_lv_rmeta_0] /dev/sda1(0)
[my_lv_rmeta_1] /dev/sdb1(0)
[my_lv_rmeta_2] /dev/sdc1(0)
lvconvert --replace /dev/sdb1 --replace /dev/sdc1 my_vg/my_lv
lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv 60.00 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
[my_lv_rimage_0] /dev/sda1(1)
[my_lv_rimage_1] /dev/sdd1(1)
[my_lv_rimage_2] /dev/sde1(1)
[my_lv_rmeta_0] /dev/sda1(0)
[my_lv_rmeta_1] /dev/sdd1(0)
[my_lv_rmeta_2] /dev/sde1(0)

```

### 118.12.2. Replacing a failed RAID device in a logical volume

RAID is not like traditional LVM mirroring. LVM mirroring required failed devices to be removed or the mirrored logical volume would hang. RAID arrays can keep on running with failed devices. In fact, for RAID types other than RAID1, removing a device would mean converting to a lower level RAID (for example, from RAID6 to RAID5, or from RAID4 or RAID5 to RAID0). Therefore, rather than removing a failed device unconditionally and potentially allocating a replacement, LVM allows you to replace a failed device in a RAID volume in a one-step solution by using the **--repair** argument of the **lvconvert** command.

In the following example, a RAID logical volume is laid out as follows.

```

lvs -a -o name,copy_percent,devices my_vg
LV Cpy%Sync Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
[my_lv_rimage_0] /dev/sde1(1)
[my_lv_rimage_1] /dev/sdc1(1)
[my_lv_rimage_2] /dev/sdd1(1)
[my_lv_rmeta_0] /dev/sde1(0)
[my_lv_rmeta_1] /dev/sdc1(0)
[my_lv_rmeta_2] /dev/sdd1(0)

```

If the **/dev/sdc** device fails, the output of the **lvs** command is as follows.

```

lvs -a -o name,copy_percent,devices my_vg
/dev/sdc: open failed: No such device or address
Couldn't find device with uuid A4kRI2-vIzA-uyCb-cci7-bOod-H5tX-lzH4Ee.
WARNING: Couldn't find all devices for LV my_vg/my_lv_rimage_1 while checking used and
assumed devices.
WARNING: Couldn't find all devices for LV my_vg/my_lv_rmeta_1 while checking used and
assumed devices.
LV Cpy%Sync Devices
my_lv 100.00 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
[my_lv_rimage_0] /dev/sde1(1)

```

```
[my_lv_rimage_1] [unknown](1)
[my_lv_rimage_2] /dev/sdd1(1)
[my_lv_rmeta_0] /dev/sde1(0)
[my_lv_rmeta_1] [unknown](0)
[my_lv_rmeta_2] /dev/sdd1(0)
```

Use the following commands to replace the failed device and display the logical volume.

```
lvconvert --repair my_vg/my_lv
/dev/sdc: open failed: No such device or address
Couldn't find device with uuid A4kRI2-vlxA-uyCb-cci7-bOod-H5tX-lzH4Ee.
WARNING: Couldn't find all devices for LV my_vg/my_lv_rimage_1 while checking used and
assumed devices.
WARNING: Couldn't find all devices for LV my_vg/my_lv_rmeta_1 while checking used and
assumed devices.
Attempt to replace failed RAID images (requires full device resync)? [y/n]: y
Faulty devices in my_vg/my_lv successfully replaced.

lvstatus -a -o name,copy_percent,devices my_vg
/dev/sdc: open failed: No such device or address
/dev/sdc1: open failed: No such device or address
Couldn't find device with uuid A4kRI2-vlxA-uyCb-cci7-bOod-H5tX-lzH4Ee.

LV Cpy%Sync Devices
my_lv 43.79 my_lv_rimage_0(0),my_lv_rimage_1(0),my_lv_rimage_2(0)
[my_lv_rimage_0] /dev/sde1(1)
[my_lv_rimage_1] /dev/sdb1(1)
[my_lv_rimage_2] /dev/sdd1(1)
[my_lv_rmeta_0] /dev/sde1(0)
[my_lv_rmeta_1] /dev/sdb1(0)
[my_lv_rmeta_2] /dev/sdd1(0)
```

Note that even though the failed device has been replaced, the display still indicates that LVM could not find the failed device. This is because, although the failed device has been removed from the RAID logical volume, the failed device has not yet been removed from the volume group. To remove the failed device from the volume group, you can execute **vgreduce --removemissing VG**.

If the device failure is a transient failure or you are able to repair the device that failed, you can initiate recovery of the failed device with the **--refresh** option of the **lvchange** command.

The following command refreshes a logical volume.

```
lvchange --refresh my_vg/my_lv
```

## 118.13. CHECKING DATA COHERENCY IN A RAID LOGICAL VOLUME (RAID SCRUBBING)

LVM provides scrubbing support for RAID logical volumes. RAID scrubbing is the process of reading all the data and parity blocks in an array and checking to see whether they are coherent.

You initiate a RAID scrubbing operation with the **--syncaction** option of the **lvchange** command. You specify either a **check** or **repair** operation. A **check** operation goes over the array and records the number of discrepancies in the array but does not repair them. A **repair** operation corrects the discrepancies as it finds them.

The format of the command to scrub a RAID logical volume is as follows:

```
lvchange --syncaction {check|repair} vg/raid_lv
```



### NOTE

The **lvchange --syncaction repair vg/raid\_lv** operation does not perform the same function as the **lvconvert --repair vg/raid\_lv** operation. The **lvchange --syncaction repair** operation initiates a background synchronization operation on the array, while the **lvconvert --repair** operation is designed to repair/replace failed devices in a mirror or RAID logical volume.

In support of the RAID scrubbing operation, the **lvs** command supports two new printable fields: **raid\_sync\_action** and **raid\_mismatch\_count**. These fields are not printed by default. To display these fields you specify them with the **-o** parameter of the **lvs**, as follows.

```
lvs -o +raid_sync_action,raid_mismatch_count vg/lv
```

The **raid\_sync\_action** field displays the current synchronization operation that the raid volume is performing. It can be one of the following values:

- **idle**: All sync operations complete (doing nothing)
- **resync**: Initializing an array or recovering after a machine failure
- **recover**: Replacing a device in the array
- **check**: Looking for array inconsistencies
- **repair**: Looking for and repairing inconsistencies

The **raid\_mismatch\_count** field displays the number of discrepancies found during a **check** operation.

The **Cpy%Sync** field of the **lvs** command now prints the progress of any of the **raid\_sync\_action** operations, including **check** and **repair**.

The **lv\_attr** field of the **lvs** command output now provides additional indicators in support of the RAID scrubbing operation. Bit 9 of this field displays the health of the logical volume, and it now supports the following indicators.

- **(m)ismatches** indicates that there are discrepancies in a RAID logical volume. This character is shown after a scrubbing operation has detected that portions of the RAID are not coherent.
- **(r)efresh** indicates that a device in a RAID array has suffered a failure and the kernel regards it as failed, even though LVM can read the device label and considers the device to be operational. The logical volume should be **(r)efreshed** to notify the kernel that the device is now available, or the device should be **(r)eplaced** if it is suspected of having failed.

When you perform a RAID scrubbing operation, the background I/O required by the **sync** operations can crowd out other I/O operations to LVM devices, such as updates to volume group metadata. This can cause the other LVM operations to slow down. You can control the rate at which the RAID logical volume is scrubbed by implementing recovery throttling.

You control the rate at which **sync** operations are performed by setting the minimum and maximum I/O rate for those operations with the **--minrecoveryrate** and **--maxrecoveryrate** options of the **lvchange** command. You specify these options as follows.

- **--maxrecoveryrate *Rate*[**bBsSkMmG**]**

Sets the maximum recovery rate for a RAID logical volume so that it will not crowd out nominal I/O operations. The *Rate* is specified as an amount per second for each device in the array. If no suffix is given, then kB/sec/device is assumed. Setting the recovery rate to 0 means it will be unbounded.

- **--minrecoveryrate *Rate*[**bBsSkMmG**]**

Sets the minimum recovery rate for a RAID logical volume to ensure that I/O for **sync** operations achieves a minimum throughput, even when heavy nominal I/O is present. The *Rate* is specified as an amount per second for each device in the array. If no suffix is given, then kB/sec/device is assumed.

## 118.14. CONVERTING A RAID LEVEL (RAID TAKEOVER)

LVM supports Raid *takeover*, which means converting a RAID logical volume from one RAID level to another (such as from RAID 5 to RAID 6). Changing the RAID level is usually done to increase or decrease resilience to device failures or to restripe logical volumes. You use the **lvconvert** for RAID takeover. For information on RAID takeover and for examples of using the **lvconvert** to convert a RAID logical volume, see the **lvmraid(7)** man page.

## 118.15. CHANGING ATTRIBUTES OF A RAID VOLUME (RAID RESHAPE)

RAID *reshaping* means changing attributes of a RAID logical volume while keeping the same RAID level. Some attributes you can change include RAID layout, stripe size, and number of stripes. For information on RAID reshaping and examples of using the **lvconvert** command to reshape a RAID logical volume, see the **lvmraid(7)** man page.

## 118.16. CONTROLLING I/O OPERATIONS ON A RAID1 LOGICAL VOLUME

You can control the I/O operations for a device in a RAID1 logical volume by using the **--writemostly** and **--writebehind** parameters of the **lvchange** command. The format for using these parameters is as follows.

- **--[raid]writemostly *PhysicalVolume*[:{t|y|n}]**

Marks a device in a RAID1 logical volume as **write-mostly**. All reads to these drives will be avoided unless necessary. Setting this parameter keeps the number of I/O operations to the drive to a minimum. By default, the **write-mostly** attribute is set to yes for the specified physical volume in the logical volume. It is possible to remove the **write-mostly** flag by appending **:n** to the physical volume or to toggle the value by specifying **:t**. The **--writemostly** argument can be specified more than one time in a single command, making it possible to toggle the write-mostly attributes for all the physical volumes in a logical volume at once.

- **--[raid]writebehind *IOCount***

Specifies the maximum number of outstanding writes that are allowed to devices in a RAID1 logical volume that are marked as **write-mostly**. Once this value is exceeded, writes become synchronous, causing all writes to the constituent devices to complete before the array signals the write has completed. Setting the value to zero clears the preference and allows the system to choose the value arbitrarily.

## 118.17. CHANGING THE REGION SIZE ON A RAID LOGICAL VOLUME

When you create a RAID logical volume, the region size for the logical volume will be the value of the **raid\_region\_size** parameter in the **/etc/lvm/lvm.conf** file. You can override this default value with the **-R** option of the **lvcreate** command.

After you have created a RAID logical volume, you can change the region size of the volume with the **-R** option of the **lvconvert** command. The following example changes the region size of logical volume **vg/raidlv** to 4096K. The RAID volume must be synced in order to change the region size.

