



# 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 © 2023 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<sup>®</sup> is the registered trademark of Linus Torvalds in the United States and other countries.

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

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

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

Node.js<sup>®</sup> 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<sup>®</sup> 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

<b>MAKING OPEN SOURCE MORE INCLUSIVE</b> .....	<b>30</b>
<b>PROVIDING FEEDBACK ON RED HAT DOCUMENTATION</b> .....	<b>31</b>
<b>PART I. DESIGN OF INSTALLATION</b> .....	<b>32</b>
<b>CHAPTER 1. SUPPORTED RHEL ARCHITECTURES AND SYSTEM REQUIREMENTS</b> .....	<b>33</b>
1.1. SUPPORTED ARCHITECTURES	33
1.2. SYSTEM REQUIREMENTS	33
<b>CHAPTER 2. PREPARING FOR YOUR INSTALLATION</b> .....	<b>34</b>
2.1. RECOMMENDED STEPS	34
2.2. RHEL INSTALLATION METHODS	34
2.3. SYSTEM REQUIREMENTS	35
2.4. INSTALLATION BOOT MEDIA OPTIONS	36
2.5. TYPES OF INSTALLATION ISO IMAGES	36
2.6. DOWNLOADING A RHEL INSTALLATION ISO IMAGE	37
2.6.1. Types of installation ISO images	37
2.6.2. Downloading an ISO image from the Customer Portal	38
2.6.3. Downloading an ISO image using curl	39
2.7. CREATING A BOOTABLE INSTALLATION MEDIUM FOR RHEL	40
2.7.1. Installation boot media options	40
2.7.2. Creating a bootable DVD or CD	41
2.7.3. Creating a bootable USB device on Linux	41
2.7.4. Creating a bootable USB device on Windows	42
2.7.5. Creating a bootable USB device on Mac OS X	43
2.8. PREPARING AN INSTALLATION SOURCE	45
2.8.1. Types of installation source	45
2.8.2. Specify the installation source	46
2.8.3. Ports for network-based installation	46
2.8.4. Creating an installation source on an NFS server	47
2.8.5. Creating an installation source using HTTP or HTTPS	48
2.8.6. Creating an installation source using FTP	50
2.8.7. Preparing a hard drive as an installation source	52
<b>CHAPTER 3. GETTING STARTED</b> .....	<b>54</b>
3.1. BOOTING THE INSTALLATION	54
3.1.1. Boot menu	54
3.1.2. Types of boot options	55
3.1.3. Editing the boot: prompt in BIOS	56
3.1.4. Editing predefined boot options using the > prompt	56
3.1.5. Editing the GRUB2 menu for the UEFI-based systems	57
3.1.6. Booting the installation from a USB, CD, or DVD	57
3.1.7. Booting the installation from a network using PXE	58
3.2. INSTALLING RHEL USING AN ISO IMAGE FROM THE CUSTOMER PORTAL	59
3.3. REGISTERING AND INSTALLING RHEL FROM THE CDN USING THE GUI	61
3.3.1. What is the Content Delivery Network	61
3.3.2. Registering and installing RHEL from the CDN	62
3.3.2.1. Installation source repository after system registration	64
3.3.3. Verifying your system registration from the CDN	65
3.3.4. Unregistering your system from the CDN	66
3.4. COMPLETING THE INSTALLATION	67

<b>CHAPTER 4. CUSTOMIZING YOUR INSTALLATION</b> .....	<b>68</b>
4.1. CONFIGURING LANGUAGE AND LOCATION SETTINGS	68
4.2. CONFIGURING LOCALIZATION OPTIONS	69
4.3. CONFIGURING SYSTEM OPTIONS	71
4.3.1. Configuring installation destination	71
4.3.2. Configuring boot loader	75
4.3.3. Configuring Kdump	76
4.3.4. Configuring network and host name options	76
4.3.4.1. Adding a virtual network interface	78
4.3.4.2. Editing network interface configuration	79
4.3.4.3. Enabling or Disabling the Interface Connection	79
4.3.4.4. Setting up Static IPv4 or IPv6 Settings	80
4.3.4.5. Configuring Routes	81
4.3.4.6. Additional resources	81
4.3.5. Configuring Connect to Red Hat	81
4.3.5.1. Introduction to System Purpose	82
4.3.5.2. Configuring Connect to Red Hat options	83
4.3.5.3. Installation source repository after system registration	84
4.3.5.4. Verifying your system registration from the CDN	85
4.3.5.5. Unregistering your system from the CDN	85
4.3.5.6. Additional resources	87
4.3.6. Installing System Aligned with a Security Policy	87
4.3.6.1. About security policy	87
4.3.6.2. Configuring a security policy	87
4.3.6.3. Additional resources	88
4.4. CONFIGURING SOFTWARE SETTINGS	88
4.4.1. Configuring installation source	88
4.4.2. Configuring software selection	90
4.5. CONFIGURING STORAGE DEVICES	92
4.5.1. Storage device selection	92
4.5.2. Filtering storage devices	93
4.5.3. Using advanced storage options	94
4.5.3.1. Discovering and starting an iSCSI session	94
4.5.3.2. Configuring FCoE parameters	96
4.5.3.3. Configuring DASD storage devices	97
4.5.3.4. Configuring FCP devices	97
4.5.4. Installing to an NVDIMM device	98
4.5.4.1. Criteria for using an NVDIMM device as an installation target	98
4.5.4.2. Configuring an NVDIMM device using the graphical installation mode	99
4.6. CONFIGURING MANUAL PARTITIONING	100
4.6.1. Starting manual partitioning	100
4.6.2. Adding a mount point file system	102
4.6.3. Configuring storage for a mount point file system	102
4.6.4. Customizing a mount point file system	103
4.6.5. Preserving the /home directory	105
4.6.6. Creating a software RAID during the installation	107
4.6.7. Creating an LVM logical volume	108
4.6.8. Configuring an LVM logical volume	109
4.7. CONFIGURING A ROOT PASSWORD	110
4.8. CREATING A USER ACCOUNT	111
4.9. EDITING ADVANCED USER SETTINGS	112
<b>CHAPTER 5. COMPLETING POST-INSTALLATION TASKS</b> .....	<b>114</b>

5.1. COMPLETING INITIAL SETUP	114
5.2. REGISTERING YOUR SYSTEM USING THE COMMAND LINE	116
5.3. REGISTERING YOUR SYSTEM USING THE SUBSCRIPTION MANAGER USER INTERFACE	117
5.4. REGISTERING RHEL 8 USING THE INSTALLER GUI	118
5.5. REGISTRATION ASSISTANT	119
5.6. CONFIGURING SYSTEM PURPOSE USING THE SUBSCRIPTION-MANAGER COMMAND-LINE TOOL	119
5.7. SECURING YOUR SYSTEM	121
5.8. DEPLOYING SYSTEMS THAT ARE COMPLIANT WITH A SECURITY PROFILE IMMEDIATELY AFTER AN INSTALLATION	122
5.8.1. Profiles not compatible with Server with GUI	122
5.8.2. Deploying baseline-compliant RHEL systems using the graphical installation	123
5.8.3. Deploying baseline-compliant RHEL systems using Kickstart	124
5.9. NEXT STEPS	125
<b>APPENDIX A. TROUBLESHOOTING</b>	<b>126</b>
<b>APPENDIX B. TOOLS AND TIPS FOR TROUBLESHOOTING AND BUG REPORTING</b>	<b>127</b>
B.1. Dracut	127
B.2. Using installation log files	127
B.2.1. Creating pre-installation log files	127
B.2.2. Transferring installation log files to a USB drive	128
B.2.3. Transferring installation log files over the network	129
B.3. Detecting memory faults using the Memtest86 application	130
B.3.1. Running Memtest86	130
B.4. Verifying boot media	131
B.5. Consoles and logging during installation	131
B.6. Saving screenshots	132
B.7. Display settings and device drivers	132
B.8. Reporting error messages to Red Hat Customer Support	133
A.1. TROUBLESHOOTING DURING THE INSTALLATION	134
A.1.1. Disks are not detected	134
A.1.2. Reporting error messages to Red Hat Customer Support	135
A.1.3. Partitioning issues for IBM Power Systems	136
<b>APPENDIX C. TROUBLESHOOTING</b>	<b>137</b>
C.1. Resuming an interrupted download attempt	137
C.2. Disks are not detected	137
C.3. Cannot boot with a RAID card	138
C.4. Graphical boot sequence is not responding	138
C.5. X server fails after log in	139
C.6. RAM is not recognized	139
C.7. System is displaying signal 11 errors	140
C.8. Unable to IPL from network storage space	141
C.9. Using XDMCP	141
C.10. Using rescue mode	142
C.10.1. Booting into rescue mode	143
C.10.2. Using an SOS report in rescue mode	144
C.10.3. Reinstalling the GRUB2 boot loader	145
C.10.4. Using RPM to add or remove a driver	146
C.10.4.1. Adding a driver using RPM	146
C.10.4.2. Removing a driver using RPM	147
C.11. ip= boot option returns an error	148
C.12. Cannot boot into the graphical installation on iLO or iDRAC devices	148
C.13. Rootfs image is not initramfs	149

<b>APPENDIX D. SYSTEM REQUIREMENTS REFERENCE</b> .....	<b>151</b>
D.1. HARDWARE COMPATIBILITY	151
D.2. SUPPORTED INSTALLATION TARGETS	151
D.3. SYSTEM SPECIFICATIONS	151
D.4. DISK AND MEMORY REQUIREMENTS	152
D.5. UEFI SECURE BOOT AND BETA RELEASE REQUIREMENTS	153
<b>APPENDIX E. PARTITIONING REFERENCE</b> .....	<b>154</b>
E.1. SUPPORTED DEVICE TYPES	154
E.2. SUPPORTED FILE SYSTEMS	154
E.3. SUPPORTED RAID TYPES	155
E.4. RECOMMENDED PARTITIONING SCHEME	156
E.5. ADVICE ON PARTITIONS	158
E.6. SUPPORTED HARDWARE STORAGE	160
<b>APPENDIX F. BOOT OPTIONS REFERENCE</b> .....	<b>162</b>
F.1. INSTALLATION SOURCE BOOT OPTIONS	162
F.2. NETWORK BOOT OPTIONS	166
Configuration methods for the automatic interface	167
F.3. CONSOLE BOOT OPTIONS	169
F.4. DEBUG BOOT OPTIONS	171
F.5. STORAGE BOOT OPTIONS	173
F.6. DEPRECATED BOOT OPTIONS	174
F.7. REMOVED BOOT OPTIONS	175
<b>APPENDIX G. CHANGING A SUBSCRIPTION SERVICE</b> .....	<b>177</b>
G.1. UNREGISTERING FROM SUBSCRIPTION MANAGEMENT SERVER	177
G.1.1. Unregistering using command line	177
G.1.2. Unregistering using Subscription Manager user interface	178
G.2. UNREGISTERING FROM SATELLITE SERVER	178
<b>APPENDIX H. ISCSI DISKS IN INSTALLATION PROGRAM</b> .....	<b>179</b>
<b>CHAPTER 6. BOOTING A BETA SYSTEM WITH UEFI SECURE BOOT</b> .....	<b>180</b>
6.1. UEFI SECURE BOOT AND RHEL BETA RELEASES	180
6.2. ADDING A BETA PUBLIC KEY FOR UEFI SECURE BOOT	180
6.3. REMOVING A BETA PUBLIC KEY	181
<b>CHAPTER 7. COMPOSING A CUSTOMIZED RHEL SYSTEM IMAGE</b> .....	<b>182</b>
7.1. IMAGE BUILDER DESCRIPTION	182
7.1.1. What is image builder?	182
7.1.2. Image builder terminology	182
7.1.3. Image builder output formats	182
7.1.4. Image builder system requirements	183
7.2. INSTALLING IMAGE BUILDER	184
7.2.1. Image builder system requirements	184
7.2.2. Installing image builder in a virtual machine	185
7.2.3. Reverting to lorax-composer image builder backend	186
7.3. CREATING SYSTEM IMAGES USING THE IMAGE BUILDER COMMAND-LINE INTERFACE	187
7.3.1. Introducing the image builder command-line interface	187
7.3.2. Creating an image builder blueprint using the command-line interface	187
7.3.3. Editing an image builder blueprint with command-line interface	189
7.3.4. Creating a system image with image builder in the command-line interface	190
7.3.5. Basic image builder command-line commands	192
7.3.6. Image builder blueprint format	193



7.3.7. Supported image customizations	194
7.3.8. Packages installed by image builder	199
7.3.9. Enabled services on custom images	202
7.4. CREATING SYSTEM IMAGES USING THE IMAGE BUILDER WEB CONSOLE INTERFACE	203
7.4.1. Accessing the image builder GUI in the RHEL web console	203
7.4.2. Creating an image builder blueprint in the web console interface	204
7.4.3. Creating a system image using image builder in the web console interface	204
7.5. PREPARING AND UPLOADING CLOUD IMAGES USING IMAGE BUILDER	206
7.5.1. Preparing to upload AWS AMI images	206
7.5.2. Uploading an AMI image to AWS using the CLI	207
7.5.3. Pushing images to AWS Cloud AMI	208
7.5.4. Preparing to upload Microsoft Azure VHD images	210
7.5.5. Uploading VHD images to Microsoft Azure cloud	212
7.5.6. Uploading VMDK images and creating a RHEL virtual machine in vSphere	213
7.5.7. Uploading images to GCP with image builder	214
7.5.7.1. Uploading a gce image to GCP using the CLI	214
7.5.7.2. Authenticating with GCP	215
7.5.7.2.1. Specifying credentials with the composer-cli command	216
7.5.7.2.2. Specifying credentials in the osbuild-composer worker configuration	216
7.5.8. Pushing VMDK images to vSphere using the GUI image builder tool	216
7.5.9. Pushing VHD images to Microsoft Azure cloud using the GUI image builder tool	219
7.5.10. Uploading QCOW2 images to OpenStack	221
7.5.11. Preparing to upload customized RHEL images to Alibaba	224
7.5.12. Uploading customized RHEL images to Alibaba	225
7.5.13. Importing images to Alibaba	226
7.5.14. Creating an instance of a customized RHEL image using Alibaba	227
<b>CHAPTER 8. PERFORMING AN AUTOMATED INSTALLATION USING KICKSTART</b> .....	<b>229</b>
8.1. KICKSTART INSTALLATION BASICS	229
8.1.1. What are Kickstart installations	229
8.1.2. Automated installation workflow	229
8.2. CREATING KICKSTART FILES	230
8.2.1. Creating a Kickstart file with the Kickstart configuration tool	230
8.2.2. Creating a Kickstart file by performing a manual installation	231
8.2.3. Converting a Kickstart file from previous RHEL installation	231
8.2.4. Creating a custom image using Image Builder	231
8.3. MAKING KICKSTART FILES AVAILABLE TO THE INSTALLATION PROGRAM	232
8.3.1. Ports for network-based installation	232
8.3.2. Making a Kickstart file available on an NFS server	232
8.3.3. Making a Kickstart file available on an HTTP or HTTPS server	233
8.3.4. Making a Kickstart file available on an FTP server	235
8.3.5. Making a Kickstart file available on a local volume	236
8.3.6. Making a Kickstart file available on a local volume for automatic loading	237
8.4. CREATING INSTALLATION SOURCES FOR KICKSTART INSTALLATIONS	238
8.4.1. Types of installation source	238
8.4.2. Ports for network-based installation	239
8.4.3. Creating an installation source on an NFS server	239
8.4.4. Creating an installation source using HTTP or HTTPS	240
8.4.5. Creating an installation source using FTP	242
8.5. STARTING KICKSTART INSTALLATIONS	244
8.5.1. Starting a Kickstart installation manually	244
8.5.2. Starting a Kickstart installation automatically using PXE	245
8.5.3. Starting a Kickstart installation automatically using a local volume	246

8.6. CONSOLES AND LOGGING DURING INSTALLATION	247
8.7. MAINTAINING KICKSTART FILES	248
8.7.1. Installing Kickstart maintenance tools	248
8.7.2. Verifying a Kickstart file	248
8.8. REGISTERING AND INSTALLING RHEL FROM THE CDN USING KICKSTART	249
8.8.1. Registering and installing RHEL from the CDN	249
8.8.2. Verifying your system registration from the CDN	251
8.8.3. Unregistering your system from the CDN	252
8.9. PERFORMING A REMOTE RHEL INSTALLATION USING VNC	252
8.9.1. Overview	252
8.9.2. Considerations	253
8.9.3. Performing a remote RHEL installation in VNC Direct mode	254
8.9.4. Performing a remote RHEL installation in VNC Connect mode	254
<b>CHAPTER 9. ADVANCED CONFIGURATION OPTIONS</b>	<b>256</b>
9.1. CONFIGURING SYSTEM PURPOSE	256
9.1.1. Overview	256
9.1.2. Configuring System Purpose in a Kickstart file	257
9.1.3. Additional resources	258
9.2. UPDATING DRIVERS DURING INSTALLATION	258
9.2.1. Overview	258
9.2.2. Types of driver update	259
9.2.3. Preparing a driver update	259
9.2.4. Performing an automatic driver update	260
9.2.5. Performing an assisted driver update	260
9.2.6. Performing a manual driver update	261
9.2.7. Disabling a driver	262
9.3. PREPARING TO INSTALL FROM THE NETWORK USING PXE	262
9.3.1. Network install overview	262
9.3.2. Configuring a TFTP server for BIOS-based clients	263
9.3.3. Configuring a TFTP server for UEFI-based clients	266
9.3.4. Configuring a network server for IBM Power systems	269
9.4. BOOT OPTIONS	271
9.4.1. Types of boot options	271
9.4.2. Editing boot options	271
9.4.2.1. Editing the boot: prompt in BIOS	271
9.4.2.2. Editing predefined boot options using the > prompt	272
9.4.2.3. Editing the GRUB2 menu for the UEFI-based systems	272
9.4.3. Installation source boot options	273
9.4.4. Network boot options	277
Configuration methods for the automatic interface	278
9.4.5. Console boot options	280
9.4.6. Debug boot options	282
9.4.7. Storage boot options	284
9.4.8. Kickstart boot options	285
9.4.9. Advanced installation boot options	286
9.4.10. Deprecated boot options	287
9.4.11. Removed boot options	288
<b>CHAPTER 10. KICKSTART REFERENCES</b>	<b>290</b>
<b>APPENDIX I. KICKSTART SCRIPT FILE FORMAT REFERENCE</b>	<b>291</b>
I.1. KICKSTART FILE FORMAT	291
I.2. PACKAGE SELECTION IN KICKSTART	292

I.2.1. Package selection section	292
I.2.2. Package selection commands	292
I.2.3. Common package selection options	294
I.2.4. Options for specific package groups	296
I.3. SCRIPTS IN KICKSTART FILE	296
I.3.1. %pre script	297
I.3.1.1. %pre script section options	297
I.3.2. %pre-install script	298
I.3.2.1. %pre-install script section options	298
I.3.3. %post script	299
I.3.3.1. %post script section options	299
I.3.3.2. Example: Mounting NFS in a post-install script	300
I.3.3.3. Example: Running subscription-manager as a post-install script	300
I.4. ANACONDA CONFIGURATION SECTION	301
I.5. KICKSTART ERROR HANDLING SECTION	301
I.6. KICKSTART ADD-ON SECTIONS	302
<b>APPENDIX J. KICKSTART COMMANDS AND OPTIONS REFERENCE .....</b>	<b>303</b>
J.1. KICKSTART CHANGES	303
J.1.1. Deprecated Kickstart commands and options	303
J.1.2. Removed Kickstart commands and options	304
J.2. KICKSTART COMMANDS FOR INSTALLATION PROGRAM CONFIGURATION AND FLOW CONTROL	304
J.2.1. cdrom	305
J.2.2. cmdline	305
J.2.3. driverdisk	305
J.2.4. eula	306
J.2.5. firstboot	306
J.2.6. graphical	307
J.2.7. halt	307
J.2.8. harddrive	308
J.2.9. install (deprecated)	308
J.2.10. liveimg	309
J.2.11. logging	310
J.2.12. mediacheck	310
J.2.13. nfs	310
J.2.14. ostreesetup	311
J.2.15. poweroff	311
J.2.16. reboot	312
J.2.17. rhsm	313
J.2.18. shutdown	313
J.2.19. sshpw	314
J.2.20. text	314
J.2.21. url	315
J.2.22. vnc	316
J.2.23. %include	316
J.2.24. %ksappend	317
J.3. KICKSTART COMMANDS FOR SYSTEM CONFIGURATION	317
J.3.1. auth or authconfig (deprecated)	317
J.3.2. authselect	318
J.3.3. firewall	318
J.3.4. group	319
J.3.5. keyboard (required)	319

J.3.6. lang (required)	320
J.3.7. module	321
J.3.8. repo	321
J.3.9. rootpw (required)	322
J.3.10. selinux	323
J.3.11. services	324
J.3.12. skipx	324
J.3.13. sshkey	324
J.3.14. syspurpose	325
J.3.15. timezone (required)	326
J.3.16. user	327
J.3.17. xconfig	328
J.4. KICKSTART COMMANDS FOR NETWORK CONFIGURATION	328
J.4.1. network (optional)	328
J.4.2. realm	333
J.5. KICKSTART COMMANDS FOR HANDLING STORAGE	333
J.5.1. device (deprecated)	333
J.5.2. autopart	334
J.5.3. bootloader (required)	336
J.5.4. zipl	339
J.5.5. clearpart	339
J.5.6. fcoe	341
J.5.7. ignoredisk	341
J.5.8. iscsi	342
J.5.9. iscsiname	343
J.5.10. logvol	344
J.5.11. mount	348
J.5.12. nvdim	349
J.5.13. part or partition	350
J.5.14. raid	354
J.5.15. reqpart	357
J.5.16. snapshot	358
J.5.17. volgroup	358
J.5.18. zerombr	359
J.5.19. zfc	360
J.6. KICKSTART COMMANDS FOR ADDONS SUPPLIED WITH THE RHEL INSTALLATION PROGRAM	360
J.6.1. %addon com_redhat_kdump	360
J.6.2. %addon org_fedora_osc	361
J.7. COMMANDS USED IN ANACONDA	363
J.7.1. pwpolicy	363
J.8. KICKSTART COMMANDS FOR SYSTEM RECOVERY	364
J.8.1. rescue	364
<b>PART II. DESIGN OF SECURITY</b>	<b>367</b>
<b>CHAPTER 11. OVERVIEW OF SECURITY HARDENING IN RHEL</b>	<b>368</b>
11.1. WHAT IS COMPUTER SECURITY?	368
11.2. STANDARDIZING SECURITY	368
11.3. CRYPTOGRAPHIC SOFTWARE AND CERTIFICATIONS	368
11.4. SECURITY CONTROLS	369
11.4.1. Physical controls	369
11.4.2. Technical controls	369
11.4.3. Administrative controls	370

11.5. VULNERABILITY ASSESSMENT	370
11.5.1. Defining assessment and testing	370
11.5.2. Establishing a methodology for vulnerability assessment	372
11.5.3. Vulnerability assessment tools	372
11.6. SECURITY THREATS	372
11.6.1. Threats to network security	372
11.6.2. Threats to server security	373
11.6.3. Threats to workstation and home PC security	374
11.7. COMMON EXPLOITS AND ATTACKS	375
<b>CHAPTER 12. SECURING RHEL DURING INSTALLATION</b> .....	<b>379</b>
12.1. BIOS AND UEFI SECURITY	379
12.1.1. BIOS passwords	379
12.1.2. Non-BIOS-based systems security	379
12.2. DISK PARTITIONING	379
12.3. RESTRICTING NETWORK CONNECTIVITY DURING THE INSTALLATION PROCESS	380
12.4. INSTALLING THE MINIMUM AMOUNT OF PACKAGES REQUIRED	380
12.5. POST-INSTALLATION PROCEDURES	380
<b>CHAPTER 13. USING SYSTEM-WIDE CRYPTOGRAPHIC POLICIES</b> .....	<b>382</b>
13.1. SYSTEM-WIDE CRYPTOGRAPHIC POLICIES	382
Tool for managing crypto policies	383
Strong crypto defaults by removing insecure cipher suites and protocols	383
Cipher suites and protocols disabled in all policy levels	383
Cipher suites and protocols enabled in the crypto-policies levels	384
13.2. SWITCHING THE SYSTEM-WIDE CRYPTOGRAPHIC POLICY TO MODE COMPATIBLE WITH EARLIER RELEASES	385
13.3. SETTING UP SYSTEM-WIDE CRYPTOGRAPHIC POLICIES IN THE WEB CONSOLE	385
13.4. SWITCHING THE SYSTEM TO FIPS MODE	386
13.5. ENABLING FIPS MODE IN A CONTAINER	387
13.6. LIST OF RHEL APPLICATIONS USING CRYPTOGRAPHY THAT IS NOT COMPLIANT WITH FIPS 140-2	387
13.7. EXCLUDING AN APPLICATION FROM FOLLOWING SYSTEM-WIDE CRYPTO POLICIES	389
13.7.1. Examples of opting out of system-wide crypto policies	389
13.8. CUSTOMIZING SYSTEM-WIDE CRYPTOGRAPHIC POLICIES WITH SUBPOLICIES	390
13.9. DISABLING SHA-1 BY CUSTOMIZING A SYSTEM-WIDE CRYPTOGRAPHIC POLICY	392
13.10. CREATING AND SETTING A CUSTOM SYSTEM-WIDE CRYPTOGRAPHIC POLICY	392
13.11. ADDITIONAL RESOURCES	393
<b>CHAPTER 14. CONFIGURING APPLICATIONS TO USE CRYPTOGRAPHIC HARDWARE THROUGH PKCS #11</b> .	<b>394</b>
14.1. CRYPTOGRAPHIC HARDWARE SUPPORT THROUGH PKCS #11	394
14.2. USING SSH KEYS STORED ON A SMART CARD	394
14.3. CONFIGURING APPLICATIONS TO AUTHENTICATE USING CERTIFICATES FROM SMART CARDS	396
14.4. USING HSMS PROTECTING PRIVATE KEYS IN APACHE	396
14.5. USING HSMS PROTECTING PRIVATE KEYS IN NGINX	397
14.6. ADDITIONAL RESOURCES	397
<b>CHAPTER 15. USING SHARED SYSTEM CERTIFICATES</b> .....	<b>398</b>
15.1. THE SYSTEM-WIDE TRUST STORE	398
15.2. ADDING NEW CERTIFICATES	398
15.3. MANAGING TRUSTED SYSTEM CERTIFICATES	399
<b>CHAPTER 16. SCANNING THE SYSTEM FOR SECURITY COMPLIANCE AND VULNERABILITIES</b> .....	<b>401</b>
16.1. CONFIGURATION COMPLIANCE TOOLS IN RHEL	401