```
lvconvert -R 4096K vg/raid1
```

```
Do you really want to change the region_size 512.00 KiB of LV vg/raid1 to 4.00 MiB? [y/n]: y
Changed region size on RAID LV vg/raid1 to 4.00 MiB.
```

# CHAPTER 119. SNAPSHOT LOGICAL VOLUMES

The LVM snapshot feature provides the ability to create virtual images of a device at a particular instant without causing a service interruption.

## 119.1. SNAPSHOT VOLUMES

The LVM snapshot feature provides the ability to create virtual images of a device at a particular instant without causing a service interruption. When a change is made to the original device (the origin) after a snapshot is taken, the snapshot feature makes a copy of the changed data area as it was prior to the change so that it can reconstruct the state of the device.



### NOTE

LVM supports thinly-provisioned snapshots.

Because a snapshot copies only the data areas that change after the snapshot is created, the snapshot feature requires a minimal amount of storage. For example, with a rarely updated origin, 3-5 % of the origin's capacity is sufficient to maintain the snapshot.



### NOTE

Snapshot copies of a file system are virtual copies, not an actual media backup for a file system. Snapshots do not provide a substitute for a backup procedure.

The size of the snapshot governs the amount of space set aside for storing the changes to the origin volume. For example, if you made a snapshot and then completely overwrote the origin the snapshot would have to be at least as big as the origin volume to hold the changes. You need to dimension a snapshot according to the expected level of change. So for example a short-lived snapshot of a read-mostly volume, such as `/usr`, would need less space than a long-lived snapshot of a volume that sees a greater number of writes, such as `/home`.

If a snapshot runs full, the snapshot becomes invalid, since it can no longer track changes on the origin volume. You should regularly monitor the size of the snapshot. Snapshots are fully resizable, however, so if you have the storage capacity you can increase the size of the snapshot volume to prevent it from getting dropped. Conversely, if you find that the snapshot volume is larger than you need, you can reduce the size of the volume to free up space that is needed by other logical volumes.

When you create a snapshot file system, full read and write access to the origin stays possible. If a chunk on a snapshot is changed, that chunk is marked and never gets copied from the original volume.

There are several uses for the snapshot feature:

- Most typically, a snapshot is taken when you need to perform a backup on a logical volume without halting the live system that is continuously updating the data.
- You can execute the **fsck** command on a snapshot file system to check the file system integrity and determine whether the original file system requires file system repair.
- Because the snapshot is read/write, you can test applications against production data by taking a snapshot and running tests against the snapshot, leaving the real data untouched.

- You can create LVM volumes for use with Red Hat Virtualization. LVM snapshots can be used to create snapshots of virtual guest images. These snapshots can provide a convenient way to modify existing guests or create new guests with minimal additional storage.

You can use the **--merge** option of the **lvconvert** command to merge a snapshot into its origin volume. One use for this feature is to perform system rollback if you have lost data or files or otherwise need to restore your system to a previous state. After you merge the snapshot volume, the resulting logical volume will have the origin volume's name, minor number, and UUID and the merged snapshot is removed.

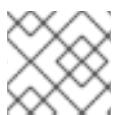
## 119.2. CREATING SNAPSHOT VOLUMES

Use the **-s** argument of the **lvcreate** command to create a snapshot volume. A snapshot volume is writable.



### NOTE

LVM snapshots are not supported across the nodes in a cluster. You cannot create a snapshot volume in a shared volume group. However, if you need to create a consistent backup of data on a shared logical volume you can activate the volume exclusively and then create the snapshot.



### NOTE

Snapshots are supported for RAID logical volumes.

LVM does not allow you to create a snapshot volume that is larger than the size of the origin volume plus needed metadata for the volume. If you specify a snapshot volume that is larger than this, the system will create a snapshot volume that is only as large as will be needed for the size of the origin.

By default, a snapshot volume is skipped during normal activation commands.

The following procedure creates an origin logical volume named **origin** and a snapshot volume of the original volume named **snap**.

1. Create a logical volume named **origin** from the volume group **VG**.

```
lvcreate -L 1G -n origin VG
Logical volume "origin" created.
```

2. Create a snapshot logical volume of **/dev/VG/origin** that is 100 MB in size named **snap**. If the original logical volume contains a file system, you can mount the snapshot logical volume on an arbitrary directory in order to access the contents of the file system to run a backup while the original file system continues to get updated.

```
lvcreate --size 100M --snapshot --name snap /dev/VG/origin
Logical volume "snap" created.
```

3. Display the status of logical volume **/dev/VG/origin**, showing all snapshot logical volumes and their status (active or inactive).

```
lvdiskutil /dev/VG/origin
--- Logical volume ---
LV Path /dev/VG/origin
```

```

LV Name origin
VG Name VG
LV UUID EsFoBp-CB9H-EpI5-pUO4-Yevi-EdFS-xtFnaF
LV Write Access read/write
LV Creation host, time host-083.virt.lab.msp.redhat.com, 2019-04-11 14:45:06 -0500
LV snapshot status source of
 snap [active]
LV Status available
open 0
LV Size 1.00 GiB
Current LE 256
Segments 1
Allocation inherit
Read ahead sectors auto
- currently set to 8192
Block device 253:6

```

4. The **lvs** command, by default, displays the origin volume and the current percentage of the snapshot volume being used. The following example shows the default output for the **lvs** command after you have created the snapshot volume, with a display that includes the devices that constitute the logical volumes.

```
lvs -a -o +devices
 LV VG Attr LSize Pool Origin Data% Meta% Move Log Cpy%Sync Convert
Devices
 origin VG owi-a-s--- 1.00g
 snap VG swi-a-s--- 100.00m origin 0.00
 /dev/sde1(0)
 /dev/sde1(256)
```

### WARNING



Because the snapshot increases in size as the origin volume changes, it is important to monitor the percentage of the snapshot volume regularly with the **lvs** command to be sure it does not fill. A snapshot that is 100% full is lost completely, as a write to unchanged parts of the origin would be unable to succeed without corrupting the snapshot.

In addition to the snapshot itself being invalidated when full, any mounted file systems on that snapshot device are forcibly unmounted, avoiding the inevitable file system errors upon access to the mount point. In addition, you can specify the **snapshot\_autoextend\_threshold** option in the **lvm.conf** file. This option allows automatic extension of a snapshot whenever the remaining snapshot space drops below the threshold you set. This feature requires that there be unallocated space in the volume group.

LVM does not allow you to create a snapshot volume that is larger than the size of the origin volume plus needed metadata for the volume. Similarly, automatic extension of a snapshot will not increase the size of a snapshot volume beyond the maximum calculated size that is necessary for the snapshot. Once a snapshot has grown large enough to cover the origin, it is no longer monitored for automatic extension.

Information on setting **snapshot\_autoextend\_threshold** and **snapshot\_autoextend\_percent** is provided in the **/etc/lvm/lvm.conf** file itself.

## 119.3. MERGING SNAPSHOT VOLUMES

You can use the **--merge** option of the **lvconvert** command to merge a snapshot into its origin volume. If both the origin and snapshot volume are not open, the merge will start immediately. Otherwise, the merge will start the first time either the origin or snapshot are activated and both are closed. Merging a snapshot into an origin that cannot be closed, for example a root file system, is deferred until the next time the origin volume is activated. When merging starts, the resulting logical volume will have the origin's name, minor number and UUID. While the merge is in progress, reads or writes to the origin appear as they were directed to the snapshot being merged. When the merge finishes, the merged snapshot is removed.

The following command merges snapshot volume **vg00/lvol1\_snap** into its origin.

```
lvconvert --merge vg00/lvol1_snap
```

You can specify multiple snapshots on the command line, or you can use LVM object tags to specify that multiple snapshots be merged to their respective origins. In the following example, logical volumes **vg00/lvol1**, **vg00/lvol2**, and **vg00/lvol3** are all tagged with the tag **@some\_tag**. The following command merges the snapshot logical volumes for all three volumes serially: **vg00/lvol1**, then **vg00/lvol2**, then **vg00/lvol3**. If the **--background** option were used, all snapshot logical volume merges would start in parallel.

```
lvconvert --merge @some_tag
```

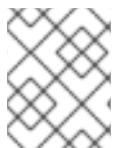
For further information on the **lvconvert --merge** command, see the **lvconvert(8)** man page.

# CHAPTER 120. CREATING AND MANAGING THINLY-PROVISIONED LOGICAL VOLUMES (THIN VOLUMES)

Logical volumes can be thinly provisioned. This allows you to create logical volumes that are larger than the available extents.

## 120.1. THINLY-PROVISIONED LOGICAL VOLUMES (THIN VOLUMES)

Logical volumes can be thinly provisioned. This allows you to create logical volumes that are larger than the available extents. Using thin provisioning, you can manage a storage pool of free space, known as a thin pool, which can be allocated to an arbitrary number of devices when needed by applications. You can then create devices that can be bound to the thin pool for later allocation when an application actually writes to the logical volume. The thin pool can be expanded dynamically when needed for cost-effective allocation of storage space.



### NOTE

Thin volumes are not supported across the nodes in a cluster. The thin pool and all its thin volumes must be exclusively activated on only one cluster node.

By using thin provisioning, a storage administrator can overcommit the physical storage, often avoiding the need to purchase additional storage. For example, if ten users each request a 100GB file system for their application, the storage administrator can create what appears to be a 100GB file system for each user but which is backed by less actual storage that is used only when needed. When using thin provisioning, it is important that the storage administrator monitor the storage pool and add more capacity if it starts to become full.

To make sure that all available space can be used, LVM supports data discard. This allows for re-use of the space that was formerly used by a discarded file or other block range.

Thin volumes provide support for a new implementation of copy-on-write (COW) snapshot logical volumes, which allow many virtual devices to share the same data in the thin pool.

## 120.2. CREATING THINLY-PROVISIONED LOGICAL VOLUMES

This procedure provides an overview of the basic commands you use to create and grow thinly-provisioned logical volumes. For detailed information on LVM thin provisioning as well as information on using the LVM commands and utilities with thinly-provisioned logical volumes, see the **lvmthin(7)** man page.

To create a thin volume, perform the following tasks:

1. Create a volume group with the **vgcreate** command.
2. Create a thin pool with the **lvcreate** command.
3. Create a thin volume in the thin pool with the **lvcreate** command.

You can use the **-T** (or **--thin**) option of the **lvcreate** command to create either a thin pool or a thin volume. You can also use **-T** option of the **lvcreate** command to create both a thin pool and a thin volume in that pool at the same time with a single command.

The following command uses the **-T** option of the **lvcreate** command to create a thin pool named **mythinpool** in the volume group **vg001** and that is 100M in size. Note that since you are creating a pool

of physical space, you must specify the size of the pool. The **-T** option of the **lvcreate** command does not take an argument; it deduces what type of device is to be created from the other options the command specifies.

```
lvcreate -L 100M -T vg001/mythinpoo
```

Thin pool volume with chunk size 64.00 KiB can address at most 15.81 TiB of data.

Logical volume "mythinpoo" created.

```
lvs
```

LV	VG	Attr	LSize	Pool	Origin	Data%	Meta%	Move	Log	Cpy%	Sync	Convert
mythinpoo	vg001	twi-a-tz--	100.00m			0.00	10.84					

The following command uses the **-T** option of the **lvcreate** command to create a thin volume named **thinvolume** in the thin pool **vg001/mythinpoo**. Note that in this case you are specifying virtual size, and that you are specifying a virtual size for the volume that is greater than the pool that contains it.

```
lvcreate -V 1G -T vg001/mythinpoo -n thinvolume
```

WARNING: Sum of all thin volume sizes (1.00 GiB) exceeds the size of thin pool vg001/mythinpoo (100.00 MiB).

WARNING: You have not turned on protection against thin pools running out of space.

WARNING: Set activation/thin\_pool\_autoextend\_threshold below 100 to trigger automatic extension of thin pools before they get full.

Logical volume "thinvolume" created.

```
lvs
```

LV	VG	Attr	LSize	Pool	Origin	Data%	Move	Log	Cpy%	Sync	Convert
mythinpoo	vg001	twi-a-tz	100.00m			0.00					
thinvolume	vg001	Vwi-a-tz	1.00g	mythinpoo		0.00					

The following command uses the **-T** option of the **lvcreate** command to create a thin pool and a thin volume in that pool by specifying both a size and a virtual size argument for the **lvcreate** command. This command creates a thin pool named **mythinpoo** in the volume group **vg001** and it also creates a thin volume named **thinvolume** in that pool.

```
lvcreate -L 100M -T vg001/mythinpoo -V 1G -n thinvolume
```

Thin pool volume with chunk size 64.00 KiB can address at most 15.81 TiB of data.

WARNING: Sum of all thin volume sizes (1.00 GiB) exceeds the size of thin pool vg001/mythinpoo (100.00 MiB).

WARNING: You have not turned on protection against thin pools running out of space.

WARNING: Set activation/thin\_pool\_autoextend\_threshold below 100 to trigger automatic extension of thin pools before they get full.

Logical volume "thinvolume" created.

```
lvs
```

LV	VG	Attr	LSize	Pool	Origin	Data%	Meta%	Move	Log	Cpy%	Sync	Convert
mythinpoo	vg001	twi-aotz--	100.00m			0.00	10.94					
thinvolume	vg001	Vwi-a-tz--	1.00g	mythinpoo		0.00						

You can also create a thin pool by specifying the **--thinpool** parameter of the **lvcreate** command. Unlike the **-T** option, the **--thinpool** parameter requires an argument, which is the name of the thin pool logical volume that you are creating. The following example specifies the **--thinpool** parameter of the **lvcreate** command to create a thin pool named **mythinpoo** in the volume group **vg001** and that is 100M in size:

```
lvcreate -L 100M --thinpool mythinpoo vg001
```

Thin pool volume with chunk size 64.00 KiB can address at most 15.81 TiB of data.

Logical volume "mythinpoo" created.

# lvs

LV	VG	Attr	LSize	Pool	Origin	Data%	Meta%	Move	Log	Cpy%	Sync	Convert
mythinpool	vg001	twi-a-tz--	100.00m			0.00	10.84					

Use the following criteria for using the chunk size:

- A smaller chunk size requires more metadata and hinders performance, but provides better space utilization with snapshots.
- A bigger chunk size requires less metadata manipulation, but makes the snapshot less space efficient.

By default, **lvm2** starts with a 64KiB chunk size and increases its value when the resulting size of the thin pool metadata device grows above 128MiB, this keeps the metadata size compact. However, this may result in some big chunk size values, which are less space efficient for snapshot usage. In such cases, a smaller chunk size and a bigger metadata size is a better option.

If the volume data size is in the range of TiB, use ~15.8GiB as the metadata size, which is the maximum supported size, and set the chunk size as per your requirement. But, note that it is not possible to increase the metadata size if you need to extend the volume's data size and have a small chunk size.



### WARNING

Red Hat does not recommend setting a chunk size smaller than the default value. If the chunk size is too small and your volume runs out of space for metadata, the volume is unable to create data. Monitor your logical volumes to ensure that they are expanded, or create more storage before the metadata volumes become completely full. Ensure that you set up your thin pool with a large enough chunk size so that they do not run out of room for the metadata.

Striping is supported for pool creation. The following command creates a 100M thin pool named **pool** in volume group **vg001** with two 64 kB stripes and a chunk size of 256 kB. It also creates a 1T thin volume, **vg00/thin\_lv**.

```
lvcreate -i 2 -I 64 -c 256 -L 100M -T vg00/pool -V 1T --name thin_lv
```

You can extend the size of a thin volume with the **lvextend** command. You cannot, however, reduce the size of a thin pool.

The following command resizes an existing thin pool that is 100M in size by extending it another 100M.

```
lvextend -L+100M vg001/mythinpool
```

Extending logical volume mythinpool to 200.00 MiB  
Logical volume mythinpool successfully resized

```
lvs
```

LV	VG	Attr	LSize	Pool	Origin	Data%	Move	Log	Copy%	Convert
mythinpool	vg001	twi-a-tz	200.00m			0.00				
thinvolume	vg001	Vwi-a-tz	1.00g	mythinpool		0.00				

As with other types of logical volumes, you can rename the volume with the **lvrename**, you can remove the volume with the **lvremove**, and you can display information about the volume with the **lvs** and **lvdiskdisplay** commands.

By default, the **lvcreate** command sets the size of the thin pool's metadata logical volume according to the formula (Pool\_LV\_size / Pool\_LV\_chunk\_size \* 64). If you will have large numbers of snapshots or if you have have small chunk sizes for your thin pool and thus expect significant growth of the size of the thin pool at a later time, you may need to increase the default value of the thin pool's metadata volume with the **--poolmetadatasize** parameter of the **lvcreate** command. The supported value for the thin pool's metadata logical volume is in the range between 2MiB and 16GiB.

You can use the **--thinpool** parameter of the **lvconvert** command to convert an existing logical volume to a thin pool volume. When you convert an existing logical volume to a thin pool volume, you must use the **--poolmetadata** parameter in conjunction with the **--thinpool** parameter of the **lvconvert** to convert an existing logical volume to the thin pool volume's metadata volume.



### NOTE

Converting a logical volume to a thin pool volume or a thin pool metadata volume destroys the content of the logical volume, since in this case the **lvconvert** does not preserve the content of the devices but instead overwrites the content.

The following example converts the existing logical volume **lv1** in volume group **vg001** to a thin pool volume and converts the existing logical volume **lv2** in volume group **vg001** to the metadata volume for that thin pool volume.

```
lvconvert --thinpool vg001/lv1 --poolmetadata vg001/lv2
Converted vg001/lv1 to thin pool.
```

## 120.3. THINLY-PROVISIONED SNAPSHOT VOLUMES

Red Hat Enterprise Linux provides support for thinly-provisioned snapshot volumes. Thin snapshot volumes allow many virtual devices to be stored on the same data volume. This simplifies administration and allows for the sharing of data between snapshot volumes.

As for all LVM snapshot volumes, as well as all thin volumes, thin snapshot volumes are not supported across the nodes in a cluster. The snapshot volume must be exclusively activated on only one cluster node.

Thin snapshot volumes provide the following benefits:

- A thin snapshot volume can reduce disk usage when there are multiple snapshots of the same origin volume.
- If there are multiple snapshots of the same origin, then a write to the origin will cause one COW operation to preserve the data. Increasing the number of snapshots of the origin should yield no major slowdown.
- Thin snapshot volumes can be used as a logical volume origin for another snapshot. This allows for an arbitrary depth of recursive snapshots (snapshots of snapshots of snapshots...).
- A snapshot of a thin logical volume also creates a thin logical volume. This consumes no data space until a COW operation is required, or until the snapshot itself is written.

- A thin snapshot volume does not need to be activated with its origin, so a user may have only the origin active while there are many inactive snapshot volumes of the origin.
- When you delete the origin of a thinly-provisioned snapshot volume, each snapshot of that origin volume becomes an independent thinly-provisioned volume. This means that instead of merging a snapshot with its origin volume, you may choose to delete the origin volume and then create a new thinly-provisioned snapshot using that independent volume as the origin volume for the new snapshot.

Although there are many advantages to using thin snapshot volumes, there are some use cases for which the older LVM snapshot volume feature may be more appropriate to your needs:

- You cannot change the chunk size of a thin pool. If the thin pool has a large chunk size (for example, 1MB) and you require a short-living snapshot for which a chunk size that large is not efficient, you may elect to use the older snapshot feature.
- You cannot limit the size of a thin snapshot volume; the snapshot will use all of the space in the thin pool, if necessary. This may not be appropriate for your needs.

In general, you should consider the specific requirements of your site when deciding which snapshot format to use.

## 120.4. CREATING THINLY-PROVISIONED SNAPSHOT VOLUMES

Red Hat Enterprise Linux provides support for thinly-provisioned snapshot volumes.



### NOTE

This section provides an overview of the basic commands you use to create and grow thinly-provisioned snapshot volumes. For detailed information on LVM thin provisioning as well as information on using the LVM commands and utilities with thinly-provisioned logical volumes, see the **lvmthin(7)** man page.



### IMPORTANT

When creating a thin snapshot volume, you do not specify the size of the volume. If you specify a size parameter, the snapshot that will be created will not be a thin snapshot volume and will not use the thin pool for storing data. For example, the command **lvcreate -s vg/thinvolume -L10M** will not create a thin snapshot, even though the origin volume is a thin volume.

Thin snapshots can be created for thinly-provisioned origin volumes, or for origin volumes that are not thinly-provisioned.

You can specify a name for the snapshot volume with the **--name** option of the **lvcreate** command. The following command creates a thinly-provisioned snapshot volume of the thinly-provisioned logical volume **vg001/thinvolume** that is named **mysnapshot1**.

```
lvcreate -s --name mysnapshot1 vg001/thinvolume
Logical volume "mysnapshot1" created
lvs
 LV VG Attr LSize Pool Origin Data% Move Log Copy% Convert
 mysnapshot1 vg001 Vwi-a-tz 1.00g mythinpool thinvolume 0.00
 mythinpool vg001 twi-a-tz 100.00m 0.00
 thinvolume vg001 Vwi-a-tz 1.00g mythinpool 0.00
```

A thin snapshot volume has the same characteristics as any other thin volume. You can independently activate the volume, extend the volume, rename the volume, remove the volume, and even snapshot the volume.

By default, a snapshot volume is skipped during normal activation commands. For information on controlling the activation of a logical volume, see [Logical volume activation](#) in the *Configuring and managing logical volumes* document.

You can also create a thinly-provisioned snapshot of a non-thinly-provisioned logical volume. Since the non-thinly-provisioned logical volume is not contained within a thin pool, it is referred to as an *external origin*. External origin volumes can be used and shared by many thinly-provisioned snapshot volumes, even from different thin pools. The external origin must be inactive and read-only at the time the thinly-provisioned snapshot is created.

To create a thinly-provisioned snapshot of an external origin, you must specify the **--thinpool** option. The following command creates a thin snapshot volume of the read-only inactive volume **origin\_volume**. The thin snapshot volume is named **mythinsnap**. The logical volume **origin\_volume** then becomes the thin external origin for the thin snapshot volume **mythinsnap** in volume group **vg001** that will use the existing thin pool **vg001/pool**. Because the origin volume must be in the same volume group as the snapshot volume, you do not need to specify the volume group when specifying the origin logical volume.

```
lvcreate -s --thinpool vg001/pool origin_volume --name mythinsnap
```

You can create a second thinly-provisioned snapshot volume of the first snapshot volume, as in the following command.

```
lvcreate -s vg001/mythinsnap --name my2ndthinsnap
```

You can display a list of all ancestors and descendants of a thin snapshot logical volume by specifying the **lv\_ancestors** and **lv\_descendants** reporting fields of the **lvs** command.

In the following example:

- **stack1** is an origin volume in volume group **vg001**.
- **stack2** is a snapshot of **stack1**
- **stack3** is a snapshot of **stack2**
- **stack4** is a snapshot of **stack3**

Additionally:

- **stack5** is also a snapshot of **stack2**
- **stack6** is a snapshot of **stack5**

```
$ lvs -o name,lv_ancestors,lv_descendants vg001
 LV Ancestors Descendants
 stack1 stack2,stack3,stack4,stack5,stack6
 stack2 stack1 stack3,stack4,stack5,stack6
 stack3 stack2,stack1 stack4
 stack4 stack3,stack2,stack1
```

```
stack5 stack2,stack1 stack6
stack6 stack5,stack2,stack1
pool
```



### NOTE

The **lv\_ancestors** and **lv\_descendants** fields display existing dependencies but do not track removed entries which can break a dependency chain if the entry was removed from the middle of the chain. For example, if you remove the logical volume **stack3** from this sample configuration, the display is as follows.

```
$ lvs -o name,lv_ancestors,lv_descendants vg001
 LV Ancestors Descendants
 stack1 stack2,stack5,stack6
 stack2 stack1 stack5,stack6
 stack4
 stack5 stack2,stack1 stack6
 stack6 stack5,stack2,stack1
 pool
```

You can configure your system to track and display logical volumes that have been removed, and you can display the full dependency chain that includes those volumes by specifying the **lv\_ancestors\_full** and **lv\_descendants\_full** fields.

## 120.5. TRACKING AND DISPLAYING THIN SNAPSHOT VOLUMES THAT HAVE BEEN REMOVED

You can configure your system to track thin snapshot and thin logical volumes that have been removed by enabling the **record\_lvs\_history** metadata option in the **lvm.conf** configuration file. This allows you to display a full thin snapshot dependency chain that includes logical volumes that have been removed from the original dependency chain and have become *historical* logical volumes.

You can configure your system to retain historical volumes for a defined period of time by specifying the retention time, in seconds, with the **lvs\_history\_retention\_time** metadata option in the **lvm.conf** configuration file.

A historical logical volume retains a simplified representation of the logical volume that has been removed, including the following reporting fields for the volume:

- **lv\_time\_removed**: the removal time of the logical volume
- **lv\_time**: the creation time of the logical volume
- **lv\_name**: the name of the logical volume
- **lv\_uuid**: the UUID of the logical volume
- **vg\_name**: the volume group that contains the logical volume.

When a volume is removed, the historical logical volume name acquires a hyphen as a prefix. For example, when you remove the logical volume **lvol1**, the name of the historical volume is **-lvol1**. A historical logical volume cannot be reactivated.

Even when the **record\_lvs\_history** metadata option enabled, you can prevent the retention of historical logical volumes on an individual basis when you remove a logical volume by specifying the **--nohistory** option of the **lvremove** command.

To include historical logical volumes in volume display, you specify the **-H|--history** option of an LVM display command. You can display a full thin snapshot dependency chain that includes historical volumes by specifying the **lv\_full\_ancestors** and **lv\_full\_descendants** reporting fields along with the **-H** option.

The following series of commands provides examples of how you can display and manage historical logical volumes.

1. Ensure that historical logical volumes are retained by setting **record\_lvs\_history=1** in the **lvm.conf** file. This metadata option is not enabled by default.
2. Enter the following command to display a thin provisioned snapshot chain.

In this example:

- **lvol1** is an origin volume, the first volume in the chain.
- **lvol2** is a snapshot of **lvol1**.
- **lvol3** is a snapshot of **lvol2**.
- **lvol4** is a snapshot of **lvol3**.
- **lvol5** is also a snapshot of **lvol3**.

Note that even though the example **lvs** display command includes the **-H** option, no thin snapshot volume has yet been removed and there are no historical logical volumes to display.