16.2. RED HAT SECURITY ADVISORIES OVAL FEED	401
16.3. VULNERABILITY SCANNING	402
16.3.1. Red Hat Security Advisories OVAL feed	403
16.3.2. Scanning the system for vulnerabilities	404
16.3.3. Scanning remote systems for vulnerabilities	404
16.4. CONFIGURATION COMPLIANCE SCANNING	405
16.4.1. Configuration compliance in RHEL	405
16.4.2. Possible results of an OpenSCAP scan	406
16.4.3. Viewing profiles for configuration compliance	406
16.4.4. Assessing configuration compliance with a specific baseline	408
16.5. REMEDIATING THE SYSTEM TO ALIGN WITH A SPECIFIC BASELINE	408
16.6. REMEDIATING THE SYSTEM TO ALIGN WITH A SPECIFIC BASELINE USING AN SSG ANSIBLE PLAYBOOK	409
16.7. CREATING A REMEDIATION ANSIBLE PLAYBOOK TO ALIGN THE SYSTEM WITH A SPECIFIC BASELINE	411
16.8. CREATING A REMEDIATION BASH SCRIPT FOR A LATER APPLICATION	411
16.9. SCANNING THE SYSTEM WITH A CUSTOMIZED PROFILE USING SCAP WORKBENCH	412
16.9.1. Using SCAP Workbench to scan and remediate the system	412
16.9.2. Customizing a security profile with SCAP Workbench	414
16.9.3. Additional resources	416
16.10. SCANNING CONTAINER AND CONTAINER IMAGES FOR VULNERABILITIES	416
16.11. ASSESSING SECURITY COMPLIANCE OF A CONTAINER OR A CONTAINER IMAGE WITH A SPECIFIC BASELINE	417
16.12. CHECKING INTEGRITY WITH AIDE	418
16.12.1. Installing AIDE	418
16.12.2. Performing integrity checks with AIDE	418
16.12.3. Updating an AIDE database	419
16.12.4. File-integrity tools: AIDE and IMA	419
16.12.5. Additional resources	420
16.13. ENCRYPTING BLOCK DEVICES USING LUKS	420
16.13.1. LUKS disk encryption	420
16.13.2. LUKS versions in RHEL	421
16.13.3. Options for data protection during LUKS2 re-encryption	422
16.13.4. Encrypting existing data on a block device using LUKS2	423
16.13.5. Encrypting existing data on a block device using LUKS2 with a detached header	425
16.13.6. Encrypting a blank block device using LUKS2	426
16.13.7. Creating a LUKS encrypted volume using the storage RHEL System Role	426
16.14. CONFIGURING AUTOMATED UNLOCKING OF ENCRYPTED VOLUMES USING POLICY-BASED DECRYPTION	428
16.14.1. Network-bound disk encryption	428
16.14.2. Installing an encryption client - Clevis	429
16.14.3. Deploying a Tang server with SELinux in enforcing mode	430
16.14.4. Rotating Tang server keys and updating bindings on clients	431
16.14.5. Configuring automated unlocking using a Tang key in the web console	433
16.14.6. Basic NBDE and TPM2 encryption-client operations	436
16.14.7. Configuring manual enrollment of LUKS-encrypted volumes	438
16.14.8. Configuring manual enrollment of LUKS-encrypted volumes using a TPM 2.0 policy	440
16.14.9. Removing a Clevis pin from a LUKS-encrypted volume manually	442
16.14.10. Configuring automated enrollment of LUKS-encrypted volumes using Kickstart	443
16.14.11. Configuring automated unlocking of a LUKS-encrypted removable storage device	444
16.14.12. Deploying high-availability NBDE systems	445
16.14.12.1. High-available NBDE using Shamir's Secret Sharing	445
16.14.12.1.1. Example 1: Redundancy with two Tang servers	445

16.14.12.1.2. Example 2: Shared secret on a Tang server and a TPM device	446
16.14.13. Deployment of virtual machines in a NBDE network	447
16.14.14. Building automatically-enrollable VM images for cloud environments using NBDE	447
16.14.15. Deploying Tang as a container	447
16.14.16. Introduction to the nbde_client and nbde_server System Roles (Clevis and Tang)	449
16.14.17. Using the nbde_server System Role for setting up multiple Tang servers	450
16.14.18. Using the nbde_client System Role for setting up multiple Clevis clients	451
<b>CHAPTER 17. USING SELINUX</b>	<b>453</b>
17.1. GETTING STARTED WITH SELINUX	453
17.1.1. Introduction to SELinux	453
17.1.2. Benefits of running SELinux	454
17.1.3. SELinux examples	455
17.1.4. SELinux architecture and packages	456
17.1.5. SELinux states and modes	457
17.2. CHANGING SELINUX STATES AND MODES	457
17.2.1. Permanent changes in SELinux states and modes	457
17.2.2. Changing to permissive mode	458
17.2.3. Changing to enforcing mode	459
17.2.4. Enabling SELinux on systems that previously had it disabled	460
17.2.5. Disabling SELinux	462
17.2.6. Changing SELinux modes at boot time	463
17.3. TROUBLESHOOTING PROBLEMS RELATED TO SELINUX	464
17.3.1. Identifying SELinux denials	464
17.3.2. Analyzing SELinux denial messages	465
17.3.3. Fixing analyzed SELinux denials	466
17.3.4. SELinux denials in the Audit log	469
17.3.5. Additional resources	470
<b>PART III. DESIGN OF NETWORK</b>	<b>471</b>
<b>CHAPTER 18. CONFIGURING IP NETWORKING WITH IFCFG FILES</b>	<b>472</b>
18.1. CONFIGURING AN INTERFACE WITH STATIC NETWORK SETTINGS USING IFCFG FILES	472
18.2. CONFIGURING AN INTERFACE WITH DYNAMIC NETWORK SETTINGS USING IFCFG FILES	473
18.3. MANAGING SYSTEM-WIDE AND PRIVATE CONNECTION PROFILES WITH IFCFG FILES	473
<b>CHAPTER 19. GETTING STARTED WITH IPVLAN</b>	<b>475</b>
19.1. IPVLAN MODES	475
19.2. COMPARISON OF IPVLAN AND MACVLAN	475
19.3. CREATING AND CONFIGURING THE IPVLAN DEVICE USING IPROUTE2	476
<b>CHAPTER 20. REUSING THE SAME IP ADDRESS ON DIFFERENT INTERFACES</b>	<b>478</b>
20.1. PERMANENTLY REUSING THE SAME IP ADDRESS ON DIFFERENT INTERFACES	478
20.2. TEMPORARILY REUSING THE SAME IP ADDRESS ON DIFFERENT INTERFACES	479
20.3. ADDITIONAL RESOURCES	481
<b>CHAPTER 21. SECURING NETWORKS</b>	<b>482</b>
21.1. USING SECURE COMMUNICATIONS BETWEEN TWO SYSTEMS WITH OPENSSSH	482
21.1.1. SSH and OpenSSH	482
21.1.2. Configuring and starting an OpenSSH server	483
21.1.3. Setting an OpenSSH server for key-based authentication	484
21.1.4. Generating SSH key pairs	485
21.1.5. Using SSH keys stored on a smart card	487
21.1.6. Making OpenSSH more secure	488
21.1.7. Connecting to a remote server using an SSH jump host	491

21.1.8. Connecting to remote machines with SSH keys using ssh-agent	492
21.1.9. Additional resources	493
21.2. PLANNING AND IMPLEMENTING TLS	493
21.2.1. SSL and TLS protocols	493
21.2.2. Security considerations for TLS in RHEL 8	494
21.2.2.1. Protocols	495
21.2.2.2. Cipher suites	495
21.2.2.3. Public key length	496
21.2.3. Hardening TLS configuration in applications	496
21.2.3.1. Configuring the Apache HTTP server to use TLS	496
21.2.3.2. Configuring the Nginx HTTP and proxy server to use TLS	497
21.2.3.3. Configuring the Dovecot mail server to use TLS	497
21.3. CONFIGURING A VPN WITH IPSEC	498
21.3.1. Libreswan as an IPsec VPN implementation	498
21.3.2. Authentication methods in Libreswan	499
21.3.3. Installing Libreswan	501
21.3.4. Creating a host-to-host VPN	501
21.3.5. Configuring a site-to-site VPN	502
21.3.6. Configuring a remote access VPN	503
21.3.7. Configuring a mesh VPN	504
21.3.8. Deploying a FIPS-compliant IPsec VPN	506
21.3.9. Protecting the IPsec NSS database by a password	509
21.3.10. Configuring an IPsec VPN to use TCP	510
21.3.11. Configuring automatic detection and usage of ESP hardware offload to accelerate an IPsec connection	511
21.3.12. Configuring ESP hardware offload on a bond to accelerate an IPsec connection	512
21.3.13. Configuring IPsec connections that opt out of the system-wide crypto policies	513
21.3.14. Troubleshooting IPsec VPN configurations	514
21.3.15. Additional resources	518
21.4. USING MACSEC TO ENCRYPT LAYER-2 TRAFFIC IN THE SAME PHYSICAL NETWORK	518
21.4.1. Configuring a MACsec connection using nmcli	519
21.4.2. Additional resources	521
21.5. USING AND CONFIGURING FIREWALLD	521
21.5.1. Getting started with firewalld	521
21.5.1.1. When to use firewalld, nftables, or iptables	521
21.5.1.2. Zones	521
21.5.1.3. Predefined services	523
21.5.1.4. Starting firewalld	523
21.5.1.5. Stopping firewalld	523
21.5.1.6. Verifying the permanent firewalld configuration	524
21.5.2. Viewing the current status and settings of firewalld	524
21.5.2.1. Viewing the current status of firewalld	524
21.5.2.2. Viewing allowed services using GUI	525
21.5.2.3. Viewing firewalld settings using CLI	525
21.5.3. Controlling network traffic using firewalld	526
21.5.3.1. Disabling all traffic in case of emergency using CLI	526
21.5.3.2. Controlling traffic with predefined services using CLI	527
21.5.3.3. Controlling traffic with predefined services using GUI	527
21.5.3.4. Adding new services	528
21.5.3.5. Opening ports using GUI	529
21.5.3.6. Controlling traffic with protocols using GUI	529
21.5.3.7. Opening source ports using GUI	529
21.5.4. Controlling ports using CLI	530



---

21.5.4.1. Opening a port	530
21.5.4.2. Closing a port	530
21.5.5. Working with firewalld zones	531
21.5.5.1. Listing zones	531
21.5.5.2. Modifying firewalld settings for a certain zone	532
21.5.5.3. Changing the default zone	532
21.5.5.4. Assigning a network interface to a zone	532
21.5.5.5. Assigning a zone to a connection using nmcli	533
21.5.5.6. Manually assigning a zone to a network connection in an ifcfg file	533
21.5.5.7. Creating a new zone	533
21.5.5.8. Zone configuration files	534
21.5.5.9. Using zone targets to set default behavior for incoming traffic	534
21.5.6. Using zones to manage incoming traffic depending on a source	535
21.5.6.1. Adding a source	535
21.5.6.2. Removing a source	536
21.5.6.3. Adding a source port	536
21.5.6.4. Removing a source port	536
21.5.6.5. Using zones and sources to allow a service for only a specific domain	536
21.5.7. Filtering forwarded traffic between zones	537
21.5.7.1. The relationship between policy objects and zones	538
21.5.7.2. Using priorities to sort policies	538
21.5.7.3. Using policy objects to filter traffic between locally hosted Containers and a network physically connected to the host	538
21.5.7.4. Setting the default target of policy objects	539
21.5.8. Configuring NAT using firewalld	539
21.5.8.1. NAT types	540
21.5.8.2. Configuring IP address masquerading	540
21.5.9. Using DNAT to forward HTTPS traffic to a different host	541
21.5.10. Managing ICMP requests	542
21.5.10.1. Listing and blocking ICMP requests	543
21.5.10.2. Configuring the ICMP filter using GUI	545
21.5.11. Setting and controlling IP sets using firewalld	545
21.5.11.1. Configuring IP set options using CLI	545
21.5.12. Prioritizing rich rules	547
21.5.12.1. How the priority parameter organizes rules into different chains	547
21.5.12.2. Setting the priority of a rich rule	547
21.5.13. Configuring firewall lockdown	548
21.5.13.1. Configuring lockdown using CLI	548
21.5.13.2. Configuring lockdown allowlist options using CLI	548
21.5.13.3. Configuring lockdown allowlist options using configuration files	550
21.5.14. Enabling traffic forwarding between different interfaces or sources within a firewalld zone	551
21.5.14.1. The difference between intra-zone forwarding and zones with the default target set to ACCEPT	551
21.5.14.2. Using intra-zone forwarding to forward traffic between an Ethernet and Wi-Fi network	551
21.5.15. Configuring firewalld using System Roles	552
21.5.15.1. Introduction to the firewall RHEL System Role	552
21.5.15.2. Resetting the firewalld settings using the firewall RHEL System Role	553
21.5.15.3. Forwarding incoming traffic from one local port to a different local port	554
21.5.15.4. Configuring ports using System Roles	555
21.5.15.5. Configuring a DMZ firewalld zone by using the firewalld RHEL System Role	556
21.5.16. Additional resources	557
21.6. GETTING STARTED WITH NFTABLES	558
21.6.1. Migrating from iptables to nftables	558

---

21.6.1.1. When to use firewalld, nftables, or iptables	558
21.6.1.2. Converting iptables and ip6tables rule sets to nftables	559
21.6.1.3. Converting single iptables and ip6tables rules to nftables	560
21.6.1.4. Comparison of common iptables and nftables commands	560
21.6.1.5. Additional resources	561
21.6.2. Writing and executing nftables scripts	561
21.6.2.1. Supported nftables script formats	561
21.6.2.2. Running nftables scripts	562
21.6.2.3. Using comments in nftables scripts	563
21.6.2.4. Using variables in nftables script	563
21.6.2.5. Including files in nftables scripts	564
21.6.2.6. Automatically loading nftables rules when the system boots	564
21.6.3. Creating and managing nftables tables, chains, and rules	565
21.6.3.1. Basics of nftables tables	565
21.6.3.2. Basics of nftables chains	566
Chain types	566
Chain priorities	566
Chain policies	567
21.6.3.3. Basics of nftables rules	567
21.6.3.4. Managing tables, chains, and rules using nft commands	568
21.6.4. Configuring NAT using nftables	570
21.6.4.1. NAT types	570
21.6.4.2. Configuring masquerading using nftables	571
21.6.4.3. Configuring source NAT using nftables	572
21.6.4.4. Configuring destination NAT using nftables	572
21.6.4.5. Configuring a redirect using nftables	573
21.6.5. Using sets in nftables commands	574
21.6.5.1. Using anonymous sets in nftables	574
21.6.5.2. Using named sets in nftables	574
21.6.5.3. Additional resources	576
21.6.6. Using verdict maps in nftables commands	576
21.6.6.1. Using anonymous maps in nftables	576
21.6.6.2. Using named maps in nftables	577
21.6.6.3. Additional resources	579
21.6.7. Example: Protecting a LAN and DMZ using an nftables script	579
21.6.7.1. Network conditions	579
21.6.7.2. Security requirements to the firewall script	579
21.6.7.3. Configuring logging of dropped packets to a file	580
21.6.7.4. Writing and activating the nftables script	581
21.6.8. Configuring port forwarding using nftables	584
21.6.8.1. Forwarding incoming packets to a different local port	584
21.6.8.2. Forwarding incoming packets on a specific local port to a different host	585
21.6.9. Using nftables to limit the amount of connections	585
21.6.9.1. Limiting the number of connections using nftables	585
21.6.9.2. Blocking IP addresses that attempt more than ten new incoming TCP connections within one minute	586
21.6.10. Debugging nftables rules	587
21.6.10.1. Creating a rule with a counter	587
21.6.10.2. Adding a counter to an existing rule	587
21.6.10.3. Monitoring packets that match an existing rule	588
21.6.11. Backing up and restoring the nftables rule set	589
21.6.11.1. Backing up the nftables rule set to a file	589
21.6.11.2. Restoring the nftables rule set from a file	590

21.6.12. Additional resources	590
<b>PART IV. DESIGN OF HARD DISK</b>	<b>591</b>
<b>CHAPTER 22. OVERVIEW OF AVAILABLE FILE SYSTEMS</b>	<b>592</b>
22.1. TYPES OF FILE SYSTEMS	592
22.2. LOCAL FILE SYSTEMS	593
22.3. THE XFS FILE SYSTEM	593
22.4. THE EXT4 FILE SYSTEM	594
22.5. COMPARISON OF XFS AND EXT4	595
22.6. CHOOSING A LOCAL FILE SYSTEM	596
22.7. NETWORK FILE SYSTEMS	597
22.8. SHARED STORAGE FILE SYSTEMS	597
22.9. CHOOSING BETWEEN NETWORK AND SHARED STORAGE FILE SYSTEMS	598
22.10. VOLUME-MANAGING FILE SYSTEMS	599
<b>CHAPTER 23. MOUNTING NFS SHARES</b>	<b>600</b>
23.1. INTRODUCTION TO NFS	600
23.2. SUPPORTED NFS VERSIONS	600
Default NFS version	600
Features of minor NFS versions	600
23.3. SERVICES REQUIRED BY NFS	601
The RPC services with NFSv4	602
23.4. NFS HOST NAME FORMATS	602
23.5. INSTALLING NFS	603
23.6. DISCOVERING NFS EXPORTS	603
23.7. MOUNTING AN NFS SHARE WITH MOUNT	603
23.8. COMMON NFS MOUNT OPTIONS	604
23.9. ADDITIONAL RESOURCES	606
<b>CHAPTER 24. EXPORTING NFS SHARES</b>	<b>607</b>
24.1. INTRODUCTION TO NFS	607
24.2. SUPPORTED NFS VERSIONS	607
Default NFS version	607
Features of minor NFS versions	607
24.3. THE TCP AND UDP PROTOCOLS IN NFSV3 AND NFSV4	608
24.4. SERVICES REQUIRED BY NFS	608
The RPC services with NFSv4	609
24.5. NFS HOST NAME FORMATS	609
24.6. NFS SERVER CONFIGURATION	610
24.6.1. The /etc/exports configuration file	610
Export entry	610
Default options	611
Default and overridden options	612
24.6.2. The exportfs utility	612
Common exportfs options	612
24.7. NFS AND RPCBIND	613
24.8. INSTALLING NFS	613
24.9. STARTING THE NFS SERVER	613
24.10. TROUBLESHOOTING NFS AND RPCBIND	614
24.11. CONFIGURING THE NFS SERVER TO RUN BEHIND A FIREWALL	615
24.11.1. Configuring the NFSv3-enabled server to run behind a firewall	615
24.11.2. Configuring the NFSv4-only server to run behind a firewall	616
24.11.3. Configuring an NFSv3 client to run behind a firewall	617

24.11.4. Configuring an NFSv4 client to run behind a firewall	618
24.12. EXPORTING RPC QUOTA THROUGH A FIREWALL	618
24.13. ENABLING NFS OVER RDMA (NFSORDMA)	619
24.14. ADDITIONAL RESOURCES	620
<b>CHAPTER 25. MOUNTING AN SMB SHARE ON RED HAT ENTERPRISE LINUX</b>	<b>621</b>
25.1. SUPPORTED SMB PROTOCOL VERSIONS	621
25.2. UNIX EXTENSIONS SUPPORT	622
25.3. MANUALLY MOUNTING AN SMB SHARE	622
25.4. MOUNTING AN SMB SHARE AUTOMATICALLY WHEN THE SYSTEM BOOTS	623
25.5. AUTHENTICATING TO AN SMB SHARE USING A CREDENTIALS FILE	623
25.6. FREQUENTLY USED MOUNT OPTIONS	624
<b>CHAPTER 26. OVERVIEW OF PERSISTENT NAMING ATTRIBUTES</b>	<b>626</b>
26.1. DISADVANTAGES OF NON-PERSISTENT NAMING ATTRIBUTES	626
26.2. FILE SYSTEM AND DEVICE IDENTIFIERS	626
File system identifiers	627
Device identifiers	627
Recommendations	627
26.3. DEVICE NAMES MANAGED BY THE UDEV MECHANISM IN /DEV/DISK/	627
26.3.1. File system identifiers	627
The UUID attribute in /dev/disk/by-uuid/	627
The Label attribute in /dev/disk/by-label/	628
26.3.2. Device identifiers	628
The WWID attribute in /dev/disk/by-id/	628
The Partition UUID attribute in /dev/disk/by-partuuid	629
The Path attribute in /dev/disk/by-path/	629
26.4. THE WORLD WIDE IDENTIFIER WITH DM MULTIPATH	629
26.5. LIMITATIONS OF THE UDEV DEVICE NAMING CONVENTION	630
26.6. LISTING PERSISTENT NAMING ATTRIBUTES	630
26.7. MODIFYING PERSISTENT NAMING ATTRIBUTES	632
<b>CHAPTER 27. GETTING STARTED WITH PARTITIONS</b>	<b>633</b>
27.1. CREATING A PARTITION TABLE ON A DISK WITH PARTED	633
27.2. VIEWING THE PARTITION TABLE WITH PARTED	634
27.3. CREATING A PARTITION WITH PARTED	635
27.4. SETTING A PARTITION TYPE WITH FDISK	636
27.5. RESIZING A PARTITION WITH PARTED	637
27.6. REMOVING A PARTITION WITH PARTED	639
<b>CHAPTER 28. GETTING STARTED WITH XFS</b>	<b>641</b>
28.1. THE XFS FILE SYSTEM	641
28.2. COMPARISON OF TOOLS USED WITH EXT4 AND XFS	642
<b>CHAPTER 29. MOUNTING FILE SYSTEMS</b>	<b>643</b>
29.1. THE LINUX MOUNT MECHANISM	643
29.2. LISTING CURRENTLY MOUNTED FILE SYSTEMS	643
29.3. MOUNTING A FILE SYSTEM WITH MOUNT	644
29.4. MOVING A MOUNT POINT	645
29.5. UNMOUNTING A FILE SYSTEM WITH UMount	645
29.6. COMMON MOUNT OPTIONS	646
<b>CHAPTER 30. SHARING A MOUNT ON MULTIPLE MOUNT POINTS</b>	<b>648</b>
30.1. TYPES OF SHARED MOUNTS	648
30.2. CREATING A PRIVATE MOUNT POINT DUPLICATE	648

30.3. CREATING A SHARED MOUNT POINT DUPLICATE	649
30.4. CREATING A SLAVE MOUNT POINT DUPLICATE	651
30.5. PREVENTING A MOUNT POINT FROM BEING DUPLICATED	652
<b>CHAPTER 31. PERSISTENTLY MOUNTING FILE SYSTEMS</b>	<b>653</b>
31.1. THE /ETC/FSTAB FILE	653
31.2. ADDING A FILE SYSTEM TO /ETC/FSTAB	653
<b>CHAPTER 32. PERSISTENTLY MOUNTING A FILE SYSTEM USING RHEL SYSTEM ROLES</b>	<b>655</b>
32.1. EXAMPLE ANSIBLE PLAYBOOK TO PERSISTENTLY MOUNT A FILE SYSTEM	655
<b>CHAPTER 33. MOUNTING FILE SYSTEMS ON DEMAND</b>	<b>656</b>
33.1. THE AUTOFS SERVICE	656
33.2. THE AUTOFS CONFIGURATION FILES	656
33.3. CONFIGURING AUTOFS MOUNT POINTS	658
33.4. AUTOMOUNTING NFS SERVER USER HOME DIRECTORIES WITH AUTOFS SERVICE	659
33.5. OVERRIDING OR AUGMENTING AUTOFS SITE CONFIGURATION FILES	659
33.6. USING LDAP TO STORE AUTOMOUNTER MAPS	661
33.7. USING SYSTEMD.AUTOMOUNT TO MOUNT A FILE SYSTEM ON DEMAND WITH /ETC/FSTAB	662
33.8. USING SYSTEMD.AUTOMOUNT TO MOUNT A FILE SYSTEM ON DEMAND WITH A MOUNT UNIT	663
<b>CHAPTER 34. USING SSSD COMPONENT FROM IDM TO CACHE THE AUTOFS MAPS</b>	<b>665</b>
34.1. CONFIGURING AUTOFS MANUALLY TO USE IDM SERVER AS AN LDAP SERVER	665
34.2. CONFIGURING SSSD TO CACHE AUTOFS MAPS	666
<b>CHAPTER 35. SETTING READ-ONLY PERMISSIONS FOR THE ROOT FILE SYSTEM</b>	<b>668</b>
35.1. FILES AND DIRECTORIES THAT ALWAYS RETAIN WRITE PERMISSIONS	668
35.2. CONFIGURING THE ROOT FILE SYSTEM TO MOUNT WITH READ-ONLY PERMISSIONS ON BOOT	669
<b>CHAPTER 36. MANAGING STORAGE DEVICES</b>	<b>670</b>
36.1. SETTING UP STRATIS FILE SYSTEMS	670
36.1.1. What is Stratis	670
36.1.2. Components of a Stratis volume	670
36.1.3. Block devices usable with Stratis	671
Supported devices	671
Unsupported devices	672
36.1.4. Installing Stratis	672
36.1.5. Creating an unencrypted Stratis pool	672
36.1.6. Creating an encrypted Stratis pool	673
36.1.7. Setting up a thin provisioning layer in Stratis filesystem	674
36.1.8. Binding a Stratis pool to NBDE	675
36.1.9. Binding a Stratis pool to TPM	676
36.1.10. Unlocking an encrypted Stratis pool with kernel keyring	677
36.1.11. Unlocking an encrypted Stratis pool with Clevis	677
36.1.12. Unbinding a Stratis pool from supplementary encryption	678
36.1.13. Starting and stopping Stratis pool	678
36.1.14. Creating a Stratis file system	679
36.1.15. Mounting a Stratis file system	680
36.1.16. Persistently mounting a Stratis file system	681
36.1.17. Setting up non-root Stratis filesystems in /etc/fstab using a systemd service	682
36.2. EXTENDING A STRATIS VOLUME WITH ADDITIONAL BLOCK DEVICES	682
36.2.1. Components of a Stratis volume	682
36.2.2. Adding block devices to a Stratis pool	683
36.2.3. Additional resources	684
36.3. MONITORING STRATIS FILE SYSTEMS	684

36.3.1. Stratis sizes reported by different utilities	684
36.3.2. Displaying information about Stratis volumes	684
36.3.3. Additional resources	685
36.4. USING SNAPSHOTS ON STRATIS FILE SYSTEMS	685
36.4.1. Characteristics of Stratis snapshots	685
36.4.2. Creating a Stratis snapshot	686
36.4.3. Accessing the content of a Stratis snapshot	686
36.4.4. Reverting a Stratis file system to a previous snapshot	687
36.4.5. Removing a Stratis snapshot	687
36.4.6. Additional resources	688
36.5. REMOVING STRATIS FILE SYSTEMS	688
36.5.1. Components of a Stratis volume	688
36.5.2. Removing a Stratis file system	689
36.5.3. Removing a Stratis pool	690
36.5.4. Additional resources	690
36.6. GETTING STARTED WITH SWAP	691
36.6.1. Overview of swap space	691
36.6.2. Recommended system swap space	691
36.6.3. Extending swap on an LVM2 logical volume	692
36.6.4. Creating an LVM2 logical volume for swap	693
36.6.5. Creating a swap file	694
36.6.6. Reducing swap on an LVM2 logical volume	695
36.6.7. Removing an LVM2 logical volume for swap	695
36.6.8. Removing a swap file	696
<b>CHAPTER 37. DEDUPLICATING AND COMPRESSING STORAGE</b> .....	<b>697</b>
37.1. DEPLOYING VDO	697
37.1.1. Introduction to VDO	697
37.1.2. VDO deployment scenarios	697
KVM	697
File systems	698
Placement of VDO on iSCSI	698
LVM	699
Encryption	699
37.1.3. Components of a VDO volume	700
37.1.4. The physical and logical size of a VDO volume	701
37.1.5. Slab size in VDO	702
37.1.6. VDO requirements	702
37.1.6.1. VDO memory requirements	702
37.1.6.2. VDO storage space requirements	703
37.1.6.3. Placement of VDO in the storage stack	704
37.1.6.4. Examples of VDO requirements by physical size	705
37.1.7. Installing VDO	706
37.1.8. Creating a VDO volume	707
37.1.9. Mounting a VDO volume	708
37.1.10. Enabling periodic block discard	709
37.1.11. Monitoring VDO	709
37.2. MAINTAINING VDO	710
37.2.1. Managing free space on VDO volumes	710
37.2.1.1. The physical and logical size of a VDO volume	710
37.2.1.2. Thin provisioning in VDO	711
37.2.1.3. Monitoring VDO	712
37.2.1.4. Reclaiming space for VDO on file systems	712

---

37.2.1.5. Reclaiming space for VDO without a file system	713
37.2.1.6. Reclaiming space for VDO on Fibre Channel or Ethernet network	713
37.2.2. Starting or stopping VDO volumes	713
37.2.2.1. Started and activated VDO volumes	714
37.2.2.2. Starting a VDO volume	714
37.2.2.3. Stopping a VDO volume	714
37.2.2.4. Additional resources	715
37.2.3. Automatically starting VDO volumes at system boot	715
37.2.3.1. Started and activated VDO volumes	715
37.2.3.2. Activating a VDO volume	716
37.2.3.3. Deactivating a VDO volume	716
37.2.4. Selecting a VDO write mode	716
37.2.4.1. VDO write modes	717
37.2.4.2. The internal processing of VDO write modes	717
37.2.4.3. Checking the write mode on a VDO volume	718
37.2.4.4. Checking for a volatile cache	718
37.2.4.5. Setting a VDO write mode	719
37.2.5. Recovering a VDO volume after an unclean shutdown	719
37.2.5.1. VDO write modes	720
37.2.5.2. VDO volume recovery	720
Automatic and manual recovery	721
37.2.5.3. VDO operating modes	721
37.2.5.4. Recovering a VDO volume online	722
37.2.5.5. Forcing an offline rebuild of a VDO volume metadata	722
37.2.5.6. Removing an unsuccessfully created VDO volume	723
37.2.6. Optimizing the UDS index	724
37.2.6.1. Components of a VDO volume	724
37.2.6.2. The UDS index	725
37.2.6.3. Recommended UDS index configuration	725
37.2.7. Enabling or disabling deduplication in VDO	726
37.2.7.1. Deduplication in VDO	726
37.2.7.2. Enabling deduplication on a VDO volume	726
37.2.7.3. Disabling deduplication on a VDO volume	727
37.2.8. Enabling or disabling compression in VDO	727
37.2.8.1. Compression in VDO	727
37.2.8.2. Enabling compression on a VDO volume	728
37.2.8.3. Disabling compression on a VDO volume	728
37.2.9. Increasing the size of a VDO volume	728
37.2.9.1. The physical and logical size of a VDO volume	728
37.2.9.2. Thin provisioning in VDO	729
37.2.9.3. Increasing the logical size of a VDO volume	730
37.2.9.4. Increasing the physical size of a VDO volume	731
37.2.10. Removing VDO volumes	731
37.2.10.1. Removing a working VDO volume	731
37.2.10.2. Removing an unsuccessfully created VDO volume	731
37.2.11. Additional resources	732
37.3. DISCARDING UNUSED BLOCKS	732
37.3.1. Block discard operations	732
Requirements	732
37.3.2. Types of block discard operations	732
Recommendations	733
37.3.3. Performing batch block discard	733
37.3.4. Enabling online block discard	734

---

37.3.5. Enabling periodic block discard	734
37.4. MANAGING VIRTUAL DATA OPTIMIZER VOLUMES USING THE WEB CONSOLE	734
37.4.1. VDO volumes in the web console	735
37.4.2. Creating VDO volumes in the web console	735
37.4.3. Formatting VDO volumes in the web console	736
37.4.4. Extending VDO volumes in the web console	738
<b>PART V. DESIGN OF LOG FILE</b>	<b>739</b>
<b>CHAPTER 38. AUDITING THE SYSTEM</b>	<b>740</b>
38.1. LINUX AUDIT	740
38.2. AUDIT SYSTEM ARCHITECTURE	741
38.3. CONFIGURING AUDITD FOR A SECURE ENVIRONMENT	742
38.4. STARTING AND CONTROLLING AUDITD	743
38.5. UNDERSTANDING AUDIT LOG FILES	744
38.6. USING AUDITCTL FOR DEFINING AND EXECUTING AUDIT RULES	748
38.7. DEFINING PERSISTENT AUDIT RULES	749
38.8. USING PRE-CONFIGURED RULES FILES	749
38.9. USING AUGENRULES TO DEFINE PERSISTENT RULES	750
38.10. DISABLING AUGENRULES	750
38.11. SETTING UP AUDIT TO MONITOR SOFTWARE UPDATES	751
38.12. MONITORING USER LOGIN TIMES WITH AUDIT	753
38.13. ADDITIONAL RESOURCES	754
<b>PART VI. DESIGN OF KERNEL</b>	<b>755</b>
<b>CHAPTER 39. THE LINUX KERNEL</b>	<b>756</b>
39.1. WHAT THE KERNEL IS	756
39.2. RPM PACKAGES	756
Types of RPM packages	756
39.3. THE LINUX KERNEL RPM PACKAGE OVERVIEW	757
39.4. DISPLAYING CONTENTS OF THE KERNEL PACKAGE	757
39.5. UPDATING THE KERNEL	758
39.6. INSTALLING SPECIFIC KERNEL VERSIONS	759
<b>CHAPTER 40. CONFIGURING KERNEL COMMAND-LINE PARAMETERS</b>	<b>760</b>
40.1. UNDERSTANDING KERNEL COMMAND-LINE PARAMETERS	760
40.2. WHAT GRUBBY IS	760
40.3. WHAT BOOT ENTRIES ARE	761
40.4. CHANGING KERNEL COMMAND-LINE PARAMETERS FOR ALL BOOT ENTRIES	761
40.5. CHANGING KERNEL COMMAND-LINE PARAMETERS FOR A SINGLE BOOT ENTRY	762
40.6. CHANGING KERNEL COMMAND-LINE PARAMETERS TEMPORARILY AT BOOT TIME	763
40.7. CONFIGURING GRUB SETTINGS TO ENABLE SERIAL CONSOLE CONNECTION	764
<b>CHAPTER 41. CONFIGURING KERNEL PARAMETERS AT RUNTIME</b>	<b>765</b>
41.1. WHAT ARE KERNEL PARAMETERS	765
41.2. CONFIGURING KERNEL PARAMETERS TEMPORARILY WITH SYSCTL	766
41.3. CONFIGURING KERNEL PARAMETERS PERMANENTLY WITH SYSCTL	766
41.4. USING CONFIGURATION FILES IN /ETC/SYSCTL.D/ TO ADJUST KERNEL PARAMETERS	767
41.5. CONFIGURING KERNEL PARAMETERS TEMPORARILY THROUGH /PROC/SYS/	768
<b>CHAPTER 42. INSTALLING AND CONFIGURING KDUMP</b>	<b>769</b>
42.1. INSTALLING KDUMP	769
42.1.1. What is kdump	769
42.1.2. Installing kdump using Anaconda	769



42.1.3. Installing kdump on the command line	770
42.2. CONFIGURING KDUMP ON THE COMMAND LINE	771
42.2.1. Estimating the kdump size	771
42.2.2. Configuring kdump memory usage	771
42.2.3. Configuring the kdump target	773
42.2.4. Configuring the kdump core collector	775
42.2.5. Configuring the kdump default failure responses	776
42.2.6. Testing the kdump configuration	777
42.3. ENABLING KDUMP	778
42.3.1. Enabling kdump for all installed kernels	778
42.3.2. Enabling kdump for a specific installed kernel	778
42.3.3. Disabling the kdump service	779
42.4. CONFIGURING KDUMP IN THE WEB CONSOLE	780
42.4.1. Configuring kdump memory usage and target location in web console	780
42.4.2. Additional resources	782
42.5. SUPPORTED KDUMP CONFIGURATIONS AND TARGETS	782
42.5.1. Memory requirements for kdump	782
42.5.2. Minimum threshold for automatic memory reservation	783
42.5.3. Supported kdump targets	784
42.5.4. Supported kdump filtering levels	785
42.5.5. Supported default failure responses	785
42.5.6. Using final_action parameter	786
42.6. TESTING THE KDUMP CONFIGURATION	786
42.7. USING KEXEC TO BOOT INTO A DIFFERENT KERNEL	787
42.8. PREVENTING KERNEL DRIVERS FROM LOADING FOR KDUMP	788
42.9. RUNNING KDUMP ON SYSTEMS WITH ENCRYPTED DISK	789
42.10. FIRMWARE ASSISTED DUMP MECHANISMS	790
42.10.1. Firmware assisted dump on IBM PowerPC hardware	790
42.10.2. Enabling firmware assisted dump mechanism	790
42.10.3. Firmware assisted dump mechanisms on IBM Z hardware	791
42.10.4. Using sadump on Fujitsu PRIMEQUEST systems	792
42.11. ANALYZING A CORE DUMP	793
42.11.1. Installing the crash utility	793
42.11.2. Running and exiting the crash utility	793
42.11.3. Displaying various indicators in the crash utility	795
42.11.4. Using Kernel Oops Analyzer	797
42.11.5. The Kdump Helper tool	798
42.12. USING EARLY KDUMP TO CAPTURE BOOT TIME CRASHES	798
42.12.1. What is early kdump	798
42.12.2. Enabling early kdump	799
42.13. RELATED INFORMATION	800
<b>CHAPTER 43. APPLYING PATCHES WITH KERNEL LIVE PATCHING .....</b>	<b>801</b>
43.1. LIMITATIONS OF KPATCH	801
43.2. SUPPORT FOR THIRD-PARTY LIVE PATCHING	801
43.3. ACCESS TO KERNEL LIVE PATCHES	802
43.4. COMPONENTS OF KERNEL LIVE PATCHING	802
43.5. HOW KERNEL LIVE PATCHING WORKS	802
43.6. SUBSCRIBING THE CURRENTLY INSTALLED KERNELS TO THE LIVE PATCHING STREAM	803
43.7. AUTOMATICALLY SUBSCRIBING ANY FUTURE KERNEL TO THE LIVE PATCHING STREAM	804
43.8. DISABLING AUTOMATIC SUBSCRIPTION TO THE LIVE PATCHING STREAM	806
43.9. UPDATING KERNEL PATCH MODULES	807
43.10. REMOVING THE LIVE PATCHING PACKAGE	808

43.11. UNINSTALLING THE KERNEL PATCH MODULE	809
43.12. DISABLING KPATCH.SERVICE	810
<b>CHAPTER 44. SETTING LIMITS FOR APPLICATIONS</b>	<b>812</b>
44.1. UNDERSTANDING CONTROL GROUPS	812
44.2. WHAT ARE KERNEL RESOURCE CONTROLLERS	813
44.3. WHAT ARE NAMESPACES	814
44.4. SETTING CPU LIMITS TO APPLICATIONS USING CGROUPS-V1	815
<b>CHAPTER 45. ANALYZING SYSTEM PERFORMANCE WITH BPF COMPILER COLLECTION</b>	<b>819</b>
45.1. INSTALLING THE BCC-TOOLS PACKAGE	819
45.2. USING SELECTED BCC-TOOLS FOR PERFORMANCE ANALYSES	819
Using execsnoop to examine the system processes	819
Using opensnoop to track what files a command opens	820
Using biotop to examine the I/O operations on the disk	821
Using xfs slower to expose unexpectedly slow file system operations	822
<b>PART VII. DESIGN OF HIGH AVAILABILITY SYSTEM</b>	<b>824</b>
<b>CHAPTER 46. HIGH AVAILABILITY ADD-ON OVERVIEW</b>	<b>825</b>
46.1. HIGH AVAILABILITY ADD-ON COMPONENTS	825
46.2. HIGH AVAILABILITY ADD-ON CONCEPTS	825
46.2.1. Fencing	825
46.2.2. Quorum	826
46.2.3. Cluster resources	826
46.3. PACEMAKER OVERVIEW	827
46.3.1. Pacemaker architecture components	827
46.3.2. Pacemaker configuration and management tools	828
46.3.3. The cluster and pacemaker configuration files	828
46.4. LVM LOGICAL VOLUMES IN A RED HAT HIGH AVAILABILITY CLUSTER	828
46.4.1. Choosing HA-LVM or shared volumes	828
46.4.2. Configuring LVM volumes in a cluster	829
<b>CHAPTER 47. GETTING STARTED WITH PACEMAKER</b>	<b>831</b>
47.1. LEARNING TO USE PACEMAKER	831
47.2. LEARNING TO CONFIGURE FAILOVER	835
<b>CHAPTER 48. THE PCS COMMAND LINE INTERFACE</b>	<b>840</b>
48.1. PCS HELP DISPLAY	840
48.2. VIEWING THE RAW CLUSTER CONFIGURATION	840
48.3. SAVING A CONFIGURATION CHANGE TO A WORKING FILE	840
48.4. DISPLAYING CLUSTER STATUS	841
48.5. DISPLAYING THE FULL CLUSTER CONFIGURATION	841
48.6. MODIFYING THE COROSYNC.CONF FILE WITH THE PCS COMMAND	842
48.7. DISPLAYING THE COROSYNC.CONF FILE WITH THE PCS COMMAND	842
<b>CHAPTER 49. CREATING A RED HAT HIGH-AVAILABILITY CLUSTER WITH PACEMAKER</b>	<b>845</b>
49.1. INSTALLING CLUSTER SOFTWARE	845
49.2. INSTALLING THE PCP-ZEROCONF PACKAGE (RECOMMENDED)	847
49.3. CREATING A HIGH AVAILABILITY CLUSTER	847
49.4. CREATING A HIGH AVAILABILITY CLUSTER WITH MULTIPLE LINKS	848
49.5. CONFIGURING FENCING	850
49.6. BACKING UP AND RESTORING A CLUSTER CONFIGURATION	851
49.7. ENABLING PORTS FOR THE HIGH AVAILABILITY ADD-ON	851
<b>CHAPTER 50. CONFIGURING AN ACTIVE/PASSIVE APACHE HTTP SERVER IN A RED HAT HIGH</b>	

<b>CHAPTER 50. CONFIGURING AN ACTIVE/PASSIVE CLUSTER WITH SERVERS IN A RED HAT HIGH AVAILABILITY CLUSTER</b> .....	<b>854</b>
50.1. CONFIGURING AN LVM VOLUME WITH AN XFS FILE SYSTEM IN A PACEMAKER CLUSTER	855
50.2. ENSURING A VOLUME GROUP IS NOT ACTIVATED ON MULTIPLE CLUSTER NODES (RHEL 8.4 AND EARLIER)	857
50.3. CONFIGURING AN APACHE HTTP SERVER	858
50.4. CREATING THE RESOURCES AND RESOURCE GROUPS	859
50.5. TESTING THE RESOURCE CONFIGURATION	861
<b>CHAPTER 51. CONFIGURING AN ACTIVE/PASSIVE NFS SERVER IN A RED HAT HIGH AVAILABILITY CLUSTER</b> .....	<b>863</b>
51.1. CONFIGURING AN LVM VOLUME WITH AN XFS FILE SYSTEM IN A PACEMAKER CLUSTER	863
51.2. ENSURING A VOLUME GROUP IS NOT ACTIVATED ON MULTIPLE CLUSTER NODES (RHEL 8.4 AND EARLIER)	865
51.3. CONFIGURING AN NFS SHARE	867
51.4. CONFIGURING THE RESOURCES AND RESOURCE GROUP FOR AN NFS SERVER IN A CLUSTER	868
51.5. TESTING THE NFS RESOURCE CONFIGURATION	871
51.5.1. Testing the NFS export	871
51.5.2. Testing for failover	872
<b>CHAPTER 52. GFS2 FILE SYSTEMS IN A CLUSTER</b> .....	<b>874</b>
52.1. CONFIGURING A GFS2 FILE SYSTEM IN A CLUSTER	874
52.2. CONFIGURING AN ENCRYPTED GFS2 FILE SYSTEM IN A CLUSTER	879
52.2.1. Configure a shared logical volume in a Pacemaker cluster	880
52.2.2. Encrypt the logical volume and create a crypt resource	883
52.2.3. Format the encrypted logical volume with a GFS2 file system and create a file system resource for the cluster	884
52.3. MIGRATING A GFS2 FILE SYSTEM FROM RHEL7 TO RHEL8	886
<b>CHAPTER 53. CONFIGURING FENCING IN A RED HAT HIGH AVAILABILITY CLUSTER</b> .....	<b>888</b>
53.1. DISPLAYING AVAILABLE FENCE AGENTS AND THEIR OPTIONS	888
53.2. CREATING A FENCE DEVICE	889
53.3. GENERAL PROPERTIES OF FENCING DEVICES	889
53.4. TESTING A FENCE DEVICE	897
53.5. CONFIGURING FENCING LEVELS	900
53.6. CONFIGURING FENCING FOR REDUNDANT POWER SUPPLIES	901
53.7. DISPLAYING CONFIGURED FENCE DEVICES	901
53.8. EXPORTING FENCE DEVICES AS PCS COMMANDS	902
53.9. MODIFYING AND DELETING FENCE DEVICES	902
53.10. MANUALLY FENCING A CLUSTER NODE	902
53.11. DISABLING A FENCE DEVICE	903
53.12. PREVENTING A NODE FROM USING A FENCING DEVICE	903
53.13. CONFIGURING ACPI FOR USE WITH INTEGRATED FENCE DEVICES	903
53.13.1. Disabling ACPI Soft-Off with the BIOS	904
53.13.2. Disabling ACPI Soft-Off in the logind.conf file	905
53.13.3. Disabling ACPI completely in the GRUB 2 file	906
<b>CHAPTER 54. CONFIGURING CLUSTER RESOURCES</b> .....	<b>907</b>
Resource creation examples	907
Deleting a configured resource	907
54.1. RESOURCE AGENT IDENTIFIERS	907
54.2. DISPLAYING RESOURCE-SPECIFIC PARAMETERS	908
54.3. CONFIGURING RESOURCE META OPTIONS	909
54.3.1. Changing the default value of a resource option	912
54.3.2. Changing the default value of a resource option for sets of resources	912

54.3.3. Displaying currently configured resource defaults	913
54.3.4. Setting meta options on resource creation	913
54.4. CONFIGURING RESOURCE GROUPS	914
54.4.1. Creating a resource group	914
54.4.2. Removing a resource group	915
54.4.3. Displaying resource groups	915
54.4.4. Group options	915
54.4.5. Group stickiness	915
54.5. DETERMINING RESOURCE BEHAVIOR	915
<b>CHAPTER 55. DETERMINING WHICH NODES A RESOURCE CAN RUN ON</b>	<b>916</b>
55.1. CONFIGURING LOCATION CONSTRAINTS	916
55.2. LIMITING RESOURCE DISCOVERY TO A SUBSET OF NODES	917
55.3. CONFIGURING A LOCATION CONSTRAINT STRATEGY	919
55.3.1. Configuring an "Opt-In" cluster	919
55.3.2. Configuring an "Opt-Out" cluster	920
55.4. CONFIGURING A RESOURCE TO PREFER ITS CURRENT NODE	920
<b>CHAPTER 56. DETERMINING THE ORDER IN WHICH CLUSTER RESOURCES ARE RUN</b>	<b>922</b>
56.1. CONFIGURING MANDATORY ORDERING	923
56.2. CONFIGURING ADVISORY ORDERING	923
56.3. CONFIGURING ORDERED RESOURCE SETS	923
56.4. CONFIGURING STARTUP ORDER FOR RESOURCE DEPENDENCIES NOT MANAGED BY PACEMAKER	925
<b>CHAPTER 57. COLOCATING CLUSTER RESOURCES</b>	<b>927</b>
57.1. SPECIFYING MANDATORY PLACEMENT OF RESOURCES	928
57.2. SPECIFYING ADVISORY PLACEMENT OF RESOURCES	928
57.3. COLOCATING SETS OF RESOURCES	929
<b>CHAPTER 58. DISPLAYING RESOURCE CONSTRAINTS AND RESOURCE DEPENDENCIES</b>	<b>930</b>
<b>CHAPTER 59. DETERMINING RESOURCE LOCATION WITH RULES</b>	<b>933</b>
59.1. PACEMAKER RULES	933
59.1.1. Node attribute expressions	933
59.1.2. Time/date based expressions	935
59.1.3. Date specifications	936
59.2. CONFIGURING A PACEMAKER LOCATION CONSTRAINT USING RULES	936
<b>CHAPTER 60. MANAGING CLUSTER RESOURCES</b>	<b>938</b>
60.1. DISPLAYING CONFIGURED RESOURCES	938
60.2. EXPORTING CLUSTER RESOURCES AS PCS COMMANDS	939
60.3. MODIFYING RESOURCE PARAMETERS	940
60.4. CLEARING FAILURE STATUS OF CLUSTER RESOURCES	940
60.5. MOVING RESOURCES IN A CLUSTER	941
60.5.1. Moving resources due to failure	941
60.5.2. Moving resources due to connectivity changes	942
60.6. DISABLING A MONITOR OPERATION	942
60.7. CONFIGURING AND MANAGING CLUSTER RESOURCE TAGS	943
60.7.1. Tagging cluster resources for administration by category	943
60.7.2. Deleting a tagged cluster resource	944
<b>CHAPTER 61. CREATING CLUSTER RESOURCES THAT ARE ACTIVE ON MULTIPLE NODES (CLONED RESOURCES)</b>	<b>945</b>
61.1. CREATING AND REMOVING A CLONED RESOURCE	945

61.2. CONFIGURING CLONE RESOURCE CONSTRAINTS	947
61.3. PROMOTABLE CLONE RESOURCES	948
61.3.1. Creating a promotable clone resource	948
61.3.2. Configuring promotable resource constraints	949
61.4. DEMOTING A PROMOTED RESOURCE ON FAILURE	949
<b>CHAPTER 62. MANAGING CLUSTER NODES</b>	<b>951</b>
62.1. STOPPING CLUSTER SERVICES	951
62.2. ENABLING AND DISABLING CLUSTER SERVICES	951
62.3. ADDING CLUSTER NODES	951
62.4. REMOVING CLUSTER NODES	953
62.5. ADDING A NODE TO A CLUSTER WITH MULTIPLE LINKS	953
62.6. ADDING AND MODIFYING LINKS IN AN EXISTING CLUSTER	953
62.6.1. Adding and removing links in an existing cluster	953
62.6.2. Modifying a link in a cluster with multiple links	954
62.6.3. Modifying the link addresses in a cluster with a single link	954
62.6.4. Modifying the link options for a link in a cluster with a single link	955
62.6.5. Modifying a link when adding a new link is not possible	956
62.7. CONFIGURING A NODE HEALTH STRATEGY	956
62.8. CONFIGURING A LARGE CLUSTER WITH MANY RESOURCES	957
<b>CHAPTER 63. PACEMAKER CLUSTER PROPERTIES</b>	<b>959</b>
63.1. SUMMARY OF CLUSTER PROPERTIES AND OPTIONS	959
63.2. SETTING AND REMOVING CLUSTER PROPERTIES	964
63.3. QUERYING CLUSTER PROPERTY SETTINGS	965
<b>CHAPTER 64. CONFIGURING A VIRTUAL DOMAIN AS A RESOURCE</b>	<b>966</b>
64.1. VIRTUAL DOMAIN RESOURCE OPTIONS	966
64.2. CREATING THE VIRTUAL DOMAIN RESOURCE	968
<b>CHAPTER 65. CONFIGURING CLUSTER QUORUM</b>	<b>970</b>
65.1. CONFIGURING QUORUM OPTIONS	970
65.2. MODIFYING QUORUM OPTIONS	971
65.3. DISPLAYING QUORUM CONFIGURATION AND STATUS	971
65.4. RUNNING INQUORATE CLUSTERS	972
<b>CHAPTER 66. INTEGRATING NON-COROSYNC NODES INTO A CLUSTER: THE PACEMAKER_REMOTE SERVICE</b>	<b>973</b>
66.1. HOST AND GUEST AUTHENTICATION OF PACEMAKER_REMOTE NODES	974
66.2. CONFIGURING KVM GUEST NODES	974
66.2.1. Guest node resource options	974
66.2.2. Integrating a virtual machine as a guest node	975
66.3. CONFIGURING PACEMAKER REMOTE NODES	976
66.3.1. Remote node resource options	976
66.3.2. Remote node configuration overview	976
66.4. CHANGING THE DEFAULT PORT LOCATION	977
66.5. UPGRADING SYSTEMS WITH PACEMAKER_REMOTE NODES	978
<b>CHAPTER 67. PERFORMING CLUSTER MAINTENANCE</b>	<b>979</b>
67.1. PUTTING A NODE INTO STANDBY MODE	979
67.2. MANUALLY MOVING CLUSTER RESOURCES	980
67.2.1. Moving a resource from its current node	980
67.2.2. Moving a resource to its preferred node	981
67.3. DISABLING, ENABLING, AND BANNING CLUSTER RESOURCES	981
Disabling a cluster resource	982

Enabling a cluster resource	982
Preventing a resource from running on a particular node	982
Forcing a resource to start on the current node	983
67.4. SETTING A RESOURCE TO UNMANAGED MODE	983
67.5. PUTTING A CLUSTER IN MAINTENANCE MODE	983
67.6. UPDATING A RHEL HIGH AVAILABILITY CLUSTER	984
67.7. UPGRADING REMOTE NODES AND GUEST NODES	984
67.8. MIGRATING VMS IN A RHEL CLUSTER	985
67.9. IDENTIFYING CLUSTERS BY UUID	986
<b>CHAPTER 68. CONFIGURING AND MANAGING LOGICAL VOLUMES</b> .....	<b>987</b>
68.1. OVERVIEW OF LOGICAL VOLUME MANAGEMENT	987
68.1.1. LVM architecture	987
68.1.2. Advantages of LVM	988
68.2. MANAGING LVM PHYSICAL VOLUMES	989
68.2.1. Overview of physical volumes	989
68.2.2. Multiple partitions on a disk	990
68.2.3. Creating LVM physical volume	991
68.2.4. Removing LVM physical volumes	992
68.2.5. Additional resources	993
68.3. MANAGING LVM VOLUME GROUPS	993
68.3.1. Creating LVM volume group	993
68.3.2. Combining LVM volume groups	994
68.3.3. Removing physical volumes from a volume group	995
68.3.4. Splitting a LVM volume group	996
68.3.5. Moving a volume group to another system	997
68.3.6. Removing LVM volume groups	998
68.4. MANAGING LVM LOGICAL VOLUMES	998
68.4.1. Overview of logical volumes	998
68.4.2. Using CLI commands	999
Specifying units in a command line argument	999
Specifying volume groups and logical volumes	1000
Increasing output verbosity	1000
Displaying help for LVM CLI commands	1001
68.4.3. Creating LVM logical volume	1001
68.4.4. Creating a RAID0 striped logical volume	1002
68.4.5. Renaming LVM logical volumes	1003
68.4.6. Removing a disk from a logical volume	1004
68.4.7. Removing LVM logical volumes	1005
68.4.8. Configuring persistent device numbers	1006
68.4.9. Specifying LVM extent size	1006
68.4.10. Managing LVM logical volumes using RHEL System Roles	1006
68.4.10.1. Example Ansible playbook to manage logical volumes	1006
68.4.10.2. Additional resources	1007
68.4.11. Removing LVM volume groups	1007
68.5. MODIFYING THE SIZE OF A LOGICAL VOLUME	1008
68.5.1. Growing a logical volume and file system	1008
68.5.2. Shrinking logical volumes	1010
68.5.3. Extending a striped logical volume	1011
68.6. CUSTOMIZED REPORTING FOR LVM	1013
68.6.1. Controlling the format of the LVM display	1013
68.6.2. LVM object display fields	1015
68.6.3. Sorting LVM reports	1023