```
lvs -H -o name,full_ancestors,full_descendants
 LV FAncestors FDescendants
 lvol1 lvol2,lvol3,lvol4,lvol5
 lvol2 lvol1 lvol3,lvol4,lvol5
 lvol3 lvol2,lvol1 lvol4,lvol5
 lvol4 lvol3,lvol2,lvol1
 lvol5 lvol3,lvol2,lvol1
 pool
```

3. Remove logical volume **lvol3** from the snapshot chain, then run the following **lvs** command again to see how historical logical volumes are displayed, along with their ancestors and descendants.

```
lvremove -f vg/lvol3
Logical volume "lvol3" successfully removed
lvs -H -o name,full_ancestors,full_descendants
 LV FAncestors FDescendants
 lvol1 lvol2,-lvol3,lvol4,lvol5
 lvol2 lvol1 -lvol3,lvol4,lvol5
 -lvol3 lvol2,lvol1 lvol4,lvol5
 lvol4 -lvol3,lvol2,lvol1
 lvol5 -lvol3,lvol2,lvol1
 pool
```

4. You can use the **lv\_time\_removed** reporting field to display the time a historical volume was removed.

```
lvs -H -o name,full_ancestors,full_descendants,time_removed
 LV FAncors FDescndts RTime
 lvol1 lvol2,-lvol3,lvol4,lvol5
 lvol2 lvol1 -lvol3,lvol4,lvol5
 -lvol3 lvol2,lvol1 lvol4,lvol5 2016-03-14 14:14:32 +0100
 lvol4 -lvol3,lvol2,lvol1
 lvol5 -lvol3,lvol2,lvol1
 pool
```

5. You can reference historical logical volumes individually in a display command by specifying the `vgname/lvname` format, as in the following example. Note that the fifth bit in the `lv_attr` field is set to `h` to indicate the volume is a historical volume.

```
lvs -H vg/-lvol3
 LV VG Attr LSize
 -lvol3 vg ----h---- 0
```

6. LVM does not keep historical logical volumes if the volume has no live descendant. This means that if you remove a logical volume at the end of a snapshot chain, the logical volume is not retained as a historical logical volume.

```
lvremove -f vg/lvol5
Automatically removing historical logical volume vg/-lvol5.
Logical volume "lvol5" successfully removed
lvs -H -o name,full_ancestors,full_descendants
 LV FAncors FDescndts
 lvol1 lvol2,-lvol3,lvol4
 lvol2 lvol1 -lvol3,lvol4
 -lvol3 lvol2,lvol1 lvol4
 lvol4 -lvol3,lvol2,lvol1
 pool
```

7. Run the following commands to remove the volume **lvol1** and **lvol2** and to see how the **lvs** command displays the volumes once they have been removed.

```
lvremove -f vg/lvol1 vg/lvol2
Logical volume "lvol1" successfully removed
Logical volume "lvol2" successfully removed
lvs -H -o name,full_ancestors,full_descendants
 LV FAncors FDescndts
 -lvol1 -lvol2,-lvol3,lvol4
 -lvol2 -lvol1 -lvol3,lvol4
 -lvol3 -lvol2,-lvol1 lvol4
 lvol4 -lvol3,-lvol2,-lvol1
 pool
```

8. To remove a historical logical volume completely, you can run the **lvremove** command again, specifying the name of the historical volume that now includes the hyphen, as in the following example.

```
lvremove -f vg/-lvol3
Historical logical volume "lvol3" successfully removed
lvs -H -o name,full_ancestors,full_descendants
 LV FAncors FDescndts
```

```
-lvol1 -lvol2,lvol4
-lvol2 -lvol1 lvol4
lvol4 -lvol2,-lvol1
pool
```

9. A historical logical volumes is retained as long as there is a chain that includes live volumes in its descendants. This means that removing a historical logical volume also removes all of the logical volumes in the chain if no existing descendant is linked to them, as shown in the following example.

```
lvremove -f vg/lvol4
```

Automatically removing historical logical volume vg/-lvol1.

Automatically removing historical logical volume vg/-lvol2.

Automatically removing historical logical volume vg/-lvol4.

Logical volume "lvol4" successfully removed

# CHAPTER 121. ENABLING CACHING TO IMPROVE LOGICAL VOLUME PERFORMANCE

You can add caching to an LVM logical volume to improve performance. LVM then caches I/O operations to the logical volume using a fast device, such as an SSD.

The following procedures create a special LV from the fast device, and attach this special LV to the original LV to improve the performance.

## 121.1. CACHING METHODS IN LVM

LVM provides the following kinds of caching. Each one is suitable for different kinds of I/O patterns on the logical volume.

### **dm-cache**

This method speeds up access to frequently used data by caching it on the faster volume. The method caches both read and write operations.

The **dm-cache** method creates logical volumes of the type **cache**.

### **dm-writecache**

This method caches only write operations. The faster volume stores the write operations and then migrates them to the slower disk in the background. The faster volume is usually an SSD or a persistent memory (PMEM) disk.

The **dm-writecache** method creates logical volumes of the type **writecache**.

## 121.2. LVM CACHING COMPONENTS

When you enable caching for a logical volume, LVM renames and hides the original volumes, and presents a new logical volume that is composed of the original logical volumes. The composition of the new logical volume depends on the caching method and whether you are using the **cachevol** or **cachepool** option.

The **cachevol** and **cachepool** options expose different levels of control over the placement of the caching components:

- With the **cachevol** option, the faster device stores both the cached copies of data blocks and the metadata for managing the cache.
  - With the **cachepool** option, separate devices can store the cached copies of data blocks and the metadata for managing the cache.
- The **dm-writecache** method is not compatible with **cachepool**.

In all configurations, LVM exposes a single resulting device, which groups together all the caching components. The resulting device has the same name as the original slow logical volume.

## 121.3. ENABLING DM-CACHE CACHING FOR A LOGICAL VOLUME

This procedure enables caching of commonly used data on a logical volume using the **dm-cache** method.

### Prerequisites

- A slow logical volume that you want to speed up using **dm-cache** exists on your system.

- The volume group that contains the slow logical volume also contains an unused physical volume on a fast block device.

## Procedure

1. Create a **cachevol** volume on the fast device:

```
lvcreate --size cachevol-size --name fastvol vg /dev/fast-pv
```

Replace the following values:

**cachevol-size**

The size of the **cachevol** volume, such as **5G**

**fastvol**

A name for the **cachevol** volume

**vg**

The volume group name

**/dev/fast-pv**

The path to the fast block device, such as **/dev/sdf1**

2. Attach the **cachevol** volume to the main logical volume to begin caching:

```
lvconvert --type cache --cachevol fastvol vg/main-lv
```

Replace the following values:

**fastvol**

The name of the **cachevol** volume

**vg**

The volume group name

**main-lv**

The name of the slow logical volume

## Verification steps

- Examine the newly created devices:

```
lvs --all --options +devices vg
 LV Pool Type Devices
 main-lv [fastvol_cvol] cache main-lv_corig(0)
 [fastvol_cvol] linear /dev/fast-pv
 [main-lv_corig] linear /dev/slow-pv
```

## Additional resources

- You can also enable **dm-cache** caching in the **cachepool** configuration. This enables you to create the cache data and the cache metadata logical volumes individually and then combine the volumes into a single logical volume.

For information on this procedure and other details, including administrative examples, see the **lvmcache(7)** man page.

## 121.4. ENABLING DM-WRITECACHE CACHING FOR A LOGICAL VOLUME

This procedure enables caching of write I/O operations to a logical volume using the **dm-writecache** method.

### Prerequisites

- A slow logical volume that you want to speed up using **dm-writecache** exists on your system.
- The volume group that contains the slow logical volume also contains an unused physical volume on a fast block device.

### Procedure

1. If the slow logical volume is active, deactivate it:

```
lvchange --activate n vg/main-lv
```

Replace the following values:

**vg**

The volume group name

**main-lv**

The name of the slow logical volume

2. Create a deactivated **cachevol** volume on the fast device:

```
lvcreate --activate n --size cachevol-size --name fastvol vg /dev/fast-pv
```

Replace the following values:

**cachevol-size**

The size of the **cachevol** volume, such as **5G**

**fastvol**

A name for the **cachevol** volume

**vg**

The volume group name

**/dev/fast-pv**

The path to the fast block device, such as **/dev/sdf1**

3. Attach the **cachevol** volume to the main logical volume to begin caching:

```
lvconvert --type writecache --cachevol fastvol vg/main-lv
```

Replace the following values:

**fastvol**

The name of the **cachevol** volume

**vg**

The volume group name

**main-lv**

The name of the slow logical volume

4. Activate the resulting logical volume:

```
lvchange --activate y vg/main-lv
```

Replace the following values:

**vg**

The volume group name

**main-lv**

The name of the slow logical volume

### Verification steps

- Examine the newly created devices:

```
lvs --all --options +devices vg
```

LV	VG	Attr	LSize	Pool	Origin	Data%	Meta%	Move	Log
Cpy%Sync									
Convert									
Devices									
main-lv	vg	Cwi-a-C---	500.00m	[fastvol_cvol]	[main-lv_wcorig]	0.00			
main-lv_wcorig(0)									
[fastvol_cvol]	vg	Cwi-aoC---	252.00m						
/dev/sdc1(0)									
[main-lv_wcorig]	vg	owi-aoC---	500.00m						
/dev/sdb1(0)									

### Additional resources

- For information, including administrative examples, see the **lvmcache(7)** man page.

## 121.5. DISABLING CACHING FOR A LOGICAL VOLUME

This procedure disables **dm-cache** or **dm-writecache** caching that is currently enabled on a logical volume.

### Prerequisites

- Caching is enabled on a logical volume.

### Procedure

1. Deactivate the logical volume:

```
lvchange --activate n vg/main-lv
```

Replace the following values:

***vg***

The volume group name

***main-lv***

The name of the logical volume where caching is enabled

2. Detach the **cachevol** or **cachepool** volume:

```
lvconvert --splitcache vg/main-lv
```

Replace the following values:

***vg***

The volume group name

***main-lv***

The name of the logical volume where caching is enabled

## Verification steps

- Check that the logical volumes are no longer attached together:

```
lvs --all --options +devices [replaceable]_vg_
```

LV	Attr	Type	Devices
fastvol	-wi-----	linear	/dev/fast-pv
main-lv	-wi-----	linear	/dev/slow-pv

## Additional resources

- The **lvmcache(7)** man page

# CHAPTER 122. LOGICAL VOLUME ACTIVATION

A logical volume that is in an active state can be used through a block device. A logical volume that is activated is accessible and is subject to change. When you create a logical volume it is activated by default.

There are various circumstances for which you need to make an individual logical volume inactive and thus unknown to the kernel. You can activate or deactivate individual logical volume with the **-a** option of the **lvchange** command.

The format for the command to deactivate an individual logical volume is as follows.

```
lvchange -an vg/lv
```

The format for the command to activate an individual logical volume is as follows.

```
lvchange -ay vg/lv
```

You can and activate or deactivate all of the logical volumes in a volume group with the **-a** option of the **vgchange** command. This is the equivalent of running the **lvchange -a** command on each individual logical volume in the volume group.

The format for the command to deactivate all of the logical volumes in a volume group is as follows.

```
vgchange -an vg
```

The format for the command to activate all of the logical volumes in a volume group is as follows.

```
vgchange -ay vg
```

## 122.1. CONTROLLING AUTOACTIVATION OF LOGICAL VOLUMES

Autoactivation of a logical volume refers to the event-based automatic activation of a logical volume during system startup. As devices become available on the system (device online events), **systemd/udev** runs the **lvm2-pvscan** service for each device. This service runs the **pvscan --cache -aay device** command, which reads the named device. If the device belongs to a volume group, the **pvscan** command will check if all of the physical volumes for that volume group are present on the system. If so, the command will activate logical volumes in that volume group.

You can use the following configuration options in the **/etc/lvm/lvm.conf** configuration file to control autoactivation of logical volumes.

- **global/event\_activation**

When **event\_activation** is disabled, **systemd/udev** will autoactivate logical volume only on whichever physical volumes are present during system startup. If all physical volumes have not appeared yet, then some logical volumes may not be autoactivated.

- **activation/auto\_activation\_volume\_list**

Setting **auto\_activation\_volume\_list** to an empty list disables autoactivation entirely. Setting **auto\_activation\_volume\_list** to specific logical volumes and volume groups limits autoactivation to those logical volumes.

For information on setting these options, see the **/etc/lvm/lvm.conf** configuration file.

## 122.2. CONTROLLING LOGICAL VOLUME ACTIVATION

You can control the activation of logical volume in the following ways:

- Through the **activation/volume\_list** setting in the **/etc/lvm/conf** file. This allows you to specify which logical volumes are activated. For information on using this option, see the **/etc/lvm/lvm.conf** configuration file.
- By means of the activation skip flag for a logical volume. When this flag is set for a logical volume, the volume is skipped during normal activation commands.

You can set the activation skip flag on a logical volume in the following ways.

- You can turn off the activation skip flag when creating a logical volume by specifying the **-kn** or **--setactivationskip n** option of the **lvcreate** command.
- You can turn off the activation skip flag for an existing logical volume by specifying the **-kn** or **--setactivationskip n** option of the **lvchange** command.
- You can turn on the activation skip flag on again for a volume where it has been turned off with the **-ky** or **--setactivationskip y** option of the **lvchange** command.

To determine whether the activation skip flag is set for a logical volume run the **lvs** command, which displays the **k** attribute as in the following example.

```
lvs vg/thin1s1
LV VG Attr LSize Pool Origin
thin1s1 vg Vwi---tz-k 1.00t pool0 thin1
```

You can activate a logical volume with the **k** attribute set by using the **-K** or **--ignoreactivationskip** option in addition to the standard **-ay** or **--activate y** option.

By default, thin snapshot volumes are flagged for activation skip when they are created. You can control the default activation skip setting on new thin snapshot volumes with the **auto\_set\_activation\_skip** setting in the **/etc/lvm/lvm.conf** file.

The following command activates a thin snapshot logical volume that has the activation skip flag set.

```
lvchange -ay -K VG/SnapLV
```

The following command creates a thin snapshot without the activation skip flag

```
lvcreate --type thin -n SnapLV -kn -s ThinLV --thinpool VG/ThinPoolLV
```

The following command removes the activation skip flag from a snapshot logical volume.

```
lvchange -kn VG/SnapLV
```

## 122.3. ACTIVATING SHARED LOGICAL VOLUMES

You can control logical volume activation of a shared logical volume with the **-a** option of the **lvchange** and **vgchange** commands, as follows.

Command	Activation
<b>lvchange -ay e</b>	Activate the shared logical volume in exclusive mode, allowing only a single host to activate the logical volume. If the activation fails, as would happen if the logical volume is active on another host, an error is reported.
<b>lvchange -asy</b>	Activate the shared logical volume in shared mode, allowing multiple hosts to activate the logical volume concurrently. If the activation fails, as would happen if the logical volume is active exclusively on another host, an error is reported. If the logical type prohibits shared access, such as a snapshot, the command will report an error and fail. Logical volume types that cannot be used concurrently from multiple hosts include thin, cache, raid, and snapshot.
<b>lvchange -an</b>	Deactivate the logical volume.

## 122.4. ACTIVATING A LOGICAL VOLUME WITH MISSING DEVICES

You can configure which logical volumes with missing devices are activated by setting the **activation\_mode** parameter with the **lvchange** command to one of the following values.

Activation Mode	Meaning
complete	Allows only logical volumes with no missing physical volumes to be activated. This is the most restrictive mode.
degraded	Allows RAID logical volumes with missing physical volumes to be activated.
partial	Allows any logical volume with missing physical volumes to be activated. This option should be used for recovery or repair only.

The default value of **activation\_mode** is determined by the **activation\_mode** setting in the **/etc/lvm/lvm.conf** file. For further information, see the **lvmraid(7)** man page.

# CHAPTER 123. CONTROLLING LVM DEVICE SCANNING

You can control LVM device scanning by configuring filters in the `/etc/lvm/lvm.conf` file. The filters in the `lvm.conf` file consist of a series of simple regular expressions that get applied to the device names in the `/dev` directory to decide whether to accept or reject each block device found.

## 123.1. THE LVM DEVICE FILTER

LVM tools scan for devices in the `/dev` directory and check every device there for LVM metadata. A filter in the `/etc/lvm/lvm.conf` file controls which devices LVM scans.

The filter is a list of patterns that LVM applies to each device found by a scan of the `/dev` directory, or the directory specified by the `dir` keyword in the `/etc/lvm/lvm.conf` file. Patterns are regular expressions delimited by any character and preceded by `a` for *accept* or `r` for *reject*. The first regular expression in the list that matches a device determines if LVM accepts or rejects (ignores) the device. LVM accepts devices that do not match any patterns.

The following is the default configuration of the filter, which scans all devices:

```
filter = ["a/.*/"]
```

## 123.2. EXAMPLES OF LVM DEVICE FILTER CONFIGURATIONS

The following examples show the use of filters to control which devices LVM scans.



### WARNING

Some of the examples presented here might unintentionally match extra devices on the system and might not represent recommended practice for your system. For example, `a/loop/` is equivalent to `a/.*loop.*/` and would match `/dev/solooperation/lvol1`.

- The following filter adds all discovered devices, which is the default behavior because no filter is configured in the configuration file:

```
filter = ["a/.*/"]
```

- The following filter removes the `cdrom` device in order to avoid delays if the drive contains no media:

```
filter = ["r|^/dev/cdrom$|"]
```

- The following filter adds all loop devices and removes all other block devices:

```
filter = ["a/loop/", "r/.*/"]
```

- The following filter adds all loop and IDE devices and removes all other block devices:

■

```
filter = ["a|loop|", "a|/dev/hd.*|", "r|.*|"]
```

- The following filter adds just partition 8 on the first IDE drive and removes all other block devices:

```
filter = ["a|^/dev/hda8$", "r|.*/"]
```

## 123.3. APPLYING AN LVM DEVICE FILTER CONFIGURATION

This procedure changes the configuration of the LVM device filter, which controls the devices that LVM scans.

### Prerequisites

- Prepare the device filter pattern that you want to use.

### Procedure

1. Test your device filter pattern without modifying the **/etc/lvm/lvm.conf** file.

Use an LVM command with the **--config 'devices{ filter = [ *your device filter pattern* ] }'** option. For example:

```
lvs --config 'devices{ filter = ["a|/dev/emcpower.*|", "r|.*|"] }'
```

2. Edit the **filter** option in the **/etc/lvm/lvm.conf** configuration file to use your new device filter pattern.

3. Check that no physical volumes or volume groups that you want to use are missing with the new configuration:

```
pvscan
```

```
vgscan
```

4. Rebuild the **initramfs** file system so that LVM scans only the necessary devices upon reboot:

```
dracut --force --verbose
```

# CHAPTER 124. CONTROLLING LVM ALLOCATION

By default, a volume group allocates physical extents according to common-sense rules such as not placing parallel stripes on the same physical volume. This is the **normal** allocation policy. You can use the **--alloc** argument of the **vgcreate** command to specify an allocation policy of **contiguous**, **anywhere**, or **cling**. In general, allocation policies other than **normal** are required only in special cases where you need to specify unusual or nonstandard extent allocation.

## 124.1. LVM ALLOCATION POLICIES

When an LVM operation needs to allocate physical extents for one or more logical volumes, the allocation proceeds as follows:

- The complete set of unallocated physical extents in the volume group is generated for consideration. If you supply any ranges of physical extents at the end of the command line, only unallocated physical extents within those ranges on the specified physical volumes are considered.
- Each allocation policy is tried in turn, starting with the strictest policy (**contiguous**) and ending with the allocation policy specified using the **--alloc** option or set as the default for the particular logical volume or volume group. For each policy, working from the lowest-numbered logical extent of the empty logical volume space that needs to be filled, as much space as possible is allocated, according to the restrictions imposed by the allocation policy. If more space is needed, LVM moves on to the next policy.

The allocation policy restrictions are as follows:

- An allocation policy of **contiguous** requires that the physical location of any logical extent that is not the first logical extent of a logical volume is adjacent to the physical location of the logical extent immediately preceding it.  
When a logical volume is striped or mirrored, the **contiguous** allocation restriction is applied independently to each stripe or mirror image (leg) that needs space.
  - An allocation policy of **cling** requires that the physical volume used for any logical extent be added to an existing logical volume that is already in use by at least one logical extent earlier in that logical volume. If the configuration parameter **allocation/cling\_tag\_list** is defined, then two physical volumes are considered to match if any of the listed tags is present on both physical volumes. This allows groups of physical volumes with similar properties (such as their physical location) to be tagged and treated as equivalent for allocation purposes.  
When a Logical Volume is striped or mirrored, the **cling** allocation restriction is applied independently to each stripe or mirror image (leg) that needs space.
  - An allocation policy of **normal** will not choose a physical extent that shares the same physical volume as a logical extent already allocated to a parallel logical volume (that is, a different stripe or mirror image/leg) at the same offset within that parallel logical volume.  
When allocating a mirror log at the same time as logical volumes to hold the mirror data, an allocation policy of **normal** will first try to select different physical volumes for the log and the data. If that is not possible and the **allocation/mirror\_logs\_require\_separate\_pvs** configuration parameter is set to 0, it will then allow the log to share physical volume(s) with part of the data.
- Similarly, when allocating thin pool metadata, an allocation policy of **normal** will follow the same considerations as for allocation of a mirror log, based on the value of the **allocation/thin\_pool\_metadata\_require\_separate\_pvs** configuration parameter.

- If there are sufficient free extents to satisfy an allocation request but a **normal** allocation policy would not use them, the **anywhere** allocation policy will, even if that reduces performance by placing two stripes on the same physical volume.

The allocation policies can be changed using the **vgchange** command.



### NOTE

If you rely upon any layout behavior beyond that documented in this section according to the defined allocation policies, you should note that this might change in future versions of the code. For example, if you supply on the command line two empty physical volumes that have an identical number of free physical extents available for allocation, LVM currently considers using each of them in the order they are listed; there is no guarantee that future releases will maintain that property. If it is important to obtain a specific layout for a particular Logical Volume, then you should build it up through a sequence of **lvcreate** and **lvconvert** steps such that the allocation policies applied to each step leave LVM no discretion over the layout.

To view the way the allocation process currently works in any specific case, you can read the debug logging output, for example by adding the **-vvvv** option to a command.

## 124.2. PREVENTING ALLOCATION ON A PHYSICAL VOLUME

You can prevent allocation of physical extents on the free space of one or more physical volumes with the **pvchange** command. This may be necessary if there are disk errors, or if you will be removing the physical volume.

The following command disallows the allocation of physical extents on **/dev/sdk1**.

```
pvchange -x n /dev/sdk1
```

You can also use the **-xy** arguments of the **pvchange** command to allow allocation where it had previously been disallowed.

## 124.3. EXTENDING A LOGICAL VOLUME WITH THE CLING ALLOCATION POLICY

When extending an LVM volume, you can use the **--alloc cling** option of the **lvextend** command to specify the **cling** allocation policy. This policy will choose space on the same physical volumes as the last segment of the existing logical volume. If there is insufficient space on the physical volumes and a list of tags is defined in the **/etc/lvm/lvm.conf** file, LVM will check whether any of the tags are attached to the physical volumes and seek to match those physical volume tags between existing extents and new extents.

For example, if you have logical volumes that are mirrored between two sites within a single volume group, you can tag the physical volumes according to where they are situated by tagging the physical volumes with **@site1** and **@site2** tags. You can then specify the following line in the **lvm.conf** file:

```
cling_tag_list = ["@site1", "@site2"]
```

In the following example, the **lvm.conf** file has been modified to contain the following line:

```
cling_tag_list = ["@A", "@B"]
```

Also in this example, a volume group **taft** has been created that consists of the physical volumes **/dev/sdb1**, **/dev/sdc1**, **/dev/sdd1**, **/dev/sde1**, **/dev/sdf1**, **/dev/sdg1**, and **/dev/sdh1**. These physical volumes have been tagged with tags **A**, **B**, and **C**. The example does not use the **C** tag, but this will show that LVM uses the tags to select which physical volumes to use for the mirror legs.