68.6.4. Specifying the units for an LVM report display	1024
68.6.5. Displaying LVM command output in JSON format	1025
68.6.6. Displaying the LVM command log	1026
68.7. CONFIGURING RAID LOGICAL VOLUMES	1027
68.7.1. RAID logical volumes	1027
68.7.2. RAID levels and linear support	1028
68.7.3. LVM RAID segment types	1029
68.7.4. Creating RAID logical volumes	1031
68.7.5. Creating a RAID0 striped logical volume	1032
68.7.6. Parameters for creating a RAID0	1033
68.7.7. Soft data corruption	1033
68.7.8. Creating a RAID LV with DM integrity	1034
68.7.9. Minimum and maximum I/O rate options	1036
68.7.10. Converting a Linear device to a RAID logical volume	1037
68.7.11. Converting an LVM RAID1 logical volume to an LVM linear logical volume	1037
68.7.12. Converting a mirrored LVM device to a RAID1 logical volume	1038
68.7.13. Resizing a RAID logical volume	1039
68.7.14. Changing the number of images in an existing RAID1 device	1039
68.7.15. Splitting off a RAID image as a separate logical volume	1041
68.7.16. Splitting and Merging a RAID Image	1042
68.7.17. Setting a RAID fault policy	1043
68.7.17.1. The allocate RAID Fault Policy	1044
68.7.17.2. The warn RAID Fault Policy	1045
68.7.18. Replacing a RAID device in a logical volume	1045
68.7.18.1. Replacing a RAID device that has not failed	1045
68.7.18.2. Failed devices in LVM RAID	1048
68.7.18.3. Recovering a failed RAID device in a logical volume	1048
68.7.18.4. Replacing a failed RAID device in a logical volume	1049
68.7.19. Checking data coherency in a RAID logical volume (RAID scrubbing)	1050
68.7.20. Converting a RAID level (RAID takeover)	1052
68.7.21. Changing attributes of a RAID volume (RAID reshape)	1052
68.7.22. Controlling I/O Operations on a RAID1 logical volume	1052
68.7.23. Changing the region size on a RAID logical volume	1052
68.8. SNAPSHOT OF LOGICAL VOLUMES	1053
68.8.1. Overview of snapshot volumes	1053
68.8.2. Creating a snapshot of the original volume	1054
68.8.3. Merging snapshot to its original volume	1055
68.9. CREATING AND MANAGING THIN PROVISIONED VOLUMES (THIN VOLUMES)	1056
68.9.1. Overview of thin provisioning	1056
68.9.2. Creating thinly-provisioned logical volumes	1057
68.9.3. Overview of chunk size	1061
68.9.4. Thinly-provisioned snapshot volumes	1062
68.9.5. Creating thinly-provisioned snapshot volumes	1062
68.10. ENABLING CACHING TO IMPROVE LOGICAL VOLUME PERFORMANCE	1064
68.10.1. Caching methods in LVM	1064
68.10.2. LVM caching components	1065
68.10.3. Enabling dm-cache caching for a logical volume	1066
68.10.4. Enabling dm-cache caching with a cachepool for a logical volume	1067
68.10.5. Enabling dm-writocache caching for a logical volume	1069
68.10.6. Disabling caching for a logical volume	1071
68.11. LOGICAL VOLUME ACTIVATION	1072
68.11.1. Controlling autoactivation of logical volumes	1072
68.11.2. Controlling logical volume activation	1073

68.11.3. Activating shared logical volumes	1074
68.11.4. Activating a logical volume with missing devices	1074
68.12. LIMITING LVM DEVICE VISIBILITY AND USAGE	1074
68.12.1. The LVM device filter	1075
68.12.1.1. Additional resources	1075
68.12.1.2. Examples of LVM device filter configurations	1075
68.13. CONTROLLING LVM ALLOCATION	1076
68.13.1. LVM allocation policies	1076
68.13.2. Preventing allocation on a physical volume	1077
68.13.3. Extending a logical volume with the cling allocation policy	1077
68.13.4. Differentiating between LVM RAID objects using tags	1079
68.14. TROUBLESHOOTING LVM	1080
68.14.1. Gathering diagnostic data on LVM	1080
68.14.2. Displaying information on failed LVM devices	1081
68.14.3. Removing lost LVM physical volumes from a volume group	1082
68.14.4. Finding the metadata of a missing LVM physical volume	1083
68.14.5. Restoring metadata on an LVM physical volume	1083
68.14.6. Rounding errors in LVM output	1085
68.14.7. Preventing the rounding error when creating an LVM volume	1085
68.14.8. Troubleshooting LVM RAID	1086
68.14.8.1. Checking data coherency in a RAID logical volume (RAID scrubbing)	1086
68.14.8.2. Failed devices in LVM RAID	1088
68.14.8.3. Recovering a failed RAID device in a logical volume	1088
68.14.8.4. Replacing a failed RAID device in a logical volume	1089
68.14.9. Troubleshooting duplicate physical volume warnings for multipathed LVM devices	1090
68.14.9.1. Root cause of duplicate PV warnings	1090
68.14.9.2. Cases of duplicate PV warnings	1091
68.14.9.3. Example LVM device filters that prevent duplicate PV warnings	1092
68.14.9.4. Applying an LVM device filter configuration	1092
68.14.9.5. Additional resources	1093





## MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see [our CTO Chris Wright's message](#).

# PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your feedback on our documentation. Let us know how we can improve it.

## Submitting comments on specific passages

1. View the documentation in the **Multi-page HTML** format and ensure that you see the **Feedback** button in the upper right corner after the page fully loads.
2. Use your cursor to highlight the part of the text that you want to comment on.
3. Click the **Add Feedback** button that appears near the highlighted text.
4. Add your feedback and click **Submit**.

## Submitting feedback through Bugzilla (account required)

1. Log in to the [Bugzilla](#) website.
2. Select the correct version from the **Version** menu.
3. Enter a descriptive title in the **Summary** field.
4. Enter your suggestion for improvement in the **Description** field. Include links to the relevant parts of the documentation.
5. Click **Submit Bug**.

## PART I. DESIGN OF INSTALLATION

# CHAPTER 1. SUPPORTED RHEL ARCHITECTURES AND SYSTEM REQUIREMENTS

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. You can deploy RHEL as a guest on supported hypervisors and Cloud provider environments as well as 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, Intel, and ARM 64-bit architectures
- IBM Power Systems, Little Endian
  - IBM Power System LC servers
  - IBM Power System AC servers
  - IBM Power System L servers
- 64-bit IBM Z



### NOTE

For installation instructions on IBM Power Servers, see [IBM installation documentation](#). To ensure that your system is supported for installing RHEL, see <https://catalog.redhat.com> and <https://access.redhat.com/articles/rhel-limits>.

## 1.2. 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. See [System requirements reference](#) for more information.

If you want to use your system as a virtualization host, review the [necessary hardware requirements for virtualization](#).

### Additional resources

- [Security hardening](#)
- [Composing a customized RHEL system image](#)

## CHAPTER 2. PREPARING FOR YOUR INSTALLATION

Before you begin to install Red Hat Enterprise Linux, review the following sections to prepare your setup for the installation.

### 2.1. RECOMMENDED STEPS

Preparing for your RHEL installation consists of the following steps:

#### Steps

1. Review and determine the installation method.
2. Check [system requirements](#).
3. Review the installation boot media options.
4. Download the required installation ISO image.
5. Create a bootable installation medium.
6. 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.

### 2.2. RHEL INSTALLATION METHODS

You can install Red Hat Enterprise Linux using any of the following methods:

- GUI-based installations
- System or cloud image-based installations
- Advanced installations



#### NOTE

This document provides details about installing RHEL using the user interfaces (GUI).

#### GUI-based installations

You can choose from the following GUI-based installation methods:

- **Install RHEL using an ISO image from the Customer Portal:** Install Red Hat Enterprise Linux by downloading the **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 supports **Boot ISO** and **DVD ISO** image files; however, the **Boot ISO** image file is recommended as the installation source defaults to CDN for the Boot ISO image file. After registering the system, the installer downloads and installs packages 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 [Customizing your installation](#).

### System or cloud image-based installations

You can use system or cloud image-based installation methods only in virtual and cloud environments.

To perform a system or cloud image-based installation, use Red Hat Image Builder. Image Builder creates customized system images of Red Hat Enterprise Linux, including the system images for cloud deployment.

For more information about installing RHEL using image builder, see [Composing a customized RHEL system image](#).

### Advanced installations

You can choose from the following advanced installation methods:

- **Perform an automated RHEL installation using Kickstart:** Kickstart is an automated process that helps you install the operating system by specifying all your requirements and configurations in a file. Kickstart file contains RHEL installation options, for example, the time zone, drive partitions, or packages to be installed. Providing a prepared Kickstart file completes installation without the need for any user intervention. This is useful when deploying Red Hat Enterprise Linux on a large number of systems at once.
- **Perform a remote RHEL installation using VNC:** The RHEL installation program offers two Virtual Network Computing (VNC) installation modes: Direct and Connect. After 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 :** With a network installation using preboot execution environment (PXE), you can 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.

### Additional resources

- For more information about the advanced installation methods, see the [Performing an advanced RHEL 8 installation](#) document.

## 2.3. 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. See [System requirements reference](#) for more information.

If you want to use your system as a virtualization host, review the [necessary hardware requirements for virtualization](#).

### Additional resources

- [Security hardening](#)
- [Composing a customized RHEL system image](#)

## 2.4. INSTALLATION BOOT MEDIA OPTIONS

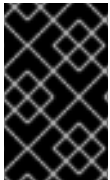
There are several options available to boot the Red Hat Enterprise Linux installation program.

### Full installation DVD or USB flash drive

Create a full installation DVD or USB flash drive using the **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.

### 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.

### Image builder

With image builder, you can create customized system and cloud images to install Red Hat Enterprise Linux in virtual and cloud environments.

### Additional resources

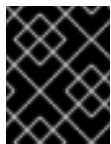
- [Performing an advanced RHEL 8 installation](#)
- [Composing a customized RHEL system image](#)

## 2.5. TYPES OF INSTALLATION ISO IMAGES

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

### DVD ISO image file

It is a full installation program that contains the BaseOS and AppStream repositories. With a DVD ISO file, you can complete the installation without access to additional repositories.



### IMPORTANT

You can use a Binary DVD for 64-bit 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:

- a. When registering and installing RHEL from the Content Delivery Network (CDN).
- b. As a minimal image that requires access to the BaseOS and AppStream repositories to



install software packages. The repositories are part of the DVD ISO image that is available for download from the [Red Hat Customer Portal](#). Download and unpack the DVD ISO image to access the repositories.

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

**Table 2.1. Boot and installation images**

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

## 2.6. DOWNLOADING A RHEL INSTALLATION ISO IMAGE

You can download Red Hat Enterprise Linux by visiting the Red Hat [customer portal](#) or you can choose to download it using the **curl** command.

### 2.6.1. Types of installation ISO images

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

#### DVD ISO image file

It is a full installation program that contains the BaseOS and AppStream repositories. With a DVD ISO file, you can complete the installation without access to additional repositories.



#### IMPORTANT

You can use a Binary DVD for 64-bit 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:

- a. When registering and installing RHEL from the Content Delivery Network (CDN).
- b. As a minimal image that requires access to the BaseOS and AppStream repositories to install software packages. The repositories are part of the DVD ISO image that is available for download from the [Red Hat Customer Portal](#). Download and unpack the DVD ISO image to access the repositories.

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

Table 2.2. Boot and installation images

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

### 2.6.2. Downloading an ISO image from the Customer Portal

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 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 [Product Downloads](#).

#### Procedure

1. Open the browser and access <https://access.redhat.com/downloads/content/rhel>. This page lists popular downloads for Red Hat Enterprise Linux.
2. Click **Download Now** beside the ISO image that you require.
3. If the desired version of RHEL is not listed, click **All Red Hat Enterprise Linux Downloads**.
  - a. From the **Product Variant** drop-down menu, select the variant and architecture that you require.
    - Optional: Select the **Packages** tab to view the packages contained in the selected variant. For information about the packages available in Red Hat Enterprise Linux 8, see the [Package Manifest](#) document.
  - b. From the **Version** drop-down menu, select the RHEL version you want to download. By default, the latest version for the selected variant and architecture is selected. The **Product Software** tab displays the image files, which include:
    - **Red Hat Enterprise Linux Binary DVD** image.
    - **Red Hat Enterprise Linux Boot ISO** image.

Additional images may be available, for example, preconfigured virtual machine images.
  - c. Click **Download Now** beside the ISO image that you require.

### 2.6.3. Downloading an ISO image using curl

With the **curl** tool, you can fetch the required file from the web using the command line to save locally or pipe it into another program as required. This section explains how to download installation images using the **curl** command.

#### Prerequisites

- The **curl** and **jq** packages are installed.  
If your Linux distribution does not use **yum** or **apt**, or if you do not use Linux, download the most appropriate software package from the [curl website](#).
- You have an offline token generated from [Red Hat API Tokens](#).
- You have a checksum of the file you want to download from [Product Downloads](#).

#### Procedure

1. Create a bash file with the following content:

```
#!/bin/bash
# set the offline token and checksum parameters
offline_token="<offline_token>"
checksum=<checksum>

# get an access token
access_token=$(curl https://sso.redhat.com/auth/realms/redhat-external/protocol/openid-connect/token -d grant_type=refresh_token -d client_id=rhsm-api -d refresh_token=$offline_token | jq -r '.access_token')

# get the filename and download url
image=$(curl -H "Authorization: Bearer $access_token" "https://api.access.redhat.com/management/v1/images/$checksum/download")
filename=$(echo $image | jq -r .body.filename)
url=$(echo $image | jq -r .body.href)

# download the file
curl $url -o $filename
```

In the text above, replace *<offline\_token>* with the token collected from the Red Hat API portal and *<checksum>* with the checksum value taken from the *Product Downloads* page.

2. Make this file executable.

```
$ chmod u+x FILEPATH/FILENAME.sh
```

3. Open a terminal window and execute the bash file.

```
$ ./FILEPATH/FILENAME.sh
```

**WARNING**

Use password management that is consistent with networking best practices.

- Do not store passwords or credentials in a plain text.
- Keep the token safe against unauthorized use.

**Additional resources**

- [Getting started with Red Hat APIs](#)

## 2.7. CREATING A BOOTABLE INSTALLATION MEDIUM FOR RHEL

This section contains information about using the ISO image file that you have downloaded to create a bootable physical installation medium, such as a USB, DVD, or CD. For more information about downloading the ISO images, see [Downloading the installation ISO image](#)

**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=RHEL8x86\_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, verify that the label is set to the correct value.

### 2.7.1. Installation boot media options

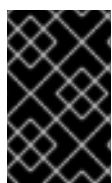
There are several options available to boot the Red Hat Enterprise Linux installation program.

**Full installation DVD or USB flash drive**

Create a full installation DVD or USB flash drive using the **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.

**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.

**Image builder**

With image builder, you can create customized system and cloud images to install Red Hat Enterprise Linux in virtual and cloud environments.

### Additional resources

- [Performing an advanced RHEL 8 installation](#)
- [Composing a customized RHEL system image](#)

## 2.7.2. 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 DVD ISO image (full install) or the Boot ISO image (minimal install). However, the 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 DVD ISO image file before you proceed. A USB flash drive is recommended when using the DVD ISO image to create bootable installation media.

## 2.7.3. Creating a bootable USB device on Linux

You can create a bootable USB device which you can then use to install Red Hat Enterprise Linux on other machines.



### IMPORTANT

Following this procedure overwrites any data previously stored on the USB drive without any warning. Back up any data or use an empty flash drive. A bootable USB drive cannot be used for storing data.

### Prerequisites

- You have downloaded an installation ISO image as described in [Downloading the installation ISO image](#).
- You have a USB flash drive with enough capacity for the ISO image. The required size varies, but the recommended USB size is 8 GB.

### Procedure

1. Connect the USB flash drive to the system.
2. Open a terminal window and display a log of recent events.

```
$ dmesg|tail
```

-

Messages resulting from the attached USB flash drive are displayed at the bottom of the log. Record the name of the connected device.

3. Log in as a root user:

```
$ su -
```

Enter your root password when prompted.

4. 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
```

5. 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,
- 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**.

6. 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.

### 2.7.4. Creating a bootable USB device on Windows

You can create a bootable USB device on a Windows system with various tools. Red Hat recommends using Fedora Media Writer, available for download at <https://github.com/FedoraQt/MediaWriter/releases>. Note that 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>.



### IMPORTANT

Following this procedure overwrites any data previously stored on the USB drive without any warning. Back up any data or use an empty flash drive. A bootable USB drive cannot be used for storing data.

#### Prerequisites

- You have downloaded an installation ISO image as described in [Downloading the installation ISO image](#).
- You have a USB flash drive with enough capacity for the ISO image. The required size varies, but the recommended USB size is 8 GB.

#### Procedure

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

### 2.7.5. Creating a bootable USB device on Mac OS X

You can create a bootable USB device which you can then use to install Red Hat Enterprise Linux on other machines.



### IMPORTANT

Following this procedure overwrites any data previously stored on the USB drive without any warning. Back up any data or use an empty flash drive. A bootable USB drive cannot be used for storing data.

#### Prerequisites

- You have downloaded an installation ISO image as described in [Downloading the installation ISO image](#).

- You have a USB flash drive with enough capacity for the ISO image. The required size varies, but the recommended USB size is 8 GB.

## 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, **disk0** is the OS X recovery disk, and **disk1** 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
```

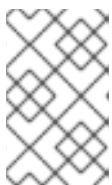
3. Identify your USB flash drive by comparing the NAME, TYPE and SIZE columns to your flash drive. For example, the NAME should be the title of the flash drive icon in the **Finder** tool. You can also compare these values to those in the information panel of the flash drive.
4. Unmount the flash drive's 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. Write the ISO image to the flash drive:

```
# sudo dd if=/path/to/image.iso of=/dev/rdisknumber
```



### 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.



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

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

6. 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.

## 2.8. 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 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).

### 2.8.1. Types of installation source

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

- **DVD:** Burn the DVD ISO image to a DVD. The DVD will be automatically used as the installation source (software package source).
- **Hard drive or USB drive:** Copy the 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 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 DVD ISO image or the installation tree (extracted contents of the DVD ISO image) to a network location and perform the installation over the network using the following protocols:
  - **NFS:** The 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.

## 2.8.2. Specify the installation source

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

- **User interface:** Select the installation source in the **Installation Source** window of the graphical install. For more information, see [Configuring installation source](#)
- **Boot option:** Configure a custom boot option to specify the installation source. For more information, see [Boot options preference](#)
- **Kickstart file:** Use the install command in a Kickstart file to specify the installation source. See the [Performing an advanced RHEL 8 installation](#) document for more information.

## 2.8.3. Ports for network-based installation

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

**Table 2.3. 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

- [Securing networks](#)

## 2.8.4. Creating an 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 an 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. For more information, see [Downloading the installation ISO image](#).
- You have created a bootable CD, DVD, or USB device from the image file. For more information, see [Creating installation media](#).
- You have verified that your firewall allows the system you are installing to access the remote installation source. For more information, see [Ports for network-based installation](#).

### Procedure

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

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

2. Copy the 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
```

- Replace */exported\_directory/* with the full path to the directory with the ISO image.
- Replace *clients* with one of the following:
  - The host name or IP address of the target system
  - The subnetwork that all target systems can use to access the ISO image
  - To allow any system with network access to the NFS server to use the ISO image, the asterisk sign (\*)

See the **exports(5)** man page for detailed information about the format of this field.

For example, a basic configuration that makes the **/rhel8-install/** directory available as read-only to all clients is:

```
/rhel8-install *
```

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

5. Start the nfs service:

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

If the service was running before you changed the **/etc/exports** file, reload the NFS server 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.

## 2.8.5. Creating an installation source using HTTP or HTTPS

You can create an installation source for a network-based installation using an installation tree, which is a directory containing extracted contents of the DVD ISO image and a valid **.treeinfo** file. The installation source is accessed over HTTP or HTTPS.

### Prerequisites

- You have an 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. For more information, see [Downloading the installation ISO image](#).
- You have created a bootable CD, DVD, or USB device from the image file. For more information, see [Creating installation media](#).
- You have verified that your firewall allows the system you are installing to access the remote installation source. For more information, see [Ports for network-based installation](#).
- The **httpd** package is installed.
- The **mod\_ssl** package is installed, if you use the **https** installation source.

**WARNING**

If your Apache web server configuration enables SSL security, prefer to enable the TLSv1.3 protocol. By default, TLSv1.2 is enabled and you may use the TLSv1 (LEGACY) protocol.

**IMPORTANT**

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

**Procedure**

1. Copy the DVD ISO image to the HTTP(S) server.
2. Create a suitable directory for mounting the DVD ISO image, for example:

```
# mkdir /mnt/rhel8-install/
```

3. Mount the DVD ISO image to the directory:

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

Replace */image\_directory/image.iso* with the path to the DVD ISO image.

4. Copy the files from the mounted image to the HTTP(S) server root.

```
# 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 other copying methods might skip the **.treeinfo** file which is required for a valid installation source. Entering the **cp** command for entire directories as shown in this procedure copies **.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 use 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

- [Deploying different types of servers](#)

## 2.8.6. Creating an installation source using FTP

You can create an installation source for a network-based installation using an installation tree, which is a directory containing extracted contents of the DVD ISO image and a valid **.treeinfo** file. The installation source is accessed over FTP.

### Prerequisites

- You have an 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. For more information, see [Downloading the installation ISO image](#).
- You have created a bootable CD, DVD, or USB device from the image file. For more information, see [Creating installation media](#).
- You have verified that your firewall allows the system you are installing to access the remote installation source. For more information, see [Ports for network-based installation](#).
- The **vsftpd** package is installed.

### Procedure

1. 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, for example, **10021** and **10031**.  
This step might be necessary in network environments featuring various firewall/NAT setups.
  - d. Optional: 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.

2. Configure the server firewall.

- a. Enable the firewall:

```
# systemctl enable firewalld
```

- b. Start the firewall:

```
# systemctl start firewalld
```

- c. Configure the firewall to allow the FTP port and port range from the previous step:

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

Replace *<min\_port>* and *<max\_port>* with the port numbers you entered into the **/etc/vsftpd/vsftpd.conf** configuration file.

- d. Reload the firewall to apply the new rules:

```
# firewall-cmd --reload
```

3. Copy the DVD ISO image to the FTP server.
4. Create a suitable directory for mounting the DVD ISO image, for example:

```
# mkdir /mnt/rhel8-install
```

5. Mount the DVD ISO image to the directory:

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

Replace ***/image-directory/image.iso*** with the path to the 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. Entering 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.

### 2.8.7. Preparing a hard drive as an installation source

This module describes how to install RHEL using a hard drive as an installation source with **ext2**, **ext3**, **ext4**, or **XFS** file systems. You can use this method for the systems without network access and the optical drive. Hard drive installations use an ISO image of the installation DVD. An ISO image is a file that contains an exact copy of the content of a DVD. With this file present on a hard drive, you can choose Hard drive as the installation source when you boot the installation program.

- To check the file system of a hard drive partition on a Windows operating system, use the **Disk Management** tool.
- To check the file system of a hard drive partition on a Linux operating system, use the **parted** tool.



#### NOTE

You cannot use ISO files on LVM (Logical Volume Management) partitions.

#### Procedure

1. Download an ISO image of the Red Hat Enterprise Linux installation DVD. Alternatively, if you have the DVD on physical media, you can create an image of an ISO with the following command on a Linux system:

```
dd if=/dev/dvd of=/path_to_image/name_of_image.iso
```

where `dvd` is your DVD drive device name, `name_of_image` is the name you give to the resulting ISO image file, and `path_to_image` is the path to the location on your system where you want to store the image.

2. Copy and paste the ISO image onto the system hard drive or a USB drive.



3. Use a **SHA256** checksum program to verify that the ISO image that you copied is intact. Many SHA256 checksum programs are available for various operating systems. On a Linux system, run:

```
$ sha256sum /path_to_image/name_of_image.iso
```

where *name\_of\_image* is the name of the ISO image file. The **SHA256** checksum program displays a string of 64 characters called a *hash*. Compare this hash to the hash displayed for this particular image on the **Downloads** page in the Red Hat Customer Portal. The two hashes should be identical.

4. Specify the HDD installation source on the kernel command line before starting the installation:

```
inst.repo=hd:<device>:/path_to_image/name_of_image.iso
```

#### Additional resources

- [Specify the installation source](#)
- [Installation source boot options](#)

## CHAPTER 3. GETTING STARTED

To get started with the installation, first review the boot menu and the available boot options. Then, depending on the choice you make, proceed to boot the installation.

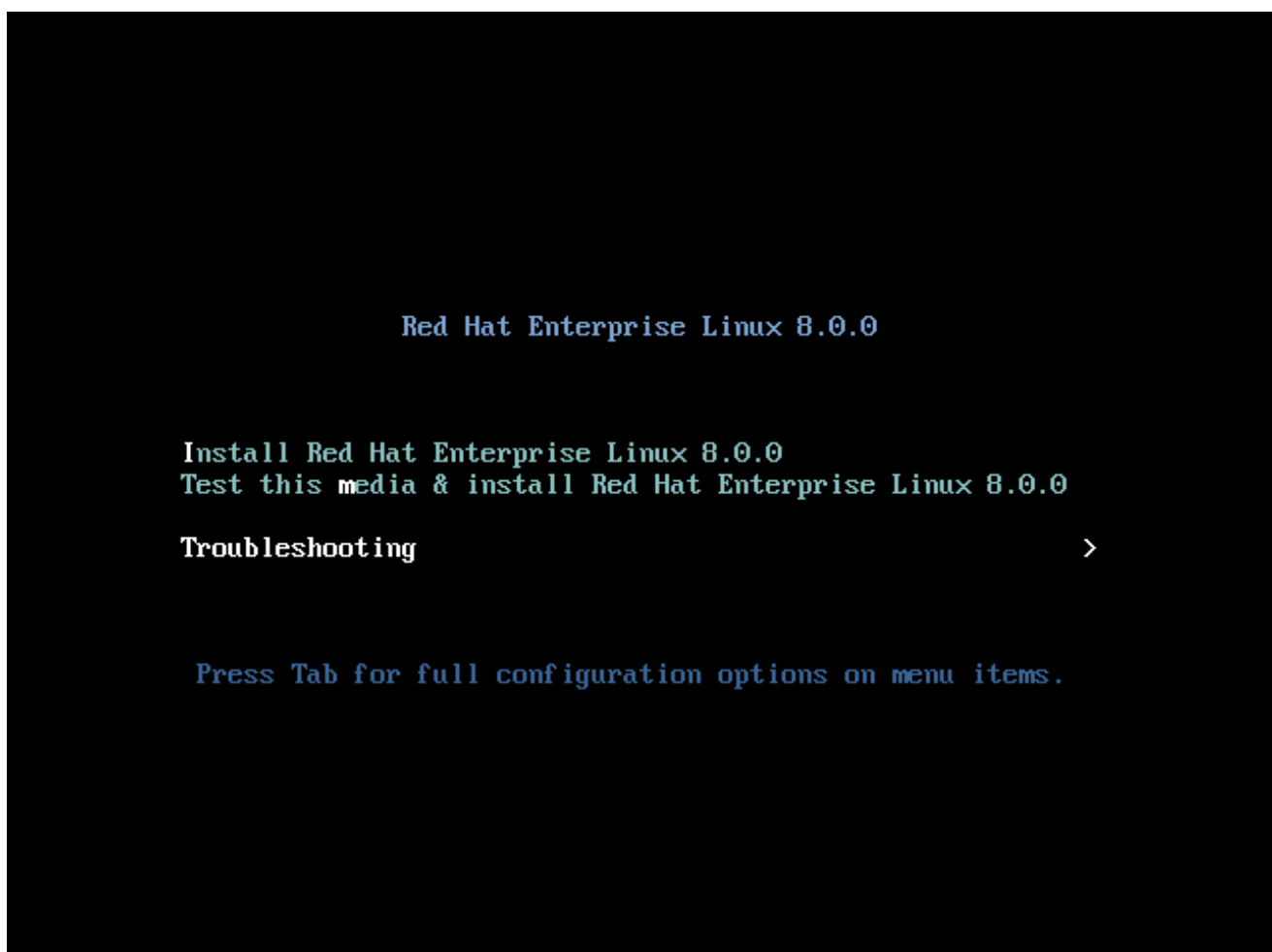
### 3.1. BOOTING THE INSTALLATION

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

#### 3.1.1. 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 3.1. 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 the **Enter** key.

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 3.1. Boot menu options

Boot menu option	Description
Install Red Hat Enterprise Linux 8	Use this option to install Red Hat Enterprise Linux using the graphical installation program. For more information, <a href="#">Installing RHEL using an ISO image from the Customer Portal</a>
Test this media & install Red Hat Enterprise Linux 8	Use this option to check the integrity of the installation media. For more information, see <a href="#">Verifying a boot media</a>
Troubleshooting >	Use this option to resolve various installation issues. Press <b>Enter</b> to display its contents.

Table 3.2. Troubleshooting options

Troubleshooting option	Description
Troubleshooting > Install Red Hat Enterprise Linux 8 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 option, restart your system and use this option. For more information, see <a href="#">Cannot boot into graphical installation</a>
Troubleshooting > Rescue a Red Hat Enterprise Linux system	Use this option to repair any issues that prevent you from booting. For more information, see <a href="#">Using a rescue mode</a>
Troubleshooting > Run a memory test	Use this option to run a memory test on your system. Press <b>Enter</b> to display its contents. For more information, see <a href="#">memtest86</a>
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.

### 3.1.2. Types of boot options

The two types of boot options are those with an equals "=" sign, and those without an equals "=" sign. Boot options are appended to the boot command line and you can append multiple options separated by 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 example, 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 installation program performs the verification and if the boot option is not present, the verification is skipped.

### 3.1.3. 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.

### 3.1.4. Editing predefined boot options using the > prompt

In BIOS-based AMD64 and Intel 64 systems, you can use the **>** prompt to edit predefined boot options. To display a full set of options, select **Test this media and install RHEL 8** from the boot menu.

#### 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 **Enter** to start the installation.
4. Press **Esc** to cancel editing and return to the boot menu.

### 3.1.5. Editing the GRUB2 menu for the UEFI-based systems

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 **e**.
2. On UEFI systems, the kernel command line starts with **linuxefi**. Move the cursor to the end of the **linuxefi** kernel command line.
3. 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.
4. When you finish editing, press **Ctrl+X** to start the installation using the specified options.

### 3.1.6. 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 [Creating a bootable DVD or CD](#) for more information.

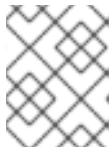
#### Procedure

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. Optionally, edit the available boot options:
  - a. **UEFI-based systems:** Press **E** to enter edit mode. Change the predefined command line to add or remove boot options. Press **Enter** to confirm your choice.
  - b. **BIOS-based systems:** Press the **Tab** key on your keyboard to enter edit mode. Change the predefined command line to add or remove boot options. Press **Enter** to confirm your choice.

**Additional Resources**

- [Graphical installation](#)
- [Boot options reference](#)

**3.1.7. 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 and supports the PXE standard. For more information, see your hardware’s documentation.

### 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.

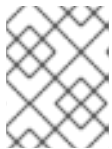


#### NOTE

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. Optionally, edit the available boot options:
  - a. **UEFI-based systems:** Press **E** to enter edit mode. Change the predefined command line to add or remove boot options. Press **Enter** to confirm your choice.
  - b. **BIOS-based systems:** Press the **Tab** key on your keyboard to enter edit mode. Change the predefined command line to add or remove boot options. Press **Enter** to confirm your choice.

### Additional Resources

- [Performing an advanced RHEL 8 installation](#)

## 3.2. INSTALLING RHEL USING AN ISO IMAGE FROM THE CUSTOMER PORTAL

Use this procedure to install RHEL using a DVD ISO image that you downloaded from the Customer Portal. The steps provide instructions to follow the RHEL Installation Program.

**WARNING**

When performing a GUI installation using the DVD ISO image file, a race condition in the installer can sometimes prevent the installation from proceeding until you register the system using the Connect to Red Hat feature. For more information, see BZ#1823578 in the Known Issues section of the [RHEL Release Notes](#) document.

**Prerequisites**

- You have downloaded the DVD ISO image file from the Customer Portal. For more information, see [Downloading beta installation images](#).
- You have created bootable installation media. For more information, see [Creating a bootable DVD or CD](#).
- You have booted the installation program and the boot menu is displayed. For more information, see [Booting the installer](#).

**Procedure**

1. From the boot menu, select **Install Red Hat Enterprise Linux 8** and press **Enter** on your keyboard.
2. In the **Welcome to Red Hat Enterprise Linux 8** window, select your language and location, and click **Continue**. The **Installation Summary** window opens and displays the default values for each setting.
3. Select **System > Installation Destination**, and in the **Local Standard Disks** pane, select the target disk and then click **Done**. The default settings are selected for the storage configuration.
4. Select **System > Network & Host Name**. The **Network and Hostname** window opens.
5. In the **Network and Hostname** window, toggle the **Ethernet** switch to **ON**, and then click **Done**. The installer connects to an available network and configures the devices available on the network. If required, from the list of networks available, you can choose a desired network and configure the devices that are available on that network.
6. Select **User Settings > Root Password**. The **Root Password** window opens.
7. In the **Root Password** window, type the password that you want to set for the root account, and then click **Done**. A root password is required to finish the installation process and to log in to the system administrator user account.
8. Optional: Select **User Settings > User Creation** to create a user account for the installation process to complete. In place of the root account, you can use this user account to perform any system administrative tasks.
9. In the **Create User** window, perform the following, and then click **Done**.
  - a. Type a name and user name for the account that you want to create.
  - b. Select the **Make this user administrator** and the **Require a password to use this account**



check boxes. The installation program adds the user to the wheel group, and creates a password protected user account with default settings. It is recommended to create a password protected administrative user account.

10. Click **Begin Installation** to start the installation, and wait for the installation to complete. It might take a few minutes.
11. When the installation process is complete, click **Reboot** to restart the system.
12. 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.



#### NOTE

If you have installed a Red Hat Enterprise Linux Beta release, on systems having UEFI Secure Boot enabled, then add the Beta public key to the system's Machine Owner Key (MOK) list.

13. From the **Initial Setup** window, accept the licensing agreement and register your system.

#### Additional resources

- [Performing a standard RHEL 8 installation](#)
- [Installation boot media options](#)

## 3.3. 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.

### 3.3.1. What is the Content Delivery Network

The Red Hat Content Delivery Network (CDN), available from [cdn.redhat.com](http://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 DVD ISO image files. However, the use of the smaller Boot ISO image file is recommended as it consumes less space than the larger 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 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 8 installation](#) document. For information about how to register and install RHEL using Kickstart, see the [Performing an advanced RHEL 8 installation](#) document.

### 3.3.2. 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 **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

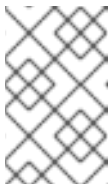
- Your system is connected to a network that can access the CDN.
- You have downloaded the **Boot ISO** image file from the Customer Portal.
- You have created bootable installation media.
- You have booted the installation program and the boot menu is displayed. Note that the installation repository 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** and press **Enter** on your keyboard.
2. In the **Welcome to Red Hat Enterprise Linux 8** window, select your language and location, and click **Continue**. The **Installation Summary** window opens and displays the default values for each setting.
3. Select **System > Installation Destination**, and in the **Local Standard Disks** pane, select the target disk and then click **Done**. The default settings are selected for the storage configuration. For more information about customizing the storage settings, see [Configuring software settings, Storage devices, Manual partitioning](#).
4. Select **System > Network & Host Name** The **Network and Hostname** window opens.
5. In the **Network and Hostname** window, toggle the **Ethernet** switch to **ON**, and then click **Done**. The installer connects to an available network and configures the devices available on the network. If required, from the list of networks available, you can choose a desired network and configure the devices that are available on that network. For more information about configuring a network or network devices, see [Network hostname](#).
6. Select **Software > Connect to Red Hat** The **Connect to Red Hat** window opens.
7. In the **Connect to Red Hat** window, perform the following steps:

- a. Select the **Authentication** method, and provide the details based on the method you select.  
For **Account** authentication method: Enter your Red Hat Customer Portal username and password details.  
  
For **Activation Key** authentication method: 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.
- b. Select the **Set System Purpose** check box, and then select the required **Role**, **SLA**, and **Usage** from the corresponding drop-down lists.  
With System Purpose you can 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.
- c. 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.  
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.
- d. Optionally, expand **Options**, and select the network communication type.
  - Select the **Use HTTP proxy** check box if your network environment allows external Internet access only or accesses the content servers through an HTTP proxy.
- a. Click **Register**. When the system is successfully registered and subscriptions are attached, the **Connect to Red Hat** window displays the attached subscription details.  
Depending on the amount of subscriptions, the registration and attachment process might take up to a minute to complete.
- b. Click **Done**.  
A *Registered* message is displayed under **Connect to Red Hat**
  1. Select **User Settings > Root Password**. The **Root Password** window opens.
  2. In the **Root Password** window, type the password that you want to set for the root account, and then click **Done**. A root password is required to finish the installation process and to log in to the system administrator user account.  
For more details about the requirements and recommendations for creating a password, see [Configuring a root password](#).
  3. Optional: Select **User Settings > User Creation** to create a user account for the installation process to complete. In place of the root account, you can use this user account to perform any system administrative tasks.
  4. In the **Create User** window, perform the following, and then click **Done**.
- c. Type a name and user name for the account that you want to create.
- d. Select the **Make this user administrator** and the **Require a password to use this account** check boxes. The installation program adds the user to the wheel group, and creates a password protected user account with default settings. It is recommended to create a password protected administrative user account.  
For more information about editing the default settings for a user account, see [Creating a user account](#).

1. Click **Begin Installation** to start the installation, and wait for the installation to complete. It might take a few minutes.
2. When the installation process is complete, click **Reboot** to restart the system.
3. 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.



#### NOTE

If you have installed a Red Hat Enterprise Linux Beta release, on systems having UEFI Secure Boot enabled, then add the Beta public key to the system's Machine Owner Key (MOK) list.

4. From the **Initial Setup** window, accept the licensing agreement and register your system.

#### Additional resources

- [How to customize your network, connect to Red Hat, system purpose, installation destination, KDUMP, and security policy](#)
- [Red Hat Insights product documentation](#)
- [Understanding Activation Keys](#)
- For information about setting up an HTTP proxy for Subscription Manager, see the **PROXY CONFIGURATION** section in the **subscription-manager** man page.

#### 3.3.2.1. 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 DVD ISO image file

If you booted the RHEL installation using either the **Boot ISO** or the **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.

### 3.3.3. 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 [Register and install from CDN using GUI](#) 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. Note: In the simple content access mode, no subscription being listed is a valid behavior.

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

## Additional resources

- [Simple Content Access](#)

### 3.3.4. 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 [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. Note: In the simple content access mode, no subscription being listed is a valid behavior.

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.

### 3.4. COMPLETING THE INSTALLATION

Wait for the installation to complete. It might take a few minutes.

After the installation is 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 8 installation* document.

## CHAPTER 4. CUSTOMIZING YOUR INSTALLATION

When installing Red Hat Enterprise Linux, you can customize location, software, and system settings and parameters, using the **Installation Summary** window.

The **Installation Summary** window contains the following categories:

### LOCALIZATION

You can configure Keyboard, Language Support, and Time and Date.

### SOFTWARE

You can configure Connect to Red Hat, Installation Source, and Software Selection.

### SYSTEM

You can configure Installation Destination, KDUMP, Network and Host Name, and Security Policy.

### USER SETTINGS

You can configure a root password to log in to the administrator account that is used for system administration tasks, and create a user account to login to the system.

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

**Table 4.1. Category status**

Status	Description
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).
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.

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.

## 4.1. CONFIGURING LANGUAGE AND LOCATION SETTINGS



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

### Prerequisites

1. You have created installation media. For more information, see [Creating a bootable DVD or CD](#).
2. You have specified an installation source if you are using the Boot ISO image file. For more information, see [Preparing an installation source](#).
3. You have booted the installation. For more information, see [Booting the installer](#).

### 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 **GeoIP** 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 [Graphical installations](#) window.
4. 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.
  - a. To continue with the installation, click **I want to proceed**, or
  - b. To quit the installation and reboot the system, click **I want to exit**.

### Additional resources

- [Configuring localization settings](#)

## 4.2. 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.



## CONFIGURING KEYBOARD, LANGUAGE, AND TIME AND DATE SETTINGS

Keyboard, Language, and Time and Date Settings are configured by default as part of [Installing RHEL using Anaconda](#). To change any of the settings, complete the following steps, otherwise proceed to [Configuring software settings](#).

### Procedure

1. Configure keyboard settings:
  - a. From the **Installation Summary** window, click **Keyboard**. The default layout depends on the option selected in [Installing RHEL using Anaconda](#).
  - b. Click **+** to open the **Add a Keyboard Layout** window and change to a different layout.
  - c. Select a layout by browsing the list or use the **Search** field.
  - d. Select the required layout and click **Add**. The new layout appears under the default layout.
  - e. Click **Options** to optionally configure a keyboard switch that you can use to cycle between available layouts. The **Layout Switching Options** window opens.
  - f. 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.

- g. Click **Done** to apply the settings and return to [Graphical installations](#).
2. Configure language settings:
  - a. 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.
  - b. 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.
  - c. Click **Done** to apply the changes and return to [Graphical installations](#).
3. Configure time and date settings:
  - a. 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 [Installing RHEL using Anaconda](#).

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](#).

- b. 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.

- c. From the **City** drop-down menu, select the city, or the city closest to your location in the same time zone.
- d. 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.

- e. Click **Done** to apply the changes and return to [Graphical installations](#).

**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.

## 4.3. CONFIGURING SYSTEM OPTIONS

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

### 4.3.1. 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.

- b. 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.
  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 [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. Click **Done**. The **Disk Encryption Password** dialog box opens.
    - i. Type your passphrase in the **Passphrase** and **Confirm** fields.
    - ii. Click **Save Password** 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 8 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 [Boot loader installation](#).

**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**.
  - a. 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.

- b. Review the displayed list of available storage devices. The **Reclaimable Space** column shows how much space can be reclaimed from each entry.
- c. 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.

- d. Click **Reclaim space** to apply the changes and return to [Graphical installations](#).



### 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.

#### Additional resources

- [How to use dm-crypt on IBM Z, LinuxONE and with the PAES cipher](#)

### 4.3.2. 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 64-bit 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.

**4.3.3. 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 [Graphical installations](#).

**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.

**4.3.4. 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

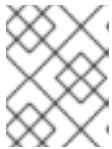
1. From the **Installation Summary** window, click **Network and Host Name**.
2. From the list in the left-hand pane, select an interface. The details are displayed in the right-hand pane.



#### 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.

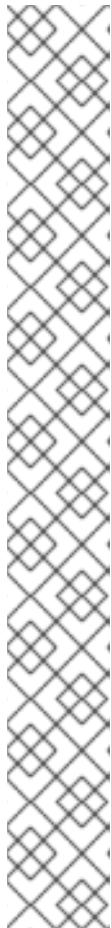
3. Toggle the **ON/OFF** switch to enable or disable the selected interface.



#### 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

- The host name can either be a fully qualified domain name (FQDN) in the format **hostname.domainname**, or a short host name without the domain. 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 system, specify only the short host name.
- When using static IP and host name configuration, it depends on the planned system use case whether to use a short name or FQDN. Red Hat Identity Management configures FQDN during provisioning but some 3rd party software products may require short name. In either case, to ensure availability of both forms in all situations, add an entry for the host in **/etc/hosts** in the format **IP FQDN short-alias**.
- The value **localhost** 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.
- Host names can only contain alphanumeric characters and - or .. Host name should be equal to or less than 64 characters. Host names cannot start or end with - and .. To be compliant with DNS, each part of a FQDN should be equal to or less than 63 characters and the FQDN total length, including dots, should not exceed 255 characters.

8. Click **Apply** to apply the host name to the installer environment.
9. Alternatively, in the **Network and Hostname** window, you can choose the Wireless option. Click **Select network** in the right-hand pane to select your wifi connection, enter the password if required, and click **Done**.

### 4.3.4.1. Adding a virtual network interface

This procedure describes how 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.

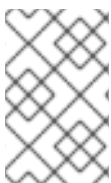
3. Select the interface type and click **Add**. An editing interface dialog box opens, allowing you to edit any available settings for your chosen interface type.  
For more information see [Editing network interface](#).
4. Click **Save** to confirm the virtual interface settings and return to the **Network & Host name** window.

**NOTE**

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

#### 4.3.4.2. 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.

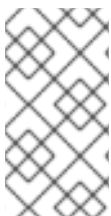
**NOTE**

On 64-bit 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 [Adding a virtual interface](#).

#### 4.3.4.3. 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.

### 4.3.4.4. 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.

#### 4.3.4.5. 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.

#### 4.3.4.6. Additional resources

- [Configuring and managing networking](#)

#### 4.3.5. Configuring Connect to Red Hat

The Red Hat Content Delivery Network (CDN), available from [cdn.redhat.com](http://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 DVD ISO image files. However, the use of the smaller Boot ISO image file is recommended as it consumes less space than the larger 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 DVD ISO image file.
- Integrated support for connecting to Red Hat Insights and enabling System Purpose.

### 4.3.5.1. 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 the **Connect to Red Hat** screen to register your system and attach your Red Hat subscription
- During a Kickstart installation when using Kickstart automation scripts
- After installation using the **subscription-manager 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

- [Composing a customized RHEL system image](#)

- [Performing an advanced RHEL 8 installation](#)
- [Using and Configuring Red Hat Subscription Manager](#)

### 4.3.5.2. 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 Red Hat Enterprise Linux 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**. However, the **Custom base URL** field requires the HTTP protocol.
- 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. A *Registered* message is displayed under **Connect to Red Hat**

#### 4.3.5.3. Installation source repository after system registration

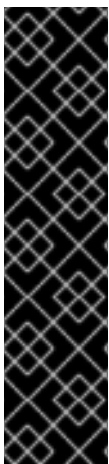
The installation source repository used after system registration is dependent on how the system was booted.

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

If you booted the RHEL installation using either the **Boot ISO** or the **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.



#### 4.3.5.4. 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 [Register and install from CDN using GUI](#) 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. Note: In the simple content access mode, no subscription being listed is a valid behavior.

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

##### Additional resources

- [Simple Content Access](#)

#### 4.3.5.5. 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 [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. Note: In the simple content access mode, no subscription being listed is a valid behavior.

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.

**4.3.5.6. Additional resources**

- 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 **PROXY CONFIGURATION** section in the **subscription-manager** man page.

**4.3.6. Installing System Aligned with a Security Policy**

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

**4.3.6.1. About security policy**

The Red Hat Enterprise Linux includes OpenSCAP suite to enable automated configuration of the system in alignment with a particular security policy. The policy is implemented using the Security Content Automation Protocol (SCAP) standard. The packages are available in the AppStream repository. However, by default, the installation and post-installation process does not enforce any policies and therefore does not involve any checks 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** and **scap-security-guide** packages are added to your package selection, providing a preinstalled tool for compliance and vulnerability scanning.

When you select a security policy, the Anaconda GUI installer requires the configuration to adhere to the policy's requirements. There might be conflicting package selections, as well as separate partitions defined. Only after all the requirements are met, you can start the installation.

At the end of the installation process, the selected OPenSCAP security policy automatically hardens the system and scans it to verify compliance, saving the scan results to the **/root/openscap\_data** directory on the installed system.

**NOTE**

By default, the installer uses the content of the **scap-security-guide** package bundled in the installation image. You can also load external content from an HTTP, HTTPS, or FTP server.

**4.3.6.2. 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.
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.

### 4.3.6.3. Additional resources

- **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.

## 4.4. CONFIGURING SOFTWARE SETTINGS

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

### 4.4.1. 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. For more information, see [Downloading a RHEL installation ISO image](#).
- You have created a bootable physical media. For more information, see [Creating a bootable CD or DVD](#).
- 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 [Configuring software selection](#).
- This option is available only when a network connection is active. See [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.
- 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**

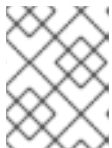
The following steps are optional and are only required if you use a proxy for network access.

- 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, 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.

#### 4.4.2. 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, for example, Server with GUI (default), Server, Minimal Install, Workstation, Custom Operating

System, Virtualization Host. The 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 8 installation* document for information about installing Red Hat Enterprise Linux using Kickstart.

#### Prerequisites

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

#### Procedure

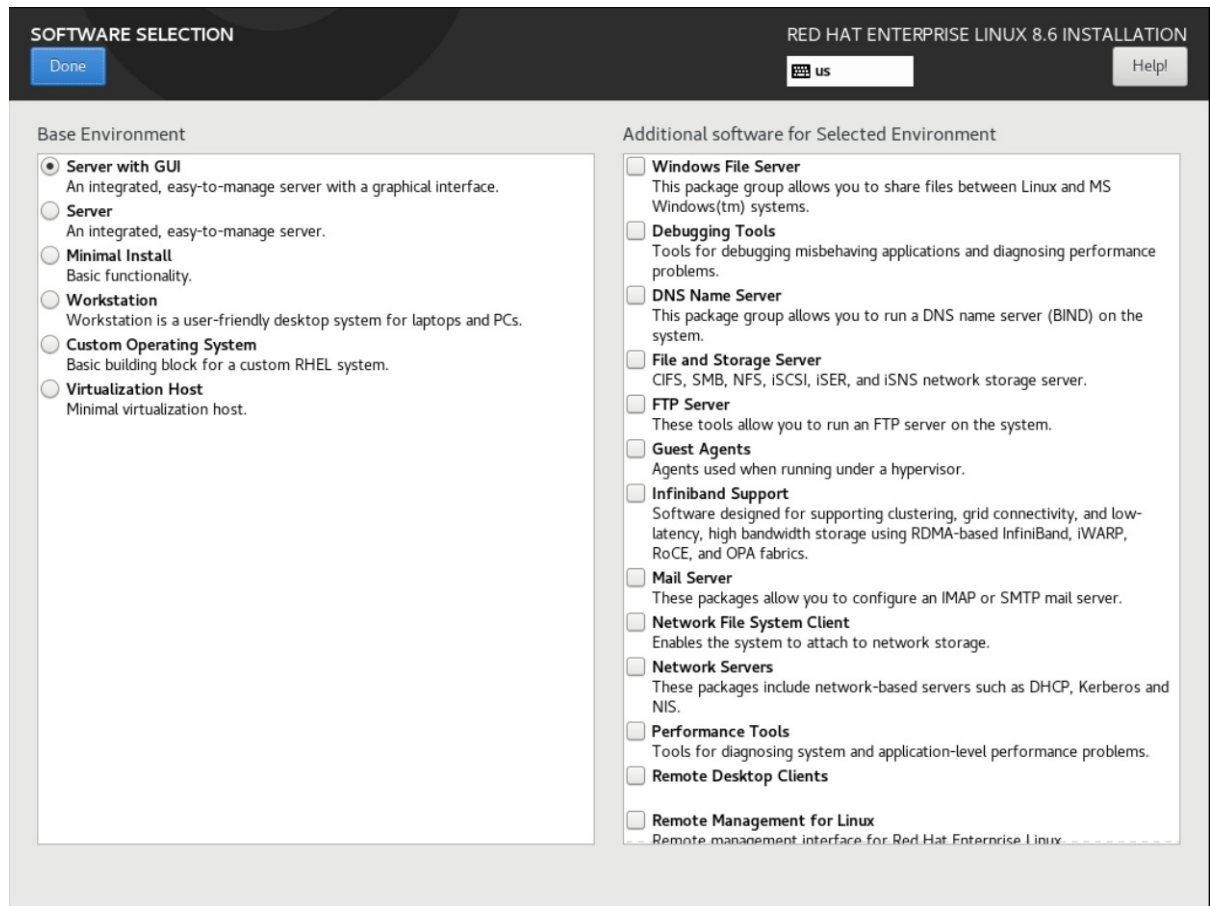
1. From the **Installation Summary** window, click **Software Selection**. The **Software Selection** window opens.
2. From the **Base Environment** pane, select a base environment. You can select only one base environment, for example, Server with GUI (default), Server, Minimal Install, Workstation, Custom Operating System, Virtualization Host.



#### 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.

Figure 4.1. Red Hat Enterprise Linux Software Selection



3. From the **Additional Software for Selected Environment** pane, select one or more options.
4. Click **Done** to apply the settings and return to [Graphical installations](#).

## 4.5. 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 64-bit 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.

### 4.5.1. 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.

## 4.5.2. 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).

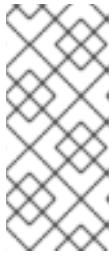
### Prerequisite

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.

### 4.5.3. 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.

#### 4.5.3.1. 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.

#### 4.5.3.2. 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 **Add 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.

### 4.5.3.3. 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.

### 4.5.3.4. Configuring FCP devices

FCP devices enable 64-bit 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 64-bit 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, you have removed 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.  
64-bit 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. When installing RHEL-8.6 or older releases or if the **zFCP** device is not configured in NPIV mode, or when **auto LUN** scanning is disabled by the **zfcplib.allow\_lun\_scan=0** kernel module parameter, provide the following values:
  - a. Type the 16 digit hexadecimal World Wide Port Number (WWPN) in the **WWPN** field.
  - b. Type the 16 digit hexadecimal FCP LUN identifier in the **LUN** field.
6. 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.

### 4.5.4. 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.

#### 4.5.4.1. 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.

#### 4.5.4.2. 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.

## 4.6. 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 on the advantages and disadvantages to using partitioning on LUNs, either directly or with LVM, see the 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: **/**, **/home**, **/boot**, and **swap**. You can also create additional partitions and volumes as you require.



### 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.

### 4.6.1. Starting manual partitioning

#### Prerequisites

- The **Installation Summary** screen is open.
- 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. Detected 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.

### 4.6.2. Adding a mount point file system

You can add multiple mount point file systems.

#### Prerequisites

- You have planned your partitions.



#### IMPORTANT

To avoid problems with space allocation, you can 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.

### 4.6.3. Configuring storage for 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.
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.



#### 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.

#### 4.6.4. Customizing a mount point file system

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 4.2. Customizing Partitions

MANUAL PARTITIONING

RED HAT ENTERPRISE LINUX 8.0 INSTALLATION  
PRE-RELEASE / TESTING

Done Help!

New Red Hat Enterprise Linux 8.0 Installation

SYSTEM

- /boot sda1 1024 MiB
- / rhel-root 17 GiB**
- swap rhel-swap 2 GiB

rhel-root

Mount Point: /

Device(s): ATA QEMU HARDDISK (sda)

Desired Capacity: 17 GiB

Device Type: LVM  Encrypt

Volume Group: rhel (0 B free)

File System: xfs  Reformat

Label:

Name: root

Update Settings

Note: The settings you make on this screen will not be applied until you click on the main menu's 'Begin Installation' button.

AVAILABLE SPACE: 1023 KiB TOTAL SPACE: 20 GiB

1 storage device selected

Reset All

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 clear the **Reformat** check box to retain your data. The newly-created partitions and volumes must be reformatted, and the check box cannot be cleared.
- 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.