```
pvs -a -o +pv_tags /dev/sd[bcdefgh]
PV VG Fmt Attr PSize PFree PV Tags
/dev/sdb1 taft lvm2 a-- 15.00g 15.00g A
/dev/sdc1 taft lvm2 a-- 15.00g 15.00g B
/dev/sdd1 taft lvm2 a-- 15.00g 15.00g B
/dev/sde1 taft lvm2 a-- 15.00g 15.00g C
/dev/sdf1 taft lvm2 a-- 15.00g 15.00g C
/dev/sdg1 taft lvm2 a-- 15.00g 15.00g A
/dev/sdh1 taft lvm2 a-- 15.00g 15.00g A
```

The following command creates a 10 gigabyte mirrored volume from the volume group **taft**.

```
lvcreate --type raid1 -m 1 -n mirror --nosync -L 10G taft
WARNING: New raid1 won't be synchronised. Don't read what you didn't write!
Logical volume "mirror" created
```

The following command shows which devices are used for the mirror legs and RAID metadata subvolumes.

```
lvs -a -o +devices
LV VG Attr LSize Log Cpy%Sync Devices
mirror taft Rwi-a-r-- 10.00g 100.00 mirror_rimage_0(0),mirror_rimage_1(0)
[mirror_rimage_0] taft iwi-aor-- 10.00g /dev/sdb1(1)
[mirror_rimage_1] taft iwi-aor-- 10.00g /dev/sdc1(1)
[mirror_rmeta_0] taft ewi-aor-- 4.00m /dev/sdb1(0)
[mirror_rmeta_1] taft ewi-aor-- 4.00m /dev/sdc1(0)
```

The following command extends the size of the mirrored volume, using the **cling** allocation policy to indicate that the mirror legs should be extended using physical volumes with the same tag.

```
lvextend --alloc cling -L +10G taft/mirror
Extending 2 mirror images.
Extending logical volume mirror to 20.00 GiB
Logical volume mirror successfully resized
```

The following display command shows that the mirror legs have been extended using physical volumes with the same tag as the leg. Note that the physical volumes with a tag of **C** were ignored.

```
lvs -a -o +devices
LV VG Attr LSize Log Cpy%Sync Devices
mirror taft Rwi-a-r-- 20.00g 100.00 mirror_rimage_0(0),mirror_rimage_1(0)
[mirror_rimage_0] taft iwi-aor-- 20.00g /dev/sdb1(1)
[mirror_rimage_0] taft iwi-aor-- 20.00g /dev/sdg1(0)
[mirror_rimage_1] taft iwi-aor-- 20.00g /dev/sdc1(1)
[mirror_rimage_1] taft iwi-aor-- 20.00g /dev/sdd1(0)
[mirror_rmeta_0] taft ewi-aor-- 4.00m /dev/sdb1(0)
[mirror_rmeta_1] taft ewi-aor-- 4.00m /dev/sdc1(0)
```

# CHAPTER 125. TROUBLESHOOTING LVM

You can use LVM tools to troubleshoot a variety of issues in LVM volumes and groups.

## 125.1. GATHERING DIAGNOSTIC DATA ON LVM

If an LVM command is not working as expected, you can gather diagnostics in the following ways.

### Procedure

- Use the following methods to gather different kinds of diagnostic data:
  - Add the **-vvvv** argument to any LVM command to increase the verbosity level of the command output.
  - In the **log** section of the **/etc/lvm/lvm.conf** configuration file, increase the value of the **level** option. This causes LVM to provide more details in the system log.
  - If the problem is related to the logical volume activation, enable LVM to log messages during the activation:
    - i. Set the **activation = 1** option in the **log** section of the **/etc/lvm/lvm.conf** configuration file.
    - ii. Run the LVM command with the **-vvvv** option.
    - iii. Examine the command output.
  - iv. Reset the **activation** option to **0**.  
If you do not reset the option to **0**, the system might become unresponsive during low memory situations.
- Display an information dump for diagnostic purposes:

```
lvmdump
```
- Display additional system information:

```
lvs -v
```

```
pvs --all
```

```
dmsetup info --columns
```
- Examine the last backup of the LVM metadata in the **/etc/lvm/backup/** directory and archived versions in the **/etc/lvm/archive/** directory.
- Check the current configuration information:

```
lvmconfig
```
- Check the **/run/lvm/hints** cache file for a record of which devices have physical volumes on them.

## Additional resources

- The **lvmdump(8)** man page

## 125.2. DISPLAYING INFORMATION ON FAILED LVM DEVICES

You can display information about a failed LVM volume that can help you determine why the volume failed.

### Procedure

- Display the failed volumes using the **vgs** or **lvs** utility.

#### Example 125.1. Failed volume groups

In this example, one of the devices that made up the volume group **vg** failed. The volume group is unusable but you can see information about the failed device.

```
vgs --options +devices

/dev/sdb: open failed: No such device or address
/dev/sdb: open failed: No such device or address
WARNING: Couldn't find device with uuid 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s.
WARNING: VG vg is missing PV 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s (last
written to /dev/sdb1).
WARNING: Couldn't find all devices for LV vg/linear while checking used and assumed
devices.
WARNING: Couldn't find all devices for LV vg/stripe while checking used and assumed
devices.
VG #PV #LV #SN Attr VSize VFree Devices
vg 2 2 0 wz-pn- <3.64t <3.60t [unknown](0)
vg 2 2 0 wz-pn- <3.64t <3.60t [unknown](5120),/dev/sdc1(0)
```

#### Example 125.2. Failed linear and striped LV

In this example, the failed device caused both a linear and a striped logical volume in the volume group to fail. The command output shows the failed logical volumes.

```
lvs --all --options +devices

/dev/sdb: open failed: No such device or address
/dev/sdb: open failed: No such device or address
WARNING: Couldn't find device with uuid 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s.
WARNING: VG vg is missing PV 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s (last
written to /dev/sdb1).
WARNING: Couldn't find all devices for LV vg/linear while checking used and assumed
devices.
WARNING: Couldn't find all devices for LV vg/stripe while checking used and assumed
devices.
LV VG Attr LSize Pool Origin Data% Meta% Move Log Cpy%Sync Convert
Devices
```

linear vg -wi-a---p- 20.00g stripe vg -wi-a---p- 20.00g (5120),/dev/sdc1(0)	[unknown](0) [unknown]
-----------------------------------------------------------------------------------	---------------------------

### Example 125.3. Failed leg of a mirrored logical volume

The following examples show the command output from the **vgs** and **lvs** utilities when a leg of a mirrored logical volume has failed.

```
vgs --all --options +devices

VG #PV #LV #SN Attr VSize VFree Devices
corey 4 4 0 rz-pnc 1.58T 1.34T my_mirror_mimage_0(0),my_mirror_mimage_1(0)
corey 4 4 0 rz-pnc 1.58T 1.34T /dev/sdd1(0)
corey 4 4 0 rz-pnc 1.58T 1.34T unknown device(0)
corey 4 4 0 rz-pnc 1.58T 1.34T /dev/sdb1(0)

lvs --all --options +devices

LV VG Attr LSize Origin Snap% Move Log Copy% Devices
my_mirror corey mwi-a- 120.00G my_mirror_mlog 1.95
my_mirror_mimage_0(0),my_mirror_mimage_1(0)
[my_mirror_mimage_0] corey iwi-ao 120.00G unknown device(0)
[my_mirror_mimage_1] corey iwi-ao 120.00G /dev/sdb1(0)
[my_mirror_mlog] corey lwi-ao 4.00M /dev/sdd1(0)
```

## 125.3. REMOVING LOST LVM PHYSICAL VOLUMES FROM A VOLUME GROUP

If a physical volume fails, you can activate the remaining physical volumes in the volume group and remove all the logical volumes that used that physical volume from the volume group.

### Procedure

1. Activate the remaining physical volumes in the volume group:

```
vgchange --activate y --partial volume-group
```

2. Check which logical volumes will be removed:

```
vgreduce --removemissing --test volume-group
```

3. Remove all the logical volumes that used the lost physical volume from the volume group:

```
vgreduce --removemissing --force volume-group
```

4. Optional: If you accidentally removed logical volumes that you wanted to keep, you can reverse the **vgreduce** operation:

```
vcfgrestore volume-group
```



### WARNING

If you removed a thin pool, LVM cannot reverse the operation.

## 125.4. RECOVERING AN LVM PHYSICAL VOLUME WITH DAMAGED METADATA

If the volume group metadata area of a physical volume is accidentally overwritten or otherwise destroyed, you get an error message indicating that the metadata area is incorrect, or that the system was unable to find a physical volume with a particular UUID. You might be able to recover the data from the physical volume by rewriting the metadata area on the physical volume.

### 125.4.1. Discovering that an LVM volume has missing or corrupted metadata

The following example shows the command output you might see if the metadata area on a physical volume is missing or corrupted.

#### Procedure

- Try to list the logical volumes:

```
lvs --all --options +devices
```

#### Example 125.4. Output with missing or corrupted metadata

In this example, certain logical volumes are located on a physical volume that has missing or corrupted metadata.

```
Couldn't find device with uuid 'FmGRh3-zhok-iVI8-7qTD-S5BI-MAEN-NYM5Sk'.
Couldn't find all physical volumes for volume group VG.
Couldn't find device with uuid 'FmGRh3-zhok-iVI8-7qTD-S5BI-MAEN-NYM5Sk'.
Couldn't find all physical volumes for volume group VG.
...
...
```

### 125.4.2. Finding the metadata of a missing LVM physical volume

This procedure finds the latest archived metadata of a physical volume that is missing or corrupted.

#### Procedure

- Find the archived metadata file of the volume group that contains the physical volume. The archived metadata files are located at the **/etc/lvm/archive/volume-group-name\_backup-number.vg** path. Select the last known valid metadata file, which has the highest number for the volume group.

- Find the UUID of the physical volume. Use one of the following methods.

- List the logical volumes:

```
lvs --all --options +devices
```

```
Couldn't find device with uuid 'FmGRh3-zhok-iVl8-7qTD-S5BI-MAEN-NYM5Sk'.
```

- Examine the archived metadata file. Find the UUID as the value labeled **id =** in the **physical\_volumes** section of the volume group configuration.
- Deactivate the volume group using the **--partial** option:

```
vgchange --activate n --partial volume-group-name
```

```
PARTIAL MODE. Incomplete logical volumes will be processed.
```

```
WARNING: Couldn't find device with uuid 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s.
```

```
WARNING: VG raid_sanity is missing PV 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s (last written to /dev/sdb1).
```

```
0 logical volume(s) in volume group "raid_sanity" now active
```

### 125.4.3. Restoring metadata on an LVM physical volume

This procedure restores metadata on a physical volume that is either corrupted or replaced with a new device.



#### WARNING

Do not attempt this procedure on a working LVM logical volume. You will lose your data if you specify the incorrect UUID.

#### Prerequisites

- You have identified the metadata of the missing physical volume. For details, see [Section 125.4.2, “Finding the metadata of a missing LVM physical volume”](#).

#### Procedure

- Restore the metadata on the physical volume:

```
pvcreate --uuid physical-volume-uuid \
--restorefile /etc/lvm/archive/volume-group-name_backup-number.vg \
block-device
```



#### NOTE

The command overwrites only the LVM metadata areas and does not affect the existing data areas.

**Example 125.5. Restoring a physical volume on /dev/sdh1**

The following example labels the **/dev/sdh1** device as a physical volume with the following properties:

- The UUID of **FmGRh3-zhok-iVI8-7qTD-S5BI-MAEN-NYM5Sk**
- The metadata information contained in **VG\_00050.vg**, which is the most recent good archived metadata for the volume group

```
pvcreate --uuid "FmGRh3-zhok-iVI8-7qTD-S5BI-MAEN-NYM5Sk" \
--restorefile /etc/lvm/archive/VG_00050.vg \
/dev/sdh1
```

...  
Physical volume "/dev/sdh1" successfully created

2. Restore the metadata of the volume group:

```
vgcfgrestore volume-group-name
```

Restored volume group *volume-group-name*

3. Display the logical volumes on the volume group:

```
lvs --all --options +devices volume-group-name
```

The logical volumes are currently inactive. For example:

LV	VG	Attr	LSize	Origin	Snap%	Move	Log	Copy%	Devices
stripe	VG	-wi---	300.00G						/dev/sdh1 (0),/dev/sda1(0)
stripe	VG	-wi---	300.00G						/dev/sdh1 (34728),/dev/sdb1(0)

4. If the segment type of the logical volumes is RAID or mirror, resynchronize the logical volumes:

```
lvchange --resync volume-group-name/logical-volume-name
```

5. Activate the logical volumes:

```
lvchange --activate y /dev/volume-group-name/logical-volume-name
```

6. If the on-disk LVM metadata takes at least as much space as what overrode it, this procedure can recover the physical volume. If what overrode the metadata went past the metadata area, the data on the volume may have been affected. You might be able to use the fsck command to recover that data.

**Verification steps**

- Display the active logical volumes:

```
lvs --all --options +devices
```

```

LV VG Attr LSize Origin Snap% Move Log Copy% Devices
stripe VG -wi-a- 300.00G /dev/sdh1 (0),/dev/sda1(0)
stripe VG -wi-a- 300.00G /dev/sdh1 (34728),/dev/sdb1(0)

```

## 125.5. REPLACING A MISSING LVM PHYSICAL VOLUME

If a physical volume fails or otherwise needs to be replaced, you can label a new physical volume to replace the one that has been lost in the existing volume group.

### Prerequisites

- You have replaced the physical volume with a new storage device.  
TODO: Reevaluate the placement of this step.

#### 125.5.1. Finding the metadata of a missing LVM physical volume

This procedure finds the latest archived metadata of a physical volume that is missing or corrupted.

### Procedure

1. Find the archived metadata file of the volume group that contains the physical volume. The archived metadata files are located at the **/etc/lvm/archive/volume-group-name\_backup-number.vg** path. Select the last known valid metadata file, which has the highest number for the volume group.
2. Find the UUID of the physical volume. Use one of the following methods.
  - List the logical volumes:

```
lvs --all --options +devices
```

Couldn't find device with uuid 'FmGRh3-zhok-iVl8-7qTD-S5BI-MAEN-NYM5SK'.

- Examine the archived metadata file. Find the UUID as the value labeled **id =** in the **physical\_volumes** section of the volume group configuration.
  - Deactivate the volume group using the **--partial** option:
- ```
# vgchange --activate n --partial volume-group-name
```
- PARTIAL MODE. Incomplete logical volumes will be processed.
WARNING: Couldn't find device with uuid 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s.
WARNING: VG *raid_sanity* is missing PV 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s (last written to /dev/sdb1).
0 logical volume(s) in volume group "raid_sanity" now active

125.5.2. Restoring metadata on an LVM physical volume

This procedure restores metadata on a physical volume that is either corrupted or replaced with a new device.

**WARNING**

Do not attempt this procedure on a working LVM logical volume. You will lose your data if you specify the incorrect UUID.

Prerequisites

- You have identified the metadata of the missing physical volume. For details, see [Section 125.5.1, “Finding the metadata of a missing LVM physical volume”](#).

Procedure

1. Restore the metadata on the physical volume:

```
# pvcreate --uuid physical-volume-uuid \
--restorefile /etc/lvm/archive/volume-group-name_backup-number.vg \
block-device
```

**NOTE**

The command overwrites only the LVM metadata areas and does not affect the existing data areas.

Example 125.6. Restoring a physical volume on /dev/sdh1

The following example labels the **/dev/sdh1** device as a physical volume with the following properties:

- The UUID of **FmGRh3-zhok-iVI8-7qTD-S5BI-MAEN-NYM5Sk**
- The metadata information contained in **VG_00050.vg**, which is the most recent good archived metadata for the volume group

```
# pvcreate --uuid "FmGRh3-zhok-iVI8-7qTD-S5BI-MAEN-NYM5Sk" \
--restorefile /etc/lvm/archive/VG_00050.vg \
/dev/sdh1
```

...

Physical volume "/dev/sdh1" successfully created

2. Restore the metadata of the volume group:

```
# vgcfgrestore volume-group-name
Restored volume group volume-group-name
```

3. Display the logical volumes on the volume group:

```
# lvs --all --options +devices volume-group-name
```

The logical volumes are currently inactive. For example:

| LV | VG | Attr | LSize | Origin | Snap% | Move | Log | Copy% | Devices |
|--------|----|--------|---------|--------|-------|------|-----|-------|--------------------------------|
| stripe | VG | -wi--- | 300.00G | | | | | | /dev/sdh1 (0),/dev/sda1(0) |
| stripe | VG | -wi--- | 300.00G | | | | | | /dev/sdh1 (34728),/dev/sdb1(0) |

4. If the segment type of the logical volumes is RAID or mirror, resynchronize the logical volumes:

```
# lvchange --resync volume-group-name/logical-volume-name
```

5. Activate the logical volumes:

```
# lvchange --activate y /dev/volume-group-name/logical-volume-name
```

6. If the on-disk LVM metadata takes at least as much space as what overrode it, this procedure can recover the physical volume. If what overrode the metadata went past the metadata area, the data on the volume may have been affected. You might be able to use the fsck command to recover that data.

Verification steps

- Display the active logical volumes:

```
# lvs --all --options +devices
```

| LV | VG | Attr | LSize | Origin | Snap% | Move | Log | Copy% | Devices |
|--------|----|--------|---------|--------|-------|------|-----|-------|--------------------------------|
| stripe | VG | -wi-a- | 300.00G | | | | | | /dev/sdh1 (0),/dev/sda1(0) |
| stripe | VG | -wi-a- | 300.00G | | | | | | /dev/sdh1 (34728),/dev/sdb1(0) |

125.6. TROUBLESHOOTING INSUFFICIENT FREE EXTENTS FOR A LOGICAL VOLUME

You might get the **Insufficient free extents** error message when attempting to create a logical volume, even when you think that the volume group has enough free space. You can troubleshoot this error to be able to create a logical volume on the volume group.

125.6.1. Volume groups

Physical volumes are combined into volume groups (VGs). This creates a pool of disk space out of which logical volumes can be allocated.

Within a volume group, the disk space available for allocation is divided into units of a fixed-size called extents. An extent is the smallest unit of space that can be allocated. Within a physical volume, extents are referred to as physical extents.

A logical volume is allocated into logical extents of the same size as the physical extents. The extent size is thus the same for all logical volumes in the volume group. The volume group maps the logical extents to physical extents.

125.6.2. Rounding errors in LVM output

LVM commands that report the space usage in volume groups round the reported number to 2 decimal places to provide human-readable output. This includes the **vgdisplay** and **vgs** utilities.

As a result of the rounding, the reported value of free space might be larger than what the physical extents on the volume group provide. If you attempt to create a logical volume the size of the reported free space, you might get the following error:

Insufficient free extents

To work around the error, you must examine the number of free physical extents on the volume group, which is the accurate value of free space. You can then use the number of extents to create the logical volume successfully.

125.6.3. Preventing the rounding error when creating an LVM volume

When creating an LVM logical volume, you can specify the size of the logical volume so that no rounding error occurs.

Procedure

1. Find the number of free physical extents in the volume group:

```
# vgdisplay volume-group-name
```

Example 125.7. Free extents in a volume group

For example, the following volume group has 8780 free physical extents:

```
--- Volume group ---  
...  
Free PE / Size 8780 / 34.30 GB
```

2. Create the logical volume. Enter the volume size in extents rather than bytes.

Example 125.8. Creating a logical volume by specifying the number of extents

```
# lvcreate --extents 8780 --name testlv testvg
```

Example 125.9. Creating a logical volume to occupy all the remaining space

Alternately, you can extend the logical volume to use a percentage of the remaining free space in the volume group. For example:

```
# lvcreate --extents 100%FREE --name testlv2 testvg
```

Verification steps

- Check the number of extents that the volume group now uses:

```
# vgs --options +vg_free_count,vg_extent_count
  VG #PV #LV #SN Attr  VSize  VFree Free #Ext
  testvg 2 1 0 wz--n- 34.30G 0 0 8780
```

125.7. TROUBLESHOOTING DUPLICATE PHYSICAL VOLUME WARNINGS FOR MULTIPATHED LVM DEVICES

When using LVM with multipathed storage, LVM commands that list a volume group or logical volume might display messages such as the following:

```
Found duplicate PV GDjTZf7Y03GJHjteqOwrye2dcSCjdaUi: using /dev/dm-5 not /dev/sdd
Found duplicate PV GDjTZf7Y03GJHjteqOwrye2dcSCjdaUi: using /dev/emcpowerb not /dev/sde
Found duplicate PV GDjTZf7Y03GJHjteqOwrye2dcSCjdaUi: using /dev/sddlmab not /dev/sdf
```

You can troubleshoot these warnings to understand why LVM displays them, or to hide the warnings.

125.7.1. Root cause of duplicate PV warnings

When a multipath software such as Device Mapper Multipath (DM Multipath), EMC PowerPath, or Hitachi Dynamic Link Manager (HDLM) manages storage devices on the system, each path to a particular logical unit (LUN) is registered as a different SCSI device. The multipath software then creates a new device that maps to those individual paths. Because each LUN has multiple device nodes in the **/dev** directory that point to the same underlying data, all the device nodes contain the same LVM metadata.

Table 125.1. Example device mappings in different multipath software

| Multipath software | SCSI paths to a LUN | Multipath device mapping to paths |
|--------------------|-------------------------------------|--|
| DM Multipath | /dev/sdb and /dev/sdc | /dev/mapper/mpath1 or
/dev/mapper/mptha |
| EMC PowerPath | | /dev/emcpowera |
| HDLM | | /dev/sddlmab |

As a result of the multiple device nodes, LVM tools find the same metadata multiple times and report them as duplicates.

125.7.2. Cases of duplicate PV warnings

LVM displays the duplicate PV warnings in either of the following cases:

- The two devices displayed in the output are both single paths to the same device.
- The two devices displayed in the output are both multipath maps.

Single paths to the same device

The following example shows a duplicate PV warning in which the duplicate devices are both single paths to the same device.

Found duplicate PV GDjTZf7Y03GJHjteqOwrye2dcSCjdaUi: using /dev/sdd not /dev/sdf

If you list the current DM Multipath topology using the **multipath -ll** command, you can find both **/dev/sdd** and **/dev/sdf** under the same multipath map.

These duplicate messages are only warnings and do not mean that the LVM operation has failed. Rather, they are alerting you that LVM uses only one of the devices as a physical volume and ignores the others.

If the messages indicate that LVM chooses the incorrect device or if the warnings are disruptive to users, you can apply a filter. The filter configures LVM to search only the necessary devices for physical volumes, and to leave out any underlying paths to multipath devices. As a result, the warnings no longer appear.

Multipath maps

The following examples show a duplicate PV warning for two devices that are both multipath maps. The duplicate physical volumes are located on two different devices rather than on two different paths to the same device.

Found duplicate PV GDjTZf7Y03GJHjteqOwrye2dcSCjdaUi: using /dev/mapper/mpatha not /dev/mapper/mpathc

Found duplicate PV GDjTZf7Y03GJHjteqOwrye2dcSCjdaUi: using /dev/emcpowera not /dev/emcpowerh

This situation is more serious than duplicate warnings for devices that are both single paths to the same device. These warnings often mean that the machine is accessing devices that it should not access: for example, LUN clones or mirrors.

Unless you clearly know which devices you should remove from the machine, this situation might be unrecoverable. Red Hat recommends that you contact Red Hat Technical Support to address this issue.

125.7.3. The LVM device filter

LVM tools scan for devices in the **/dev** directory and check every device there for LVM metadata. A filter in the **/etc/lvm/lvm.conf** file controls which devices LVM scans.

The filter is a list of patterns that LVM applies to each device found by a scan of the **/dev** directory, or the directory specified by the **dir** keyword in the **/etc/lvm/lvm.conf** file. Patterns are regular expressions delimited by any character and preceded by **a** for *accept* or **r** for *reject*. The first regular expression in the list that matches a device determines if LVM accepts or rejects (ignores) the device. LVM accepts devices that do not match any patterns.

The following is the default configuration of the filter, which scans all devices:

filter = ["a/.*/"]

125.7.4. Example LVM device filters that prevent duplicate PV warnings

The following examples show LVM device filters that avoid the duplicate physical volume warnings that are caused by multiple storage paths to a single logical unit (LUN).

The filter that you configure must include all devices that LVM needs to be check for metadata, such as the local hard drive with the root volume group on it and any multipathed devices. By rejecting the underlying paths to a multipath device (such as **/dev/sdb**, **/dev/sdd**, and so on), you can avoid these duplicate PV warnings, because LVM finds each unique metadata area once on the multipath device itself.

- This filter accepts the second partition on the first hard drive and any DM Multipath devices, but rejects everything else:

```
filter = [ "a|/dev/sda2$", "a|/dev/mapper/mpath.*|", "r|.*|"]
```

- This filter accepts all HP SmartArray controllers and any EMC PowerPath devices:

```
filter = [ "a|/dev/cciss.*|", "a|/dev/emcpower.*|", "r|.*|"]
```

- This filter accepts any partitions on the first IDE drive and any multipath devices:

```
filter = [ "a|/dev/hda.*|", "a|/dev/mapper/mpath.*|", "r|.*|"]
```

125.7.5. Applying an LVM device filter configuration

This procedure changes the configuration of the LVM device filter, which controls the devices that LVM scans.

Prerequisites

- Prepare the device filter pattern that you want to use.

Procedure

1. Test your device filter pattern without modifying the **/etc/lvm/lvm.conf** file.

Use an LVM command with the **--config 'devices{ filter = [your device filter pattern] }'** option. For example:

```
# lvs --config 'devices{ filter = [ "a|/dev/emcpower.*|", "r|.*|"] }'
```

2. Edit the **filter** option in the **/etc/lvm/lvm.conf** configuration file to use your new device filter pattern.

3. Check that no physical volumes or volume groups that you want to use are missing with the new configuration:

```
# pvscan
```

```
# vgscan
```

4. Rebuild the **initramfs** file system so that LVM scans only the necessary devices upon reboot:

```
# dracut --force --verbose
```

125.7.6. Additional resources

- Controlling LVM device scanning

PART IX. VIRTUALIZATION ON ARM 64 SYSTEMS

CHAPTER 126. GETTING STARTED WITH VIRTUALIZATION ON ARM 64

When using RHEL 8 on ARM 64 hardware with specific Red Hat subscriptions, it is possible to use KVM virtualization. However, when [enabling the KVM hypervisor](#) on your system, extra steps are needed compared to virtualization on AMD64 and Intel 64 architectures. Certain RHEL 8 virtualization features also have [different or restricted functionality](#) on ARM 64.



IMPORTANT

KVM virtualization on ARM 64 systems is only supported with specific Red Hat subscriptions. If not supported on your subscription, Red Hat strongly recommends not using virtualization on ARM 64 in production environments. For more information about your subscription, contact Red Hat Customer Services.

Apart from the information in the following sections, using virtualization on ARM 64 works the same as on AMD64 and Intel 64. Therefore, you can refer to other [RHEL 8 virtualization documentation](#) for more information when using virtualization on ARM 64.

126.1. ENABLING VIRTUALIZATION ON ARM 64

To set up a KVM hypervisor in order to create virtual machines (VMs) on an ARM 64 system running RHEL 8, follow the instructions below.

Prerequisites

- RHEL 8 is installed and registered on your host machine.
- The following minimum system resources are available:
 - 6 GB free disk space for the host, plus another 6 GB for each intended guest.
 - 4 GB of RAM for the host, plus another 4 GB for each intended guest.

Procedure

1. Enable the Advanced Virtualization module and disable the standard virtualization module.

```
# yum module disable virt:rhel
# yum module enable virt:8.2
```

2. Install the **qemu-kvm**, **libvirt**, and **virt-install** packages.

```
# yum install qemu-kvm libvirt virt-install
```

3. Start the **libvirtd** service.

```
# systemctl start libvirtd
```

4. Verify that your system is prepared to be a virtualization host:

```
# virt-host-validate
```

```
[...]
QEMU: Checking if device /dev/vhost-net exists : PASS
QEMU: Checking if device /dev/net/tun exists : PASS
QEMU: Checking for cgroup 'memory' controller support : PASS
QEMU: Checking for cgroup 'memory' controller mount-point : PASS
[...]
QEMU: Checking for cgroup 'blkio' controller support : PASS
QEMU: Checking for cgroup 'blkio' controller mount-point : PASS
QEMU: Checking if IOMMU is enabled by kernel : WARN (Unknown if this
platform has IOMMU support)
```

5. If all **virt-host-validate** checks return a **PASS** value, your system is prepared for [creating virtual machines](#).

If any of the checks return a **FAIL** value, follow the displayed instructions to fix the problem.

If any of the checks return a **WARN** value, consider following the displayed instructions to improve virtualization capabilities.

Additional information

- Note that if virtualization is not supported by your host CPU, **virt-host-validate** generates the following output:

```
QEMU: Checking for hardware virtualization: FAIL (Only emulated CPUs are available,
performance will be significantly limited)
```

However, attempting to create VMs on such a host system will fail, rather than have performance problems.

126.2. HOW VIRTUALIZATION ON ARM 64 DIFFERS FROM AMD64 AND INTEL 64

KVM virtualization in RHEL 8 on ARM 64 systems is different from KVM on AMD64 and Intel 64 systems in a number of aspects, notably:

Support

KVM virtualization on ARM 64 systems is only supported with specific Red Hat subscriptions. If not supported on your subscription, Red Hat strongly recommends not using virtualization on ARM 64 in production environments. For more information about your subscription, contact Red Hat Customer Services.

Guest operating systems

The only guest operating system currently supported on ARM 64 virtual machines (VMs) is RHEL 8, with kernel version 4.18 or later.

Display protocols

The SPICE protocol is not supported on ARM 64 systems. To display the graphical output of a VM, use the VNC protocol. In addition, only the **virtio-gpu-pci** virtual graphics card device is available.

Web console management

Some features of [VM management in the RHEL 8 web console](#) may not work correctly on ARM 64 hardware.

SecureBoot

The SecureBoot feature is not available on ARM 64 systems.

PXE

Booting in the Preboot Execution Environment (PXE) is only possible with the **virtio-net-pci** network interface controller (NIC). In addition, the built-in **VirtioNetDxe** driver of the virtual machine UEFI platform firmware (installed with the **edk2-aarch64** package) needs to be used for PXE booting. Note that iPXE option ROMs are not supported.

Device memory

Device memory features, such as the dual in-line memory module (DIMM) and non-volatile DIMM (NVDIMM), do not work on ARM 64.

pvpanic

The pvpanic device is currently not functional on ARM 64. Make sure to remove the **<panic>** element from the **<devices>** section of the guest XML configuration on ARM 64, as its presence can lead to the VM failing to boot.

OVMF

VMs on an ARM 64 host cannot use the OVMF UEFI firmware used on AMD64 and Intel 64, included in the **edk2-ovmf** package. Instead, these VMs use UEFI firmware included in the **edk2-aarch64** package, which provides a similar interface and implements a similar set of features.

Specifically, **edk2-aarch64** provides a built-in UEFI shell, but does not support the following functionality:

- SecureBoot
- Management Mode
- TPM-2.0 support

kvm-clock

The **kvm-clock** service does not have to be configured for time management in VMs on ARM 64.

Peripheral devices

ARM 64 systems do not support all the peripheral devices that are supported on AMD64 and Intel 64 systems. In some cases, the device functionality is not supported at all, and in other cases, a different device is supported for the same functionality.

Serial console configuration

When [setting up a serial console on a VM](#), it is not necessary to add the **console=ttyS0** parameter to the **/etc/default/grub** file.

Non-maskable interrupts

Sending non-maskable interrupts (NMIs) to an ARM 64 VM is currently not possible.

Nested virtualization

Creating nested VMs is currently not possible on ARM 64 hosts.

v2v and p2v

The **virt-v2v** and **virt-p2v** utilities are only supported on the AMD64 and Intel 64 architecture and are, therefore, not provided on ARM 64.

126.3. RELATED INFORMATION

- For more information about using virtualization on RHEL 8, see the [Configuring and managing virtualization](#) document.

CHAPTER 127. CONFIGURING THE SCALABLE VECTOR EXTENSION ON ARM 64 VIRTUAL MACHINES

Using a RHEL 8.2 or later hypervisor on the ARM 64 architecture provides the **Scalable Vector Extension** (SVE) functionality for your virtual machines (VM). This enables faster computation of vector mathematics and faster string operations in these VMs.

The base-line level of SVE is enabled by default on host CPUs that support it. However, Red Hat recommends configuring each vector length explicitly. This ensures that the VM can only be launched on or migrated to compatible hosts.



WARNING

This feature is currently only supported with specific subscriptions. For more information about your subscription, contact Red Hat Customer Services.

Prerequisites

- Your CPU must have the SVE feature. To verify:

```
$ grep -m 1 Features /proc/cpuinfo | grep -w sve
Features      : fp asimd evtstrm aes pmull sha1 sha2 crc32 atomics fphp asimdhp cpuid
asimdrdm fcma dcpop sve
```

If the output of this command includes **sve** or if its exit code is **0**, your CPU supports SVE.

Procedure

1. Open the XML configuration of the VM you want to modify.

```
# virsh edit vm-name
```

2. Edit the `<cpu>` element similarly to the following.

```
<cpu mode='host-passthrough' check='none'>
  <feature policy='require' name='sve' />
  <feature policy='require' name='sve128' />
  <feature policy='require' name='sve256' />
  <feature policy='disable' name='sve384' />
  <feature policy='require' name='sve512' />
</cpu>
```

This example explicitly enables SVE vector lengths 128, 256, and 512, and explicitly disables vector length 384.

CHAPTER 128. SHARING FILES BETWEEN THE HOST AND ITS VIRTUAL MACHINES USING VIRTIOFS

When using RHEL 8.2 or later as the hypervisor, you can efficiently share files between your host system and its virtual machines (VM) using the **virtiofs** feature.



WARNING

This feature is currently only supported with specific subscriptions. For more information about your subscription, contact Red Hat Customer Services.

Prerequisites

- Virtualization must be [installed and enabled](#) on your system.
- A directory that you want to share with your VMs. If you do not want to share any of your existing directories, create a new one, for example named *shared-files*.

mkdir shared-files

Procedure

- For each directory that you want to share with your VM, set it as a virtiofs file system in the VM's XML configuration.
 - Open the XML configuration of the intended VM.

virsh edit *vm-name*

- Add an entry similar to the following to the **<devices>** section of the VM's XML configuration.

```
<filesystem type='mount' accessmode='passthrough'>
  <driver type='virtiofs' />
  <binary path='/usr/libexec/virtiofsd' xattr='on' />
  <source dir='/root/shared-files' />
  <target dir='host-file-share' />
</filesystem>
```

This example sets the **/root/shared-files** directory on the host to be visible as **host-file-share** to the VM.

- Add a NUMA topology for shared memory to the XML configuration. The following example adds a basic topology for all CPUs and all RAM.

```
<cpu mode='host-passthrough' check='none'>
  <numa>
    <cell id='0' cpus='0-{number-vcpus - 1}' memory='{ram-amount-KiB}' unit='KiB'
```

```
memAccess='shared'>
</numa>
</cpu>
```

3. Add shared memory backing to the **<domain>** section of the XML configuration:

```
<domain>
[...]
<memoryBacking>
<access mode='shared' />
</memoryBacking>
[...]
</domain>
```

4. Edit the **/etc/libvirt/qemu.conf** file and add the following line:

```
memory_backing_dir = "/dev/shm"
```

5. Boot up the VM.

```
# virsh start vm-name
```

6. Mount the file-system in the guest operating system (OS). The following example mounts the previously configured **host-file-share** directory with a Linux guest OS.

```
# mount -t virtiofs host-file-share /mnt
```

Verification

- Ensure that the shared directory became accessible on the VM and that you can now open files stored in the directory.

Known issues and limitations

- File-system mount options related to access time, such as **noatime** and **strictatime**, are not likely to work with virtiofs, and Red Hat discourages their use.