#### 4.6.5. Preserving the `/home` directory

In a Red Hat Enterprise Linux 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 Red Hat Enterprise Linux 8 system is set in the same way as it was on your RHEL 7 system. Note that this applies only for users on Red Hat Enterprise Linux 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.

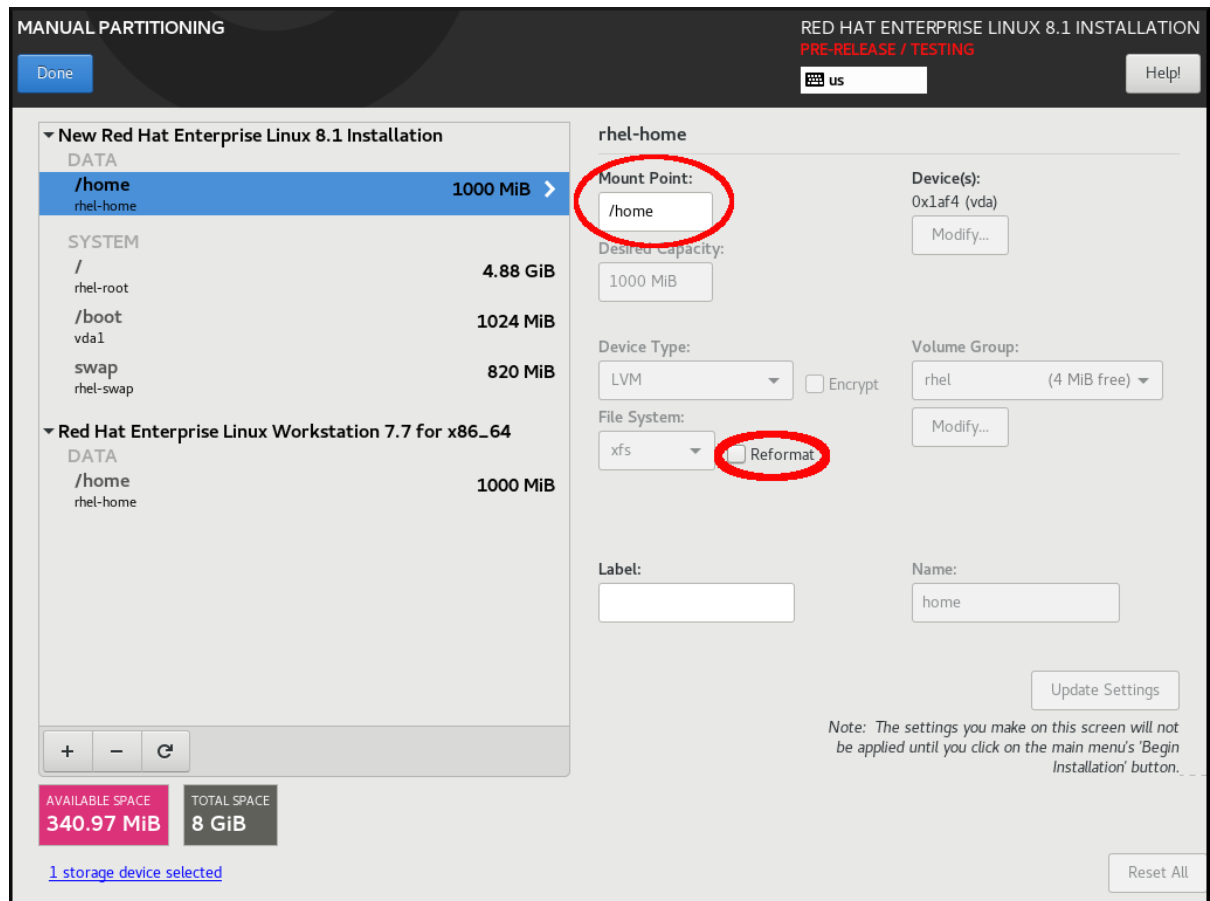
**Prerequisites**

- You have RHEL 7 installed on your computer.
- The **/home** directory is located on a separate **/home** partition on your RHEL 7 system.
- The Red Hat Enterprise Linux 8 **Installation Summary** window is open.

**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 4.3. Ensuring that /home is not formatted



5. Optional: You can also customize various aspects of the **/home** partition required for your Red Hat Enterprise Linux 8 system as described in [Customizing a mount point file system](#). 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.

#### 4.6.6. Creating a software RAID during the installation

Redundant Arrays of Independent Disks (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 that 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 64-bit 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. Depending on the RAID type you want to create, at least two disks are required.
- You have created a mount point. By configuring a mount point, you can 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 to return to the **Installation Summary** window.

## Additional resources

- [Creating a RAID LV with DM integrity](#)

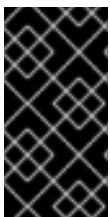
### 4.6.7. Creating an LVM logical volume

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

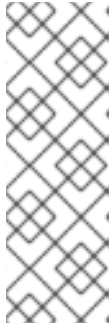
During text-mode installation, LVM configuration is not available. To create an LVM configuration, press **Ctrl+Alt+F2** to use a shell prompt in a different virtual console. You can run **vgcreate** and **lv** commands in this shell. To return to the text-mode installation, press **Ctrl+Alt+F1**.

## Procedure

1. From the left-hand pane of the **Manual Partitioning** window, select the mount point.



- 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 8 installation](#) document for more information about Kickstart.

#### Additional resources

- [Configuring and managing logical volumes](#)

### 4.6.8. Configuring an LVM logical volume

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

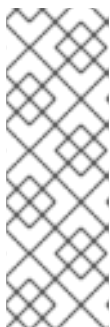


#### WARNING

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 8 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.

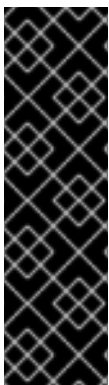
6. From the **Size policy** drop-down menu, select the size policy for the volume group.  
The available policy options are:
  - **Automatic:** The size of the volume group is set automatically so that it is large enough to contain the configured logical volumes. This is optimal if you do not need free 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.

#### Additional resources

- [How to use dm-crypt on IBM Z, LinuxONE and with the PAES cipher](#)

## 4.7. 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 **Installation Summary** window, select **User Settings > 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 the **Installation Summary** window.

**NOTE**

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

**4.8. 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. On the **Installation Summary** window, select **User Settings > 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. Click **Done** to apply the changes and return to the **Installation Summary** window.

## 4.9. EDITING ADVANCED USER SETTINGS

This procedure describes how to edit the default settings for the user account in the **Advanced User Configuration** dialog box.

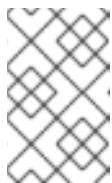
### Procedure

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

4. 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).

5. Click **Save Changes** to apply the updates and return to the **Create User** window.

## CHAPTER 5. 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 [Registering and installing RHEL from the CDN](#) for more information.

- Securing your system

### 5.1. 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 [Installing RHEL using an ISO image from the Customer Portal](#).
- 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.

- 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 [Subscription manager post installation](#) for more information. If you configured network settings, as described in [Network hostname](#), you can register your system immediately, as shown in the following steps:

- 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.

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

**Additional resources**

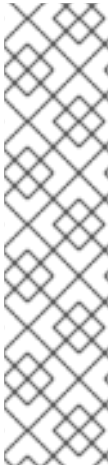
Depending on your requirements, there are five methods to register your system:

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

- After installation using the Subscription Manager user interface. See [Subscription manager post install UI](#) 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.

## 5.2. REGISTERING YOUR SYSTEM USING THE COMMAND LINE

This section contains information about how to register your Red Hat Enterprise Linux 8 subscription 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.
- For an improved and simplified experience registering your hosts to Red Hat, use remote host configuration (RHC). The RHC client registers your system to Red Hat Insights and Red Hat Subscription Manager, making your system ready for Insights data collection and enabling direct issue remediation from Insights for Red Hat Enterprise Linux. For more information, see [RHC registration](#) and [remediation using Insights](#).

### 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 as root.

### Procedure

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

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

2. When the system 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 Red Hat Enterprise Linux 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 --auto
```

7. When a 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

An alternative method for registering your Red Hat Enterprise Linux 8 system is by logging in to the system as a **root** user and using the Subscription Manager graphical user interface.

## 5.3. 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 [Installing RHEL using an ISO image from the Customer Portal](#) .
- 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 from a terminal window using the **subscription-manager status** command.

**Additional resources**

- [Using and Configuring Red Hat Subscription Manager](#)
- [Configuring Virtual Machine Subscriptions in Red Hat Subscription Management](#)

## 5.4. REGISTERING RHEL 8 USING THE INSTALLER GUI

Use the following steps to register a newly installed Red Hat Enterprise Linux 8 using the RHEL installer GUI.

**Prerequisites**

- You have a valid user account on the Red Hat Customer Portal. See the [Create a Red Hat Login page](#).  
If the user account has appropriate entitlements (or the account operates in Simple Content Access mode) they can register using username and password only, without presenting an activation key.
- You have a valid Activation Key and Organization id.

**Procedure**

1. Authenticate your Red Hat account using the **Account** or **Activation Key** option.
2. Select the **Set System Purpose** field and from the drop-down menu select the **Role, SLA, and Usage for the RHEL 8 installation**  
At this point, your Red Hat Enterprise Linux 8 system has been successfully registered.

## 5.5. 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.

## 5.6. CONFIGURING SYSTEM PURPOSE USING THE SUBSCRIPTION-MANAGER COMMAND-LINE TOOL

System Purpose is an optional but recommended feature of the Red Hat Enterprise Linux installation. You can 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. If System Purpose was not configured during the installation process, you can use the **subscription-manager syspurpose** command-line tool after installation to set the required attributes.

### Prerequisites

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



### 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 command-line `subscription-manager syspurpose {role, usage, service-level}` tools to set the required purpose attributes, and lastly run `subscription-manager attach --auto` to re-entitle the system with considerations for the updated attributes.

### Procedure

Complete the steps in this procedure to configure System Purpose after installation using the **subscription-manager 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:

```
# subscription-manager syspurpose role --set "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:

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

- a. Optional: Before setting a value, see the available roles supported by the subscriptions for your organization:

```
# subscription-manager syspurpose role --list
```

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

```
# subscription-manager syspurpose role --unset
```

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

```
# subscription-manager syspurpose service-level --set "VALUE"
```

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

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

For example:

```
# subscription-manager syspurpose service-level --set "Standard"
```

- a. Optional: Before setting a value, see the available service-levels supported by the subscriptions for your organization:

```
# subscription-manager syspurpose service-level --list
```

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

```
# subscription-manager syspurpose service-level --unset
```

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

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

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

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

For example:

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

- a. Optional: Before setting a value, see the available usages supported by the subscriptions for your organization:

```
# subscription-manager syspurpose usage --list
```

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

```
# subscription-manager syspurpose usage --unset
```

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

```
# subscription-manager syspurpose --show
```

- a. Optional: For more detailed syntax information run the following command to access the **subscription-manager** man page and browse to the SYSPURPOSE OPTIONS:

```
# man subscription-manager
```

### Verification steps

- To verify the system's subscription status:

```
# subscription-manager status
+-----+
System Status Details
+-----+
Overall Status: Current

System Purpose Status: Matched
```

- An overall status **Current** means that all of the installed products are covered by the subscription(s) attached and entitlements to access their content set repositories has been granted.
- A system purpose status **Matched** means that all of the system purpose attributes (role, usage, service-level) that were set on the system are satisfied by the subscription(s) attached.
- When the status information is not ideal, additional information is displayed to help the system administrator decide what corrections to make to the attached subscriptions to cover the installed products and intended system purpose.

## 5.7. 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
```

- 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
```

- 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
```

## 5.8. DEPLOYING SYSTEMS THAT ARE COMPLIANT WITH A SECURITY PROFILE IMMEDIATELY AFTER AN INSTALLATION

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

### 5.8.1. Profiles not compatible with Server with GUI

Certain security profiles provided as part of the **SCAP Security Guide** are not compatible with the extended package set included in the **Server with GUI** base environment. Therefore, do not select **Server with GUI** when installing systems compliant with one of the following profiles:

Table 5.1. Profiles not compatible with Server with GUI

Profile name	Profile ID	Justification	Notes
CIS Red Hat Enterprise Linux 8 Benchmark for Level 2 - Server	<b>xccdf_org.ssgproject.content_profile_cis</b>	Packages <b>xorg-x11-server-Xorg</b> , <b>xorg-x11-server-common</b> , <b>xorg-x11-server-utils</b> , and <b>xorg-x11-server-Xwayland</b> are part of the <b>Server with GUI</b> package set, but the policy requires their removal.	

Profile name	Profile ID	Justification	Notes
CIS Red Hat Enterprise Linux 8 Benchmark for Level 1 - Server	<b>xccdf_org.ssgproject.content_profile_cis_server_l1</b>	Packages <b>xorg-x11-server-Xorg</b> , <b>xorg-x11-server-common</b> , <b>xorg-x11-server-utils</b> , and <b>xorg-x11-server-Xwayland</b> are part of the <b>Server with GUI</b> package set, but the policy requires their removal.	
Unclassified Information in Non-federal Information Systems and Organizations (NIST 800-171)	<b>xccdf_org.ssgproject.content_profile_cui</b>	The <b>nfs-utils</b> package is part of the <b>Server with GUI</b> package set, but the policy requires its removal.	
Protection Profile for General Purpose Operating Systems	<b>xccdf_org.ssgproject.content_profile_ospp</b>	The <b>nfs-utils</b> package is part of the <b>Server with GUI</b> package set, but the policy requires its removal.	
DISA STIG for Red Hat Enterprise Linux 8	<b>xccdf_org.ssgproject.content_profile_stig</b>	Packages <b>xorg-x11-server-Xorg</b> , <b>xorg-x11-server-common</b> , <b>xorg-x11-server-utils</b> , and <b>xorg-x11-server-Xwayland</b> are part of the <b>Server with GUI</b> package set, but the policy requires their removal.	To install a RHEL system as a <b>Server with GUI</b> aligned with DISA STIG in RHEL version 8.4 and later, you can use the <b>DISA STIG with GUI</b> profile.

## 5.8.2. Deploying baseline-compliant RHEL systems using the graphical installation

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



### WARNING

Certain security profiles provided as part of the **SCAP Security Guide** are not compatible with the extended package set included in the **Server with GUI** base environment. For additional details, see [Profiles not compatible with a GUI server](#) .

## Prerequisites

- You have booted into the **graphical** installation program. Note that the **OSCAP Anaconda Add-on** does not support interactive text-only installation.
- 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.
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

- To check the current status of the system after installation is complete, reboot the system and start a new scan:

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

## Additional resources

- [Configuring manual partitioning](#)

### 5.8.3. Deploying baseline-compliant RHEL systems using Kickstart

Use this procedure to deploy RHEL systems that are aligned with a specific baseline. This example uses 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/kickstart/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.
3. Start a Kickstart installation as described in [Performing an automated installation using Kickstart](#).



### IMPORTANT

Passwords in Kickstart files are not checked for OSPP requirements.

## Verification

1. To check the current status of the system after installation is complete, reboot the system and start a new scan:

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

## Additional resources

- [OSCAP Anaconda Add-on](#)

## 5.9. NEXT STEPS

When you have completed the required post-installation steps, you can configure basic system settings. For information about completing tasks such as installing software with yum, using systemd for service management, managing users, groups, and file permissions, using chrony to configure NTP, and working with Python 3, see the [Configuring basic system settings](#) document.

## APPENDIX A. TROUBLESHOOTING

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

## APPENDIX B. TOOLS AND TIPS FOR TROUBLESHOOTING AND BUG REPORTING

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.

### B.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.

### B.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 B.1. Log files generated during the installation

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

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.

#### B.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 open.

### 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/
```

### Additional resources

- [Boot options reference](#)
- [Console logging during installation](#)

## B.2.2. Transferring installation log files to a USB drive

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

### Prerequisites

- You have backed up data from the USB drive.
- 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
```

- Note the name of the connected device. In the above example, it is **sdb**.
- 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
```

- 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
```

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

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

```
# ls
```

- Copy the log files to the mounted device.

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

- 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
```

### B.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

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

```
# cd /tmp
```

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

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

- 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.

### B.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.

#### B.3.1. Running Memtest86

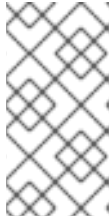
Use this procedure to run the **Memtest86** application to test your system's memory for faults before you install Red Hat Enterprise Linux.

##### 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**

- [How to use Memtest86](#)

**B.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.

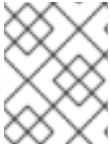
**B.5. Consoles and logging during installation**

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 B.2. Available tmux windows**

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

## B.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**.

## B.7. Display settings and device drivers

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 [Console boot options](#).

**Table B.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 <b>Troubleshooting &gt; Install Red Hat Enterprise Linux in basic graphics mode</b> from the boot menu, or edit the installation program's boot options and append <b>inst.xdriver=vesa</b> 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 <b>inst.resolution=x</b> option at the boot menu, where x is your display's resolution, for example, 1024x768.
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 <b>inst.xdriver=x</b> option, where x is the device driver you want to use (for example, nouveau)*.
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 <a href="#">Performing a remote RHEL installation using VNC</a> section of the <i>Performing an advanced RHEL 8 installation</i> 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.

## B.8. 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.1. 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.1.1. 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 8 installation* document.

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

### A.1.2. 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

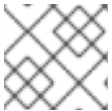
The graphical installation program encountered an error and displayed the **unknown error** dialog box.

#### 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.1.3. 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)

#### Additional resources

- See [Recommended Partitioning Scheme](#)

## APPENDIX C. TROUBLESHOOTING

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.

### C.1. 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

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-x.x-x86_64-dvd.iso
85a...46c rhel-x.x-x86_64-dvd.iso`
```

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

#### Example C.1. Resuming an interrupted download attempt

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

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

### C.2. 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 8 installation* document.

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

### C.3. 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.

### C.4. 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.

### C.5. 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](http://www.linfo.org/x_server.html)

### C.6. 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. If the total amount of memory that your system reports 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.

Use this procedure to manually set the amount of memory.

### Procedure

1. Check the amount of memory that your system currently reports in MiB:

```
$ free -m
```

2. Reboot your system and wait until the boot loader menu is displayed.  
If your boot timeout period is set to **0**, press the **Esc** key to access the menu.
3. 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.
4. 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
```

5. Replace **xx** with the amount of RAM you have in MiB.
6. Press **F10** or **Ctrl+X** to boot your system with the edited options.
7. Wait for the system to boot, log in, and open a command line.
8. Check the amount of memory that your system reports in MiB:

```
$ free -m
```

9. If the total amount of RAM displayed by the command now matches your expectations, make the change permanent:

```
# grubby --update-kernel=ALL --args="mem=xxM"
```

## C.7. 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 [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 [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 [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>.

## C.8. Unable to IPL from network storage space



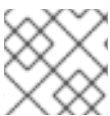
#### NOTE

- This issue is for IBM Power Systems.
- The **PReP** Boot partitions are not required on PowerNV 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.

## C.9. 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.

#### 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. If you are using XDMCP, ensure that **WaylandEnable=false** is present in the **/etc/gdm/custom.conf** file.
4. Save the file and exit the text editor.
5. 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
```

6. 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
```

7. 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
```

### Additional resources

- [X Window System documentation](#)

## C.10. 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 64-bit 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.

### C.10.1. Booting into rescue mode

This procedure describes how 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/sysroot/. 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/sysroot/**. 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/sysroot/**, but in read-only mode. If you select **3**, your file system is not mounted.

For the system root, the installer supports two mount points **/mnt/sysimage** and **/mnt/sysroot**. The **/mnt/sysroot** path is used to mount **/** of the target system. Usually, the physical root and the system root are the same, so **/mnt/sysroot** is attached to the same file system as

**/mnt/sysimage**. The only exceptions are rpm-ostree systems, where the system root changes based on the deployment. Then, **/mnt/sysroot** is attached to a subdirectory of **/mnt/sysimage**. It is recommended to use **/mnt/sysroot** for chroot.

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#
```

7. 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/sysroot
```

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.

8. 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).

9. 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 **pvdisk**, **vgdisplay** or **lvdisplay** commands.

## C.10.2. 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.

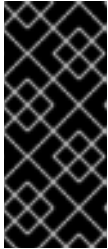
### Procedure

1. Change the root directory to the **/mnt/sysroot/** directory:

```
sh-4.2# chroot /mnt/sysroot/
```

2. 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/sysroot/var/tmp/sosreport new_location
```

6. For transferring the archive through the network, use the **scp** utility:

```
sh-4.2# scp /mnt/sysroot/var/tmp/sosreport username@hostname:sosreport
```

#### Additional resources

- [What is an sosreport and how to create one in Red Hat Enterprise Linux?](#)
- [How to generate sosreport from the rescue environment](#)
- [How do I make sosreport write to an alternative location?](#)
- [Sosreport fails. What data should I provide in its place?](#)

### C.10.3. 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 (MBR) on AMD64 and Intel 64 systems with BIOS, or on the little-endian variants of IBM Power Systems with Open

Firmware.

### Prerequisites

- You have booted into rescue mode.
- You have mounted the installed system / **(root)** partition in read-write mode.
- You have mounted the **/boot** mount point in read-write mode.

### Procedure

1. Change the root partition:

```
sh-4.2# chroot /mnt/sysroot/
```

2. Reinstall the GRUB2 boot loader, where the **install\_device** block device was installed:

```
sh-4.2# /sbin/grub2-install install_device
```



#### IMPORTANT

Running the **grub2-install** command could lead to the machine being unbootable if all the following conditions apply:

- The system is an AMD64 or Intel 64 with Extensible Firmware Interface (EFI).
- Secure Boot is enabled.

After you run the **grub2-install** command, you cannot boot the AMD64 or Intel 64 systems that have Extensible Firmware Interface (EFI) and Secure Boot enabled. This issue occurs because the **grub2-install** command installs an unsigned GRUB2 image that boots directly instead of using the shim application. When the system boots, the shim application validates the image signature, which when not found fails to boot the system.

3. Reboot the system.

## C.10.4. 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.

### C.10.4.1. 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.

### Procedure

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/sysroot/**, for example: **/mnt/sysroot/root/drivers/**.
2. Change the root directory to **/mnt/sysroot/**:

```
sh-4.2# chroot /mnt/sysroot/
```

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/sysroot/root/drivers/** directory in the original rescue environment.

4. Exit the chroot environment:

```
sh-4.2# exit
```

#### C.10.4.2. 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.

### Procedure

1. Change the root directory to the **/mnt/sysroot/** directory:

```
sh-4.2# chroot /mnt/sysroot/
```

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.

### C.11. 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.

#### Additional resources

- [Network boot options](#)

### C.12. Cannot boot into the graphical installation on iLO or iDRAC devices

The graphical installer for a remote ISO installation on iLO or iDRAC devices may not be available due to a slow internet connection. To proceed with the installation in this case, you can choose one of the following methods:

1. Avoid the timeout. To do so:
  - a. Press the **Tab** key in case of BIOS usage, or the **e** key in case of UEFI usage when booting from an installation media. That will allow you to modify the kernel command line arguments.
  - b. To proceed with the installation, append the **rd.live.ram=1** and press **Enter** in case of BIOS usage, or **Ctrl+x** in case of UEFI usage. This might take longer time to load the installation program.



- Another option to extend the loading time for the graphical installer is to set the **inst.xtimeout** kernel argument in seconds.

```
inst.xtimeout=N
```

- You can install the system in text mode. For more details, see [Installing RHEL8 in text mode](#).
- In the remote management console, such as iLO or iDRAC, instead of a local media source, use the direct URL to the installation ISO file from the [Download center](#) on the Red Hat Customer Portal. You must be logged in to access this section.

### C.13. Rootfs image is not initramfs

If you get the following message on the console during booting the installer, the transfer of the installer **initrd.img** might have had errors:

```
[...] rootfs image is not initramfs
```

To resolve this issue, download **initrd** again or run the **sha256sum** with **initrd.img** and compare it with the checksum stored in the **.treeinfo** file on the installation medium, for example,

```
$ sha256sum dvd/images/pxeboot/initrd.img
fdb1a70321c06e25a1ed6bf3d8779371b768d5972078eb72b2c78c925067b5d8
dvd/images/pxeboot/initrd.img
```

To view the checksum in **.treeinfo**:

```
$ grep sha256 dvd/.treeinfo
images/efiboot.img =
sha256:d357d5063b96226d643c41c9025529554a422acb43a4394e4ebcaa779cc7a917
images/install.img =
sha256:8c0323572f7fc04e34dd81c97d008a2ddfc2cfc525aef8c31459e21bf3397514
images/pxeboot/initrd.img =
sha256:db1a70321c06e25a1ed6bf3d8779371b768d5972078eb72b2c78c925067b5d8
images/pxeboot/vmlinuz =
sha256:b9510ea4212220e85351cbb7f2ebc2b1b0804a6d40ccb93307c165e16d1095db
```

Despite having correct **initrd.img**, if you get the following kernel messages during booting the installer, often a boot parameter is missing or mis-spelled, and the installer could not load **stage2**, typically referred to by the **inst.repo=** parameter, providing the full installer initial ramdisk for its in-memory root file system:

```
[...] No filesystem could mount root, tried:
[...] Kernel panic - not syncing: VFS: Unable to mount root fs on unknown-block(1,0)
[...] CPU: 0 PID: 1 Comm: swapper/0 Not tainted 5.14.0-55.el9.s390x #1
[...] ...
[...] Call Trace:
[...] ([<...>] show_trace+0x.../0x...)
[...] [<...>] show_stack+0x.../0x...
[...] [<...>] panic+0x.../0x...
[...] [<...>] mount_block_root+0x.../0x...
[...] [<...>] prepare_namespace+0x.../0x...
[...] [<...>] kernel_init_freeable+0x.../0x...
```

```
[...] [<...>] kernel_init+0x.../0x...  
[...] [<...>] kernel_thread_starter+0x.../0x...  
[...] [<...>] kernel_thread_starter+0x.../0x...
```

To resolve this issue, check

- if the installation source specified is correct on the kernel command line (**inst.repo=**) or in the kickstart file
- the network configuration is specified on the kernel command line (if the installation source is specified as network)
- the network installation source is accessible from another system

## APPENDIX D. SYSTEM REQUIREMENTS REFERENCE

This section provides information and guidelines for hardware, installation target, system, memory, and RAID when installing Red Hat Enterprise Linux.

### D.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.

### D.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](#).

### D.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 [Configuring software settings](#) 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.

## D.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.
- The **Prep** Boot partitions are not required on PowerNV systems.

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 [Partitioning reference](#)

**Table D.1. Minimum RAM requirements**

| Installation type                       | Recommended minimum RAM                                                                                                                             |
|-----------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| Local media installation (USB, DVD)     | <ul style="list-style-type: none"> <li>• 1.5 GiB for aarch64, s390x and x86_64 architectures</li> <li>• 3 GiB for ppc64le architecture</li> </ul>   |
| NFS network installation                | <ul style="list-style-type: none"> <li>• 1.5 GiB for aarch64, s390x and x86_64 architectures</li> <li>• 3 GiB for ppc64le architecture</li> </ul>   |
| HTTP, HTTPS or FTP network installation | <ul style="list-style-type: none"> <li>• 3 GiB for s390x and x86_64 architectures</li> <li>• 4 GiB for aarch64 and ppc64le architectures</li> </ul> |



#### 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 8 installation](#) document for more information.

## D.5. UEFI SECURE BOOT AND BETA RELEASE REQUIREMENTS

If you plan to install a Beta release of Red Hat Enterprise Linux, on systems having UEFI Secure Boot enabled, then first disable the UEFI Secure Boot option and then begin the installation.

UEFI Secure Boot requires that the operating system kernel is signed with a recognized private key, which the system's firmware verifies using the corresponding public key. For Red Hat Enterprise Linux Beta releases, the kernel is signed with a Red Hat Beta-specific public key, which the system fails to recognize by default. As a result, the system fails to even boot the installation media.

## APPENDIX E. PARTITIONING REFERENCE

### E.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. 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.

### E.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. The XFS filesystem cannot be shrunk to get free space.

#### 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.



#### NOTE

- The **PReP** Boot partitions are not required on PowerNV systems.

## E.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.

### 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.

## E.4. RECOMMENDED PARTITIONING SCHEME

Red Hat recommends that you create separate file systems at the following mount points. However, if required, you can also create the file systems at **/usr**, **/var**, and **/tmp** mount points.

- **/boot**
- **/** (root)
- **/home**
- **swap**
- **/boot/efi**
- **PReP**

This partition scheme is recommended for bare metal deployments and it does not apply to virtual and cloud deployments.

#### **/boot** partition - recommended size at least 1 GiB

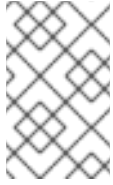
The partition mounted on **/boot** contains the operating system kernel, which allows your system to boot Red Hat Enterprise Linux 8, along with files used during the bootstrap process. Due to the limitations of most firmwares, creating a small partition to hold these is recommended. In most scenarios, 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 GiB**

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 1TiB. 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 1TiB, you must edit the partitioning layout manually.

**Table E.1. Recommended system swap space**

| Amount of RAM in the system | Recommended swap space               | Recommended swap space if allowing for hibernation |
|-----------------------------|--------------------------------------|----------------------------------------------------|
| Less than 2 GiB             | 2 times the amount of RAM            | 3 times the amount of RAM                          |
| 2 GiB - 8 GiB               | Equal to the amount of RAM           | 2 times the amount of RAM                          |
| 8 GiB - 64 GiB              | 4 GiB to 0.5 times the amount of RAM | 1.5 times the amount of RAM                        |
| More than 64 GiB            | Workload dependent (at least 4GiB)   | Hibernation not recommended                        |

### **/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.

At the border between each range, for example, a system with 2 GiB, 8 GiB, or 64 GiB 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.

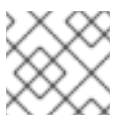


#### **NOTE**

- 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.



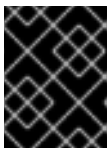
#### **NOTE**

- The **PreP** Boot partitions are not required on PowerNV systems.

## **E.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** allows you to preserve the database during a re-installation 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](#) for information about various system directories and their contents.
- Each kernel requires approximately: 60MiB (initrd 34MiB, 11MiB vmlinuz, and 5MiB System.map)
- For rescue mode: 100MiB (initrd 76MiB, 11MiB vmlinuz, and 5MiB System map)
- When **kdump** is enabled in system it will take approximately another 40MiB (another initrd with 33MiB)  
The default partition size of 1 GiB 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 YUM package manager to temporarily store downloaded package updates. Make sure that the partition or volume containing **/var** has at least 5 GiB.
- 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 GiB for minimal installations, and at least 10 GiB 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 under them. For example, a separate partition for **/var/www** works without issues.



### IMPORTANT

Some security policies require the separation of **/usr** and **/var**, even though it makes administration more complex.

- 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 cannot 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 plan to alter 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. Note that you cannot create a BIOS Boot or EFI System Partition in graphical installation if your system does **not** require one - in that case, they are 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**.

### Additional resources

- [How to use dm-crypt on IBM Z, LinuxONE and with the PAES cipher](#)

## E.6. SUPPORTED HARDWARE STORAGE

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. 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 F. 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 [Boot options for RHEL installer](#) document.

### F.1. INSTALLATION SOURCE BOOT OPTIONS

This section describes 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.

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 F.1. Types and format for the inst.repo= boot option and installation source**

| Source type                          | Boot option format                                                                 | Source format                                                                                                                                              |
|--------------------------------------|------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CD/DVD drive                         | <b>inst.repo=cdrom:&lt;device&gt;</b>                                              | Installation DVD as a physical disk. <sup>[a]</sup>                                                                                                        |
| Mountable device (HDD and USB stick) | <b>inst.repo=hd:&lt;device&gt;:/&lt;path&gt;</b>                                   | Image file of the installation DVD.                                                                                                                        |
| NFS Server                           | <b>inst.repo=nfs:<br/>[options:]&lt;server&gt;:/&lt;path&gt;</b>                   | Image file of the installation DVD, or an installation tree, which is a complete copy of the directories and files on the installation DVD. <sup>[b]</sup> |
| HTTP Server                          | <b>inst.repo=http://&lt;host&gt;/&lt;path&gt;</b>                                  | Installation tree that is a complete copy of the directories and files on the installation DVD.                                                            |
| HTTPS Server                         | <b>inst.repo=https://&lt;host&gt;/&lt;path&gt;</b>                                 |                                                                                                                                                            |
| FTP Server                           | <b>inst.repo=ftp://&lt;username&gt;:&lt;password&gt;@&lt;host&gt;/&lt;path&gt;</b> |                                                                                                                                                            |
| HMC                                  | <b>inst.repo=hmc</b>                                                               |                                                                                                                                                            |

| Source type | Boot option format                                                                                                                                                                                  | Source format |
|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|
| [a]         | If <i>device</i> 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, add <b>nfsvers=X</b> to <i>options</i> , replacing <i>X</i> with the version number that you want to use. |               |

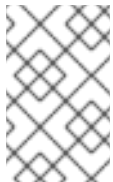
Set disk device names with the following formats:

- 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 you can use 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.

For more information about unified ISO, see [Unified ISO](#).

**Table F.2. Installation sources and boot option format**

| Installation source             | Boot option format                                                          | Additional information                                                                                                                                                                                                              |
|---------------------------------|-----------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Installable tree at a URL       | <b>inst.addrepo=REPO_NAME, [http,https,ftp]://&lt;host&gt;/&lt;path&gt;</b> | Looks for the installable tree at a given URL.                                                                                                                                                                                      |
| Installable tree at an NFS path | <b>inst.addrepo=REPO_NAME,nfs://&lt;server&gt;:/&lt;path&gt;</b>            | Looks for the installable tree at a given NFS path. A colon is required after the host. The installation program passes everything after <b>nfs://</b> 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 | <b>inst.addrepo=REPO_NAME,file:/// &lt;path&gt;</b>          | 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 <b>/run/install/source/REPO_ISO_PATH</b> . Additionally, you can mount the repository directory in the <b>%pre</b> section in the Kickstart file. The path must be absolute and start with <b>/</b> , for example <b>inst.addrepo=REPO_NAME,file:/// &lt;path&gt;</b> |
| Hard Drive                                       | <b>inst.addrepo=REPO_NAME,hd:&lt;device&gt;:&lt;path&gt;</b> | Mounts the given <i>&lt;device&gt;</i> partition and installs from the ISO that is specified by the <i>&lt;path&gt;</i> . If the <i>&lt;path&gt;</i> is not specified, the installation program looks for a valid installation ISO on the <i>&lt;device&gt;</i> . This installation method requires an ISO with a valid installable tree.                                                                                                                                                                                                                                                                                                                                                                                   |

### inst.stage2=

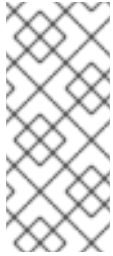
The **inst.stage2=** boot option specifies the location of the installation program's runtime image. This option expects the path to a directory that contains a valid **.treeinfo** file and reads the runtime image location from the **.treeinfo** file. If the **.treeinfo** file is not available, the installation program attempts to load the image from **images/install.img**.

When you do not specify the **inst.stage2** option, the installation program attempts to use the location specified with the **inst.repo** option.

Use this option when you want to manually specify the installation source in the installation program at a later time. For example, when you want to select the Content Delivery Network (CDN) as an installation source. The installation DVD and Boot ISO already contain a suitable **inst.stage2** option to boot the installation program from the respective ISO.

If you want to specify an installation source, use the **inst.repo=** option instead.





## 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-x-0-0-BaseOS-x86\_64**. If you modify the default label of the file system that contains the runtime image, or if you use a customized procedure to boot the installation system, verify that the **inst.stage2=** boot option is set to the correct value.

### inst.noverifyssl

Use the **inst.noverifyssl** boot option to prevent the installer from verifying SSL certificates for all HTTPS connections with the exception of additional Kickstart repositories, where **--noverifyssl** can be set per repository.

For example, if your remote installation source is using self-signed SSL certificates, the **inst.noverifyssl** boot option enables the installer to complete the installation without verifying the SSL certificates.

#### Example when specifying the source using **inst.stage2=**

```
inst.stage2=https://hostname/path_to_install_image/ inst.noverifyssl
```

#### Example when specifying the source using **inst.repo=**

```
inst.repo=https://hostname/path_to_install_repository/ inst.noverifyssl
```

### inst.stage2.all

Use the **inst.stage2.all** boot option 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. For more information on how to update drivers during installation, see the [Performing an advanced RHEL 8 installation](#) document.

### inst.repo=hmc

This option eliminates the requirement of an external network setup and expands the installation options. 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 the **inst.repo=hmc** option to the kernel parameters. The installation program then enables support element (SE) and hardware management console (HMC) file access, fetches the images for stage2 from the DVD, and provides access to the packages on the DVD for software selection.

### inst.proxy=

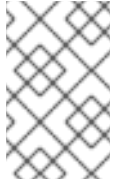
The **inst.proxy=** boot option is used when performing an installation from a HTTP, HTTPS, and FTP protocol. For example:

```
[PROTOCOL://][USERNAME[:PASSWORD]@]HOST[:PORT]
```

**inst.nosave=**

Use the **inst.nosave=** boot option to control the installation logs and related files that are not saved to the installed system, for example **input\_ks**, **output\_ks**, **all\_ks**, **logs** and **all**. You can combine multiple values separated by a comma. For example,

```
inst.nosave=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.

**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**.

**inst.memcheck**

The **inst.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.

**inst.nomemcheck**

The **inst.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.

## F.2. NETWORK BOOT OPTIONS

If your scenario requires booting from an image over the network instead of booting from a local image, you can use the following options to customize network booting.

**NOTE**

Initialize the network with the **dracut** tool. For complete list of **dracut** options, see the **dracut.cmdline(7)** man page.

**ip=**

Use the **ip=** boot option to configure one or more network interfaces. To configure multiple interfaces, use one of the following methods;

- use the **ip** option multiple times, once for each interface; to do so, use the **rd.neednet=1** option, and specify a primary boot interface using the **bootdev** option.
- 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.

In the following tables:

- The **ip** parameter specifies the client IP address and **IPv6** requires square brackets, for example 192.0.2.1 or [2001:db8::99].
- The **gateway** parameter is the default gateway. **IPv6** requires square brackets.
- 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 F.3. Boot option formats to configure the network interface**

| Boot option format                                          | Configuration method                                                                                                                                                                                                   |
|-------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>ip=method</b>                                            | Automatic configuration of any interface                                                                                                                                                                               |
| <b>ip=interface:method</b>                                  | Automatic configuration of a specific interface                                                                                                                                                                        |
| <b>ip=ip::gateway:netmask:hostname:interface:none</b>       | Static configuration, for example, IPv4:<br><b>ip=192.0.2.1::192.0.2.254:255.255.255.0:server.example.com:enp1s0:none</b><br><br>IPv6: <b>ip=[2001:db8::1]::<br/>[2001:db8::ffe]:64:server.example.com:enp1s0:none</b> |
| <b>ip=ip::gateway:netmask:hostname:interface:method:mtu</b> | Automatic configuration of a specific interface with an override                                                                                                                                                       |

### Configuration methods for the automatic interface

The method **automatic configuration of a specific interface with an override** opens 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:

#### 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 default value of the **ip** option is **ip=dhcp**.
- To connect to an iSCSI target automatically, activate a network device for accessing the target by using 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.

**vlan=**

Use the **vlan=** option to configure a Virtual LAN (VLAN) device on a specified interface with a given name. The syntax is **vlan=name:interface**. For example:

```
vlan=vlan5:enp0s1
```

This configures a VLAN device named **vlan5** on the **enp0s1** interface. The name can take the following forms:

- VLAN\_PLUS\_VID: **vlan0005**
- VLAN\_PLUS\_VID\_NO\_PAD: **vlan5**
- DEV\_PLUS\_VID: **enp0s1.0005**
- DEV\_PLUS\_VID\_NO\_PAD: **enp0s1.5**

**bond=**

Use the **bond=** option to configure a bonding device with the following syntax: **bond=name[:interfaces][:options]**. Replace *name* with the bonding device name, *interfaces* with a comma-separated list of physical (Ethernet) interfaces, and *options* with a comma-separated list of bonding options. For example:

```
bond=bond0:enp0s1,enp0s2:mode=active-backup,tx_queues=32,downdelay=5000
```

For a list of available options, execute the **modinfo** bonding command.

**team=**

Use the **team=** option to configure a team device with the following syntax: **team=name:interfaces**. Replace *name* with the desired name of the team device and *interfaces* with a comma-separated list of physical (Ethernet) devices to be used as underlying interfaces in the team device. For example:

```
team=team0:enp0s1,enp0s2
```

**bridge=**

Use the **bridge=** option to configure a bridge device with the following syntax: **bridge=name:interfaces**. Replace *name* with the desired name of the bridge device and *interfaces* with a comma-separated list of physical (Ethernet) devices to be used as underlying interfaces in the bridge device. For example:

```
bridge=bridge0:enp0s1,enp0s2
```

**Additional resources**

- [Configuring and managing networking](#)

**F.3. CONSOLE BOOT OPTIONS**

This section describes how to configure 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**. When using the **console=** argument, the installation starts with a text UI. If you must use the **console=** option multiple times, the boot message is displayed on all specified console. However, the installation program uses only the last specified console. 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. To view the list of locales, enter the command **locale -a | grep \_** or the **localectl list-locales | grep \_** command.

**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:

- Disable geolocation: **inst.geoloc=0**
- Use the Fedora GeolP API: **inst.geoloc=provider\_fedora\_geoip**
- Use the Hostip.info GeolP API: **inst.geoloc=provider\_hostip**

If you do not specify the **inst.geoloc=** option, the default option is **provider\_fedora\_geoip**.

**inst.keymap=**

Use the **inst.keymap=** option to specify the keyboard layout 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. The graphical 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** you can use the **inst.noninteractive** option with a graphical or text installation. When you use the **inst.noninteractive** option 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 Virtual Network Computing (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. You can use this option by entering the command **vncviewer -listen**.

**inst.xdriver=**

Use the **inst.xdriver=** option to specify the name of the X driver 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 the **inst.xdriver=fbdev** option.

**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. 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 64-bit 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.

**F.4. DEBUG BOOT OPTIONS**

This section describes the options you can use when debugging issues.

### **inst.rescue**

Use the **inst.rescue** option to run the rescue environment for diagnosing and fixing systems. For example, you can [repair a filesystem in rescue mode](#).

### **inst.updates=**

Use the **inst.updates=** option to specify the location of the **updates.img** file that you want to apply during installation. The **updates.img** file can be derived from one of several sources.

**Table F.4. updates.img file sources**

| Source                            | Description                                                                                                                                                                                                                                      | Example                                                                                                                                                      |
|-----------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Updates from a network            | Specify the network location of <b>updates.img</b> . This does not require any modification to the installation tree. To use this method, edit the kernel command line to include <b>inst.updates</b> .                                          | <b>inst.updates=http://website.com/path/to/updates.img.</b>                                                                                                  |
| Updates from a disk image         | Save an <b>updates.img</b> on a floppy drive or a USB key. This can be done only with an <b>ext2</b> filesystem type of <b>updates.img</b> . To save the contents of the image on your floppy drive, insert the floppy disc and run the command. | <b>dd if=updates.img of=/dev/fd0 bs=72k count=20.</b> To use a USB key or flash media, replace <b>/dev/fd0</b> with the device name of your USB flash drive. |
| Updates from an installation tree | If you are using a CD, hard drive, HTTP, or FTP install, save the <b>updates.img</b> in the installation tree so that all installations can detect the <b>.img</b> file. The file name must be <b>updates.img</b> .                              | For NFS installs, save the file in the <b>images/</b> directory, or in the <b>RHupdates/</b> directory.                                                      |

### **inst.loglevel=**

Use the **inst.loglevel=** option to specify the minimum level of messages logged on a terminal. This option applies only to 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**
- **critical**



The default value is **info**, which means that by default, the logging terminal displays messages ranging from **info** to **critical**.

#### **inst.syslog=**

Sends log messages to the **syslog** process on the specified host when the installation starts. You can use **inst.syslog=** only if the remote **syslog** process is configured to accept incoming connections.

#### **inst.virtio-log=**

Use the **inst.virtio-log=** option to specify which virtio port (a character device at `/dev/virtio-ports/name`) to use for forwarding logs. The default value is **org.fedoraproject.anaconda.log.0**.

#### **inst.zram=**

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 using the hard drive. This setup allows the installation program to run with less available memory and improve installation speed. You can configure the **inst.zram=** option using the following values:

- **inst.zram=1** to enable zRAM swap, regardless of system memory size. By default, swap on zRAM is enabled on systems with 2 GiB or less RAM.
- **inst.zram=0** to disable zRAM swap, regardless of system memory size. By default, swap on zRAM is disabled on systems with more than 2 GiB of memory.

#### **rd.live-ram**

Copies the **stage 2** image in **images/install.img** into RAM. Note that this increases the memory required for installation by the size of the image which is usually between 400 and 800MB.

#### **inst.nokill**

Prevent the installation program from rebooting when a fatal error occurs, or at the end of the installation process. Use it capture installation logs which would be lost upon reboot.

#### **inst.noshell**

Prevent a shell on terminal session 2 (tty2) during installation.

#### **inst.notmux**

Prevent the use of tmux during installation. The output is generated without terminal control characters and is meant for non-interactive uses.

#### **inst.remotelog=**

Sends all 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.

## F.5. STORAGE BOOT OPTIONS

This section describes the options you can specify to customize booting from a storage device.

#### **inst.nodmraid**

Disables **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 such as, **dmraid** or **wipefs**.

**inst.nompath**

Disables support for multipath devices. Use this option only if your system has a false-positive that incorrectly identifies a normal block device as a multipath device.

**WARNING**

Use this option with caution. Do not use this option with multipath hardware. Using this option to install to a single path of a multipath device is not supported.

**inst.gpt**

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  $2^{32}$  sectors in size or larger. Disk sectors are typically 512 bytes in size, meaning that this is usually equivalent to 2 TiB. The **inst.gpt** boot option allows a GPT to be written to smaller disks.

## F.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**.

**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 F.5. Values for the ksdevice boot option**

| Value                       | Information                                                        |
|-----------------------------|--------------------------------------------------------------------|
| Not present                 | N/A                                                                |
| <b>ksdevice=link</b>        | Ignored as this option is the same as the default behavior         |
| <b>ksdevice=bootif</b>      | Ignored as this option is the default if <b>BOOTIF=</b> is present |
| <b>ksdevice=ibft</b>        | Replaced with <b>ip=ibft</b> . See <b>ip</b> for details           |
| <b>ksdevice=&lt;MAC&gt;</b> | Replaced with <b>BOOTIF=\${MAC}/:-}</b>                            |
| <b>ksdevice=&lt;DEV&gt;</b> | Replaced with <b>bootdev</b>                                       |

## F.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. Use **inst.repo** or specify the appropriate network options.

#### blacklist, nofirewire

The **modprobe** option now handles blocklisting 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.

#### repo=nfsiso

Use the **inst.repo=nfs:** option.

#### 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 because many options are 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 because 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 by using **ipv6.disable=1**. This setting is used by the installed system.

#### **upgradeany**

This option was no longer required because the installation program no longer handles upgrades.

## APPENDIX G. 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 version 6.11



#### NOTE

To receive the system updates, register your system with either of the management server.

## G.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.

### G.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](#).

#### Additional resources

- [Using and Configuring Red Hat Subscription Manager](#)

## G.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](#) .

### Additional resources

- [Using and Configuring Red Hat Subscription Manager](#)

## G.2. UNREGISTERING FROM SATELLITE SERVER

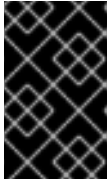
To unregister a Red Hat Enterprise Linux system from Satellite Server, remove the system from Satellite Server.

For more information, see [Removing a Host from Red Hat Satellite](#) in the *Managing Hosts* guide from Satellite Server documentation.

## APPENDIX H. ISCSI DISKS IN INSTALLATION PROGRAM

The Red Hat Enterprise Linux installer can discover and log in to iSCSI disks in two ways:

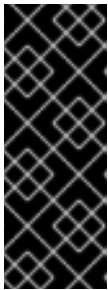
- When the installer starts, it checks if the BIOS or add-on boot ROMs of the system support iSCSI Boot Firmware Table (iBFT), a BIOS extension for systems that can boot from iSCSI. If the BIOS supports iBFT, the installer reads the iSCSI target information for the configured boot disk from the BIOS 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 6. BOOTING A BETA SYSTEM WITH UEFI SECURE BOOT

To enhance the security of your operating system, use the UEFI Secure Boot feature for signature verification when booting a Red Hat Enterprise Linux Beta release on systems having UEFI Secure Boot enabled.

### 6.1. UEFI SECURE BOOT AND RHEL BETA RELEASES

UEFI Secure Boot requires that the operating system kernel is signed with a recognized private key. UEFI Secure Boot then verifies the signature using the corresponding public key.

For Red Hat Enterprise Linux Beta releases, the kernel is signed with a Red Hat Beta-specific private key. UEFI Secure Boot attempts to verify the signature using the corresponding public key, but because the hardware does not recognize the Beta private key, Red Hat Enterprise Linux Beta release system fails to boot. Therefore, to use UEFI Secure Boot with a Beta release, add the Red Hat Beta public key to your system using the Machine Owner Key (MOK) facility.

### 6.2. ADDING A BETA PUBLIC KEY FOR UEFI SECURE BOOT

This section contains information about how to add a Red Hat Enterprise Linux Beta public key for UEFI Secure Boot.

#### Prerequisites

- The UEFI Secure Boot is disabled on the system.
- The Red Hat Enterprise Linux Beta release is installed, and Secure Boot is disabled even after system reboot.
- You are logged in to the system, and the tasks in the **Initial Setup** window are complete.

#### Procedure

1. Begin to enroll the Red Hat Beta public key in the system's Machine Owner Key (MOK) list:

```
# mokutil --import /usr/share/doc/kernel-keys/$(uname -r)/kernel-signing-ca.cer
```

**\$(uname -r)** is replaced by the kernel version - for example, **4.18.0-80.el8.x86\_64**.

2. Enter a password when prompted.
3. Reboot the system and press any key to continue the startup. The Shim UEFI key management utility starts during the system startup.
4. Select **Enroll MOK**.
5. Select **Continue**.
6. Select **Yes** and enter the password. The key is imported into the system's firmware.
7. Select **Reboot**.
8. Enable Secure Boot on the system.



## 6.3. REMOVING A BETA PUBLIC KEY

If you plan to remove the Red Hat Enterprise Linux Beta release, and install a Red Hat Enterprise Linux General Availability (GA) release, or a different operating system, then remove the Beta public key.

The procedure describes how to remove a Beta public key.

### Procedure

1. Begin to remove the Red Hat Beta public key from the system's Machine Owner Key (MOK) list:

```
# mokutil --reset
```

2. Enter a password when prompted.
3. Reboot the system and press any key to continue the startup. The Shim UEFI key management utility starts during the system startup.
4. Select **Reset MOK**.
5. Select **Continue**.
6. Select **Yes** and enter the password that you had specified in step 2. The key is removed from the system's firmware.
7. Select **Reboot**.

# CHAPTER 7. COMPOSING A CUSTOMIZED RHEL SYSTEM IMAGE

## 7.1. IMAGE BUILDER DESCRIPTION

To deploy a system on a cloud platform, create a system image. To create RHEL system images, use the image builder tool.

### 7.1.1. What is image builder?

You can use image builder to create customized system images of RHEL, including system images prepared for deployment on cloud platforms. Image builder automatically handles the setup details for each output type and is therefore easier to use and faster to work with than manual methods of image creation. You can access the image builder functionality through a command-line interface in the **composer-cli** tool, or a graphical user interface in the RHEL web console.



#### NOTE

From RHEL 8.3 onward, the **osbuild-composer** back end replaces **lorax-composer**. The new service provides REST APIs for image building.

### 7.1.2. Image builder terminology

#### Blueprint

A blueprint is a description of a customized system image. It lists the packages and customizations that will be part of the system. You can edit blueprints with customizations and save them as a particular version. When you create a system image from a blueprint, the image is associated with the blueprint in the image builder interface of the RHEL web console.

You can create blueprints in the TOML format.

#### Compose

Composes are individual builds of a system image, based on a specific 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 image that are not packages. This includes users, groups, and SSH keys.

### 7.1.3. Image builder output formats

Image builder can create images in multiple output formats shown in the following table. To check the supported types, run the command:

```
# composer-cli compose types
```

Table 7.1. Image builder output formats

| Description      | CLI name     | file extension |
|------------------|--------------|----------------|
| QEMU QCOW2 Image | <b>qcow2</b> | <b>.qcow2</b>  |

| Description                        | CLI name                         | file extension |
|------------------------------------|----------------------------------|----------------|
| TAR Archive                        | <b>tar</b>                       | <b>.tar</b>    |
| Amazon Machine Image Disk          | <b>ami</b>                       | <b>.raw</b>    |
| Azure Disk Image                   | <b>vhd</b>                       | <b>.vhd</b>    |
| Google Cloud Platform              | <b>gce</b>                       | <b>.vhd</b>    |
| VMware Virtual Machine Disk        | <b>vmdk</b>                      | <b>.vmdk</b>   |
| Openstack                          | <b>openstack</b>                 | <b>.qcow2</b>  |
| RHEL for Edge Commit               | <b>edge-commit</b>               | <b>.tar</b>    |
| RHEL for Edge Container            | <b>edge-container</b>            | <b>.tar</b>    |
| RHEL for Edge Installer            | <b>edge-installer</b>            | <b>.iso</b>    |
| RHEL for Edge Raw                  | <b>edge-raw-image</b>            | <b>.tar</b>    |
| RHEL for Edge Simplified Installer | <b>edge-simplified-installer</b> | <b>.iso</b>    |
| ISO image                          | <b>image-installer</b>           | <b>.iso</b>    |

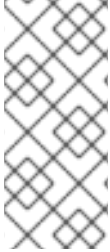
#### 7.1.4. Image builder system requirements

The environment where image builder runs, for example a dedicated virtual machine, must meet requirements listed in the following table.

**Table 7.2. Image builder system requirements**

| Parameter   | Minimal Required Value                                                                                                              |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------|
| System type | A dedicated virtual machine. Note that image builder is not supported on containers, including Red Hat Universal Base Images (UBI). |
| Processor   | 2 cores                                                                                                                             |
| Memory      | 4 GiB                                                                                                                               |
| Disk space  | 20 GiB of free space in the <b>/var</b> filesystem                                                                                  |

| Parameter         | Minimal Required Value     |
|-------------------|----------------------------|
| Access privileges | Administrator level (root) |
| Network           | Internet connectivity      |

**NOTE**

If you do not have internet connectivity, you can use image builder in isolated networks if you reconfigure it to not connect to Red Hat Content Delivery Network (CDN). For that, you must override the default repositories to point to your local repositories. Ensure that you have your content mirrored internally or use Red Hat Satellite. See [Managing repositories](#) for more details.

**Additional resources**

- [Provisioning to Satellite using a Red Hat image builder image](#)

**7.2. 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.

**7.2.1. Image builder system requirements**

The environment where image builder runs, for example a dedicated virtual machine, must meet requirements listed in the following table.

**Table 7.3. Image builder system requirements**

| Parameter         | Minimal Required Value                                                                                                              |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| System type       | A dedicated virtual machine. Note that image builder is not supported on containers, including Red Hat Universal Base Images (UBI). |
| Processor         | 2 cores                                                                                                                             |
| Memory            | 4 GiB                                                                                                                               |
| Disk space        | 20 GiB of free space in the <b>/var</b> filesystem                                                                                  |
| Access privileges | Administrator level (root)                                                                                                          |
| Network           | Internet connectivity                                                                                                               |



## NOTE

If you do not have internet connectivity, you can use image builder in isolated networks if you reconfigure it to not connect to Red Hat Content Delivery Network (CDN). For that, you must override the default repositories to point to your local repositories. Ensure that you have your content mirrored internally or use Red Hat Satellite. See [Managing repositories](#) for more details.

### Additional resources

- [Provisioning to Satellite using a Red Hat image builder image](#)

## 7.2.2. Installing image builder in a virtual machine

To install image builder on a dedicated virtual machine (VM), follow these steps:

### Prerequisites

- You must be connected to a RHEL VM.
- The VM for image builder must be running and subscribed to Red Hat Subscription Manager (RHSM) or Red Hat Satellite.

### Procedure

1. Install the image builder and other necessary packages on the VM:

- **osbuild-composer** - supported from RHEL 8.3 onward
- **composer-cli**
- **cockpit-composer**
- **bash-completion**

```
# yum install osbuild-composer composer-cli cockpit-composer bash-completion
```

The web console is installed as a dependency of the *cockpit-composer* package.

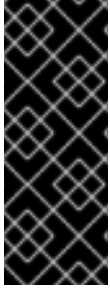
2. Enable image builder to start after each reboot:

```
# systemctl enable --now osbuild-composer.socket
# systemctl enable --now cockpit.socket
```

The **osbuild-composer** and **cockpit** services start automatically on first access.

3. 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
```



## IMPORTANT

The **osbuild-composer** package is the new backend engine that will be the preferred default and focus of all new functionality beginning with Red Hat Enterprise Linux 8.3 and later. The previous backend **lorax-composer** package is considered deprecated, will only receive select fixes for the remainder of the Red Hat Enterprise Linux 8 life cycle and will be omitted from future major releases. It is recommended to uninstall **lorax-composer** in favor of **osbuild-composer**.

## Verification

You can use a system journal to track image builder service activities. Additionally, you can find the log messages in the file.

- To find the journal output for traceback, run the following commands:

```
$ journalctl | grep osbuild
```

- To show both remote or local workers:

```
$ journalctl -u osbuild-worker*
```

- To show the running services:

```
$ journalctl -u osbuild-composer.service
```

### 7.2.3. Reverting to lorax-composer image builder backend

The **osbuild-composer** backend, though much more extensible, does not currently achieve feature parity with the previous **lorax-composer** backend.

To revert to the previous backend, follow the steps:

#### Prerequisites

- You have installed the **osbuild-composer** package

#### Procedure

1. Remove the **osbuild-composer** backend.

```
# yum remove osbuild-composer  
# yum remove weldr-client
```

2. In the **/etc/yum.conf** file, add an exclude entry for **osbuild-composer** package.

```
# cat /etc/yum.conf  
[main]  
gpgcheck=1  
installonly_limit=3  
clean_requirements_on_remove=True
```

```
best=True
skip_if_unavailable=False
exclude=osbuild-composer weldr-client
```

3. Install the **lorax-composer** package.

```
# yum install lorax-composer composer-cli
```

4. Enable and start the **lorax-composer** service to start after each reboot.

```
# systemctl enable --now lorax-composer.socket
# systemctl start lorax-composer
```

### Additional resources

- [Create a Case at Red Hat Support](#) .

## 7.3. CREATING SYSTEM IMAGES USING THE 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, you can use the command-line interface (CLI) or the web console interface. Currently, however, the CLI is the preferred method to use image builder.

### 7.3.1. Introducing the image builder command-line interface

The image builder command-line interface (CLI) is currently the preferred method to use image builder. It offers more functionality than the web console interface. To use the CLI, run the **composer-cli** command with the 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** commands as non-root, the user must be in the **weldr** or **root** groups.

- To add a user to the **weldr** or **root** groups, run the following commands:

```
$ sudo usermod -a -G weldr user
$ newgrp weldr
```

### 7.3.2. Creating an image builder blueprint using the command-line interface

You can create a new image builder blueprint using the command-line interface (CLI). The blueprint describes the final image and its customizations, such as packages, and kernel customizations.

## Prerequisite

- Access to the image builder tool.

## 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 the name of the package, such as **httpd**, **gdb-doc**, or **coreutils**.

Replace *package-version* with the version to use. This field supports **dnf** version specifications:

- For a specific version, use the exact version number such as **8.7.0**.
  - For the latest available version, use the asterisk **\***.
  - For the latest minor version, use formats such as **8.\***.
3. Customize your blueprints to suit your needs. For example, disable Simultaneous Multi Threading (SMT), add the following lines to the blueprint file:

```
[customizations.kernel]
append = "nosmt=force"
```

For additional customizations available, see [Supported Image Customizations](#).

4. Save the file, for example, 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.





## NOTE

To create images using **composer-cli** as non-root, add your user to the **weldr** or **root** groups.

```
# usermod -a -G weldr user
$ newgrp weldr
```

## Verification

- List the existing blueprints to verify that the blueprint has been pushed and exists:

```
# composer-cli blueprints list
```

- Display the blueprint configuration you have just added:

```
# composer-cli blueprints show BLUEPRINT-NAME
```

- Check whether the components and versions listed in the blueprint and their dependencies are valid:

```
# composer-cli blueprints depsolve BLUEPRINT-NAME
```

If image builder is unable to depsolve a package from your custom repositories, follow the steps:

- Remove the osbuild-composer cache:

```
$ sudo rm -rf /var/cache/osbuild-composer/*
$ sudo systemctl restart osbuild-composer
```

## Additional resources

- [osbuild-composer is unable to depsolve a package from my custom repository](#) .
- [Composing a customized RHEL system image with proxy server](#) .

### 7.3.3. Editing an image builder blueprint with command-line interface

You can edit an existing image builder blueprint in the command-line (CLI) interface to, for example, add a new package, or define a new group, and to create your customized images. For that, follow the steps:

#### Prerequisites

- You have created a blueprint.

#### Procedure

1. Save (export) the blueprint to a local text file:

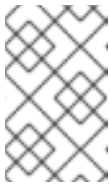
```
# composer-cli blueprints save BLUEPRINT-NAME
```

2. Edit the `BLUEPRINT-NAME.toml` file with a text editor and make your changes.
3. Before finishing the edits, verify that the file is a valid blueprint:

- a. Remove this line, if present:

```
packages = []
```

- b. Increase the version number, for example, from 0.0.1 to 0.1.0. Remember that image builder blueprint versions must use the [Semantic Versioning](#) scheme. Note also that if you do not change the version, the **patch** version component increases 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 text editor.
5. Push (import) the blueprint back into image builder:

```
# composer-cli blueprints push BLUEPRINT-NAME.toml
```



#### NOTE

To import the blueprint back into image builder, supply the file name including the **.toml** extension, while in other commands use only the blueprint name.

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
```

#### Additional resources

- [Supported Image Customizations](#).

### 7.3.4. Creating a system image with image builder in the command-line interface

You can build a custom image using the image builder command-line interface.

#### Prerequisites

- You have a blueprint prepared for the image. See [Creating an image builder blueprint using the command-line interface](#).

## 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 the image. For the available values, see the output of the **composer-cli compose types** command.

The compose process starts in the background and shows the composer Universally Unique Identifier (UUID).

2. Wait until the compose process is finished. The image creation can take up to ten minutes to complete.

To check the status of the compose:

```
# composer-cli compose status
```

A finished compose shows the **FINISHED** status value. To identify your compose in the list, use its UUID.

3. After the compose process is finished, download the resulting image file:

```
# composer-cli compose image UUID
```

Replace *UUID* with the UUID value shown in the previous steps.

## Verification

After you create your image, you can check the image creation progress using the following commands:

- Check the compose status:

```
$ sudo composer-cli compose status
```

- Download the metadata of the image:

```
$ sudo composer-cli compose metadata UUID
```

- Download the logs of the image:

```
$ sudo composer-cli compose logs UUID
```

The command creates a **.tar** file that contains the logs for the image creation. If the logs are empty, you can check the journal.

- Check the journal:

```
$ journalctl | grep osbuild
```

- Check the manifest:

```
$ sudo cat /var/lib/osbuild-composer/jobs/job_UUID.json
```

You can find the `job_UUID.json` in the journal.

### Additional resources

- [Tracing image builder](#)

## 7.3.5. 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

#### List the available image types

```
# composer-cli compose types
```

#### Start a compose

```
# composer-cli compose start BLUEPRINT COMPOSE-TYPE
```

Replace `BLUEPRINT` with the 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
```

See more subcommands and options

```
# composer-cli help
```

Additional resources

- *composer-cli*(1) man page

### 7.3.6. Image builder blueprint format

Image builder blueprints are presented to the user as plain text in the TOML format.

The elements of a typical blueprint file include the following:

**The blueprint metadata**

```
name = "BLUEPRINT-NAME"
description = "LONG FORM DESCRIPTION TEXT"
version = "VERSION"
```

The *BLUEPRINT-NAME* and *LONG FORM DESCRIPTION TEXT* field are a name and description for your blueprint.

The *VERSION* is a version number according to the [Semantic Versioning](#) scheme.

This part is present only once for the entire blueprint file.

The *modules* entry lists the package names and versions of packages to be installed into the image.

The *group* entry describes a group of packages to be installed into the image. Groups use the following package categories:

- Mandatory
  - Default
  - Optional
- Blueprints install the mandatory and default packages. There is no mechanism for selecting optional packages.

### Groups to include in the image

```
[[groups]]
name = "group-name"
```

The *group-name* is the name of the group, for example, **anaconda-tools**, **widget**, **wheel** or **users**.

### Packages to include in the image

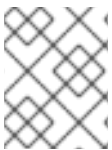
```
[[packages]]
name = "package-name"
version = "package-version"
```

*package-name* is the name of the package, such as **httpd**, **gdb-doc**, or **coreutils**.

*package-version* is a version to use. This field supports **dnf** version specifications:

- For a specific version, use the exact version number such as **8.7.0**.
- For latest available version, use the asterisk **\***.
- For a latest minor version, use a format such as **8.\***.

Repeat this block for every package to include.

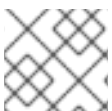


#### NOTE

Currently there are no differences between packages and modules in the image builder tool. Both are treated as RPM package dependencies.

### 7.3.7. Supported image customizations

You can customize your image by adding to your blueprint an additional RPM package, by enabling a service, or by customizing a kernel command line parameter. You can use several image customizations within blueprints. To make use of these options, you must configure the customizations in the blueprint and import (push) it to image builder.



#### NOTE

These customizations are not supported when using image builder in the web console.

### Select a package group

```
[[packages]]
name = "package_group_name"
```

Replace `"package_group_name"` with the name of the group. For example, `"@server with 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
```

The GID is optional and must already exist in the image. Optionally, a package creates it, or the blueprint creates the GID by using the `[[customizations.group]]` entry.



#### IMPORTANT

To generate the **password hash**, you must install **python3** on your system.

```
# yum install python3
```

Replace `PASSWORD-HASH` with the actual **password hash**. To generate the **password 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.

You must enter the **name**. You can omit any of the lines that you do not need.

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

The "Set an existing users SSH key" customization is only applicable for existing users. To create a user and set an SSH key, see the **User specifications for the resulting system image** customization.

### Append a kernel boot parameter option to the defaults

```
[customizations.kernel]  
append = "KERNEL-OPTION"
```

By default, image builder builds a default kernel into the image. But, you can customize the kernel with the following configuration in blueprint

```
[customizations.kernel]  
name = "KERNEL-rt"
```

### Define a kernel name to use in an image

```
[customizations.kernel.name]  
name = "KERNEL-NAME"
```

### 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]  
languages = ["LANGUAGE"]  
keyboard = "KEYBOARD"
```

Setting both the language and the keyboard options is mandatory. You can add many other languages. The first language you add will be the primary language and the other languages will be secondary. For example:



```
[customizations.locale]
languages = ["en_US.UTF-8"]
keyboard = "us"
```

To list the values supported by the languages, run the following command:

```
$ localectl list-locales
```

To list the values supported by the keyboard, run the following command:

```
$ localectl list-keymaps
```

### Set the firewall for the resulting system image

```
[customizations.firewall]
port = ["PORTS"]
```

To enable lists, you can use numeric ports, or their names from the `/etc/services` file.

### Customize the firewall services

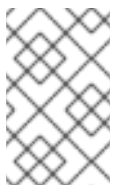
Review the available firewall services.

```
$ firewall-cmd --get-services
```

In the blueprint, under section **customizations.firewall.service**, specify the firewall services that you want to customize.

```
[customizations.firewall.services]
enabled = ["SERVICES"]
disabled = ["SERVICES"]
```

The services listed in **firewall.services** are different from the service-names available in the `/etc/services` file.



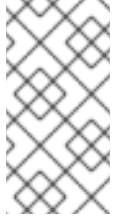
#### NOTE

If you do not want to customize the firewall services, omit the **[customizations.firewall]** and **[customizations.firewall.services]** sections from the blueprint.

### 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 to ensure that the image works correctly and this setup cannot be overridden. The **[customizations.services]** customization in the blueprint do not replace these services, but add them to the list of services already present in the image templates.



## NOTE

Each time a build starts, it clones the repository of the host system. If you refer to a repository with a large amount of history, it might take some time to clone and it uses a significant amount of disk space. Also, the clone is temporary and the build removes it after it creates the RPM package.

## Specify a custom filesystem configuration

You can specify a custom filesystem configuration in your blueprints and therefore create images with a specific disk layout, instead of the default layout configuration. By using the non-default layout configuration in your blueprints, you can benefit from:

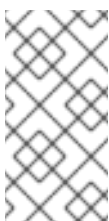
- security benchmark compliance
- protection against out-of-disk errors
- improved performance
- consistency with existing setups

To customize the filesystem configuration in your blueprint:

```
[[customizations.filesystem]]
mountpoint = "MOUNTPOINT"
size = MINIMUM-PARTITION-SIZE
```

The blueprint supports the following **mountpoints** and their sub-directories:

- **/** - the root mount point
- **/var**
- **/home**
- **/opt**
- **/srv**
- **/usr**
- **/app**
- **/data**
- **/boot** - Supported from RHEL 8.7 and RHEL 9.1 onward.



## NOTE

Customizing mount points is only supported from RHEL 8.5 and RHEL 9.0 distributions onward, by using the CLI. In earlier distributions, you can only specify the **root** partition as a mount point and specify the **size** argument as an alias for the image size.

If you have more than one partition in the customized image, you can create images with a customized file system partition on LVM and resize those partitions at runtime. To do this, you can specify a customized filesystem configuration in your blueprint and

therefore create images with the desired disk layout. The default filesystem layout remains unchanged - if you use plain images without file system customization, and **cloud-init** resizes the root partition.



#### NOTE

From 8.6 onward, for the **osbuild-composer-46.1-1.el8** RPM and later version, the physical partitions are no longer available and filesystem customizations create logical volumes.

The blueprint automatically converts the file system customization to a LVM partition.

The **MINIMUM-PARTITION-SIZE** value has no default size format. The blueprint customization supports the following values and units: kB to TB and KiB to TiB. For example, you can define the mount point size in bytes:

```
[[customizations.filesystem]]
mountpoint = "/var"
size = 1073741824
```

You can also define the mount point size by using units.



#### NOTE

You can only define the mount point size by using units for the package version provided for RHEL 8.6 and RHEL 9.0 distributions onward.

For example:

```
[[customizations.filesystem]]
mountpoint = "/opt"
size = "20 GiB"

or

[[customizations.filesystem]]
mountpoint = "/boot"
size = "1 GiB"
```

#### Additional resources

- [Blueprint import fails after adding filesystem customization "size"](#) .

### 7.3.8. Packages installed by image builder

When you create a system image using image builder, the system installs a set of base packages. By default, image builder uses the **Core** group as the base list of packages.

**Table 7.4. Default packages to support image type creation**

| Image type  | Default Packages                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ami         | checkpolicy, chrony, cloud-init, cloud-utils-growpart, @Core, dhcp-client, gdisk, insights-client, kernel, langpacks-en, net-tools, NetworkManager, redhat-release, redhat-release-eula, rng-tools, rsync, selinux-policy-targeted, tar, yum-utils                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| openstack   | @core, langpacks-en                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| qcow2       | @core, chrony, dnf, kernel, yum, nfs-utils, dnf-utils, cloud-init, python3-jjsonschema, qemu-guest-agent, cloud-utils-growpart, dracut-norescue, tar, tcpdump, rsync, dnf-plugin-spacewalk, rhn-client-tools, rhnlib, rhnsd, rhn-setup, NetworkManager, dhcp-client, cockpit-ws, cockpit-system, subscription-manager-cockpit, redhat-release, redhat-release-eula, rng-tools, insights-client                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| tar         | policycoreutils, selinux-policy-targeted                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| vhd         | @core, langpacks-en                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| vmdk        | @core, chrony, cloud-init, firewalld, langpacks-en, open-vm-tools, selinux-policy-targeted                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| edge-commit | attr, audit, basesystem, bash, bash-completion, chrony, clevis, clevis-dracut, clevis-luks, container-selinux, coreutils, criu, cryptsetup, curl, dnsmasq, dosfstools, dracut-config-generic, dracut-network, e2fsprogs, firewalld, fuse-overlayfs, fwupd, glibc, glibc-minimal-langpack, gnupg2, greenboot, gzip, hostname, ima-evm-utils, iproute, iptables, iputils, keyutils, less, lvm2, NetworkManager, NetworkManager-wifi, NetworkManager-wwan, nss-altfiles, openssh-clients, openssh-server, passwd, pinentry, platform-python, podman, policycoreutils, policycoreutils-python-utils, polkit, procps-ng, redhat-release, rootfiles, rpm, rpm-ostree, rsync, selinux-policy-targeted, setools-console, setup, shadow-utils, shadow-utils, skopeo, slirp4netns, sudo, systemd, tar, tmux, traceroute, usbguard, util-linux, vim-minimal, wpa_supplicant, xz |

| Image type            | Default Packages                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>edge-container</b> | dnf, dosfstools, e2fsprogs, glibc, lorax-templates-generic, lorax-templates-rhel, lvm2, policycoreutils, python36, python3-iniparse, qemu-img, selinux-policy-targeted, systemd, tar, xfsprogs, xz                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>edge-installer</b> | aajohan-comfortaa-fonts, abattis-cantarell-fonts, alsa-firmware, alsa-tools-firmware, anaconda, anaconda-install-env-deps, anaconda-widgets, audit, bind-utils, bitmap-fangsongti-fonts, bzip2, cryptsetup, dbus-x11, dejavu-sans-fonts, dejavu-sans-mono-fonts, device-mapper-persistent-data, dnf, dump, ethtool, fcoe-utils, ftp, gdb-gdbserver, gdisk, gfs2-utils, glibc-all-langpacks, google-noto-sans-cjk-ttc-fonts, gsettings-desktop-schemas, hdparm, hexedit, initscripts, ipmitool, iwl3945-firmware, iwl4965-firmware, iwl6000g2a-firmware, iwl6000g2b-firmware, jomolhari-fonts, kacst-farsi-fonts, kacst-qurn-fonts, kbd, kbd-misc, kdump-anaconda-addon, khmeros-base-fonts, libblockdev-lvm-dbus, libertas-sd8686-firmware, libertas-sd8787-firmware, libertas-usb8388-firmware, libertas-usb8388-olpc-firmware, libibverbs, libreport-plugin-bugzilla, libreport-plugin-reportuploader, libreport-rhel-anaconda-bugzilla, librsvg2, linux-firmware, lklug-fonts, lldpad, lohit-assamese-fonts, lohit-bengali-fonts, lohit-devanagari-fonts, lohit-gujarati-fonts, lohit-gurmukhi-fonts, lohit-kannada-fonts, lohit-odia-fonts, lohit-tamil-fonts, lohit-telugu-fonts, lsof, madan-fonts, metacity, mtr, mt-st, net-tools, nmap-ncat, nm-connection-editor, nss-tools, openssh-server, oscap-anaconda-addon, pciutils, perl-interpreter, pigz, python3-pyatspi, rdma-core, redhat-release-eula, rpm-ostree, rsync, rsyslog, sg3_utils, sil-abyssinica-fonts, sil-padauk-fonts, sil-scheherazade-fonts, smartmontools, smc-meera-fonts, spice-vdagent, strace, system-storage-manager, thai-scalable-waree-fonts, tigervnc-server-minimal, tigervnc-server-module, udisks2, udisks2-iscsi, usbutils, vim-minimal, volume_key, wget, xfsdump, xorg-x11-drivers,xorg-x11-fonts-misc,xorg-x11-server-utils,xorg-x11-server-Xorg, xorg-x11-xauth |

| Image type                       | Default Packages                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>edge-simplified-installer</b> | <b>attr, basesystem, binutils, bsdtar, clevis-dracut, clevis-luks, cloud-utils-growpart, coreos-installer, coreos-installer-dracut, coreutils, device-mapper-multipath, dnsmasq, dosfstools, dracut-live, e2fsprogs, fcoe-utils, fdo-init, gzip, ima-evm-utils, iproute, iptables, iputils, iscsi-initiator-utils, keyutils, lldpad, lvm2, passwd, policycoreutils, policycoreutils-python-utils, procps-ng, rootfiles, setools-console, sudo, traceroute, util-linux</b> |
| <b>image-installer</b>           | <b>anaconda-dracut, curl, dracut-config-generic, dracut-network, hostname, iwl100-firmware, iwl1000-firmware, iwl105-firmware, iwl135-firmware, iwl2000-firmware, iwl2030-firmware, iwl3160-firmware, iwl5000-firmware, iwl5150-firmware, iwl6000-firmware, iwl6050-firmware, iwl7260-firmware, kernel, less, nfs-utils, openssh-clients, ostree, plymouth, prefixdevname, rng-tools, rpcbind, selinux-policy-targeted, systemd, tar, xfsprogs, xz</b>                    |
| <b>edge-raw-image</b>            | <b>dnf, dosfstools, e2fsprogs, glibc, lorax-templates-generic, lorax-templates-rhel, lvm2, policycoreutils, python36, python3-iniparse, qemu-img, selinux-policy-targeted, systemd, tar, xfsprogs, xz</b>                                                                                                                                                                                                                                                                 |
| <b>gce</b>                       | <b>@core, langpacks-en, acpid, dhcp-client, dnf-automatic, net-tools, python3, rng-tools, tar, vim</b>                                                                                                                                                                                                                                                                                                                                                                    |



## NOTE

When you add additional components to your blueprint, ensure that the packages in the components you added do not conflict with any other package components. Otherwise, the system fails to solve dependencies and creating your customized image fails. You can check if there is no conflict between the packages by running the command:

```
# composer-cli blueprints depsolve BLUEPRINT-NAME
```

## Additional resources

- [Image builder description](#)

## 7.3.9. Enabled services on custom images

When you use image builder to configure a custom image, the default services that the image uses are determined by the following:

- The RHEL release on which you use the **osbuild-composer** utility
- The image type

For example, the **ami** image type enables the **sshd**, **chronyd**, and **cloud-init** services by default. If these services are not enabled, the custom image does not boot.

**Table 7.5. Enabled services to support image type creation**

| Image type              | Default enabled Services                                                        |
|-------------------------|---------------------------------------------------------------------------------|
| <b>ami</b>              | sshd, cloud-init, cloud-init-local, cloud-config, cloud-final                   |
| <b>openstack</b>        | sshd, cloud-init, cloud-init-local, cloud-config, cloud-final                   |
| <b>qcow2</b>            | cloud-init                                                                      |
| <b>rhel-edge-commit</b> | No extra service enables by default                                             |
| <b>tar</b>              | No extra service enables by default                                             |
| <b>vhd</b>              | sshd, chronyd, waagent, cloud-init, cloud-init-local, cloud-config, cloud-final |
| <b>vmdk</b>             | sshd, chronyd, vmttoolsd, cloud-init                                            |

Note: You can customize which services to enable during the system boot. However, the customization does not override services enabled by default for the mentioned image types.

#### Additional resources

- [Supported Image Customizations](#)

## 7.4. CREATING SYSTEM IMAGES USING THE 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.

### 7.4.1. Accessing the image builder GUI in the RHEL web console

With the **cockpit-composer** plugin for the RHEL web console, you can manage image builder blueprints and composes using a graphical interface. The preferred method for controlling image builder is the command-line interface.

### Prerequisites

- You must have root access to the system.
- You installed image builder.

### Procedure

1. Open <https://localhost:9090/> in a web browser on the system where image builder is installed. For more information about how to remotely access image builder, see [Managing systems using the RHEL web console](#) document.
2. Log in to the web console as the root user.
3. To display the image builder controls, click the **image builder** icon, in the upper-left corner of the window.  
The image builder view opens, listing existing blueprints.

## 7.4.2. Creating an image builder blueprint in the web console interface

Creating a blueprint is a necessary step before describing the customized system image.

### Prerequisites

- You have opened the image builder app from web console in a browser. See [Accessing the image builder GUI in the RHEL web console](#).

### Procedure

1. Click **Create Blueprint** in the upper-right corner.  
A dialog wizard with fields for the blueprint name and description opens.
2. Enter the name of the blueprint and, optionally, its description.
3. Click **Create**.

The image builder view opens, listing existing blueprints.

## 7.4.3. Creating a system image using image builder in the web console interface

You can create a system image from a blueprint by completing the following steps:

### Prerequisites

- You opened the image builder app from web console in a browser.
- You created a blueprint.

### Procedure

1. Click **Back to blueprints** to show the blueprints table.
2. On the blueprint table, find the blueprint you want to build an image.
  - a. Optionally, you can find the blueprint using the search box. Enter the blueprint name.



3. On the right side of the chosen blueprint, click **Create Image**. The **Create image** dialog wizard opens.
4. On the **Image output** page, complete the following steps:
  - a. From the **Image output type** list, select the image type you want.
    - i. You can upload some images to their target cloud environment, such as **Amazon Web Service** and **Oracle Cloud Infrastructure**. For that, check the **Upload to Target cloud** box.
    - ii. You are prompted to add credentials for the cloud environment on the next page.
5. From the **Image Size** field, enter the image size. The minimum size depends on the image type. Click **Next**.
6. On the **Upload to Targeted\_Cloud** page, complete the following steps:

NOTE: This page is not visible if you did not check the box to upload your image to the cloud environment.

  - a. On the **Authentication** page, enter the information related to your target cloud account ID and click **Next**.
  - b. On the **Destination** page, enter the information related to your target cloud account type and click **Next**.
7. On the **Customizations** page, complete the following steps:
  - a. On the **System** page, enter the Hostname. If you do not enter a hostname, the operating system determines a hostname for your system.
  - b. On the **Users** page, click **Add user**:
    - i. Mandatory: Enter a Username.
    - ii. Enter a password.
    - iii. Enter an SSH key.
    - iv. Check the box if you want to make the user a Server administrator. Click **Next**.
8. On the **Package** page, complete the following steps:
  - a. On the **Available packages** search field, enter the package name you want to add to your system image.

**NOTE**

Searching for the package can take some time to complete.
  - b. Click the > arrow to add the selected package or packages. Click **Next**.
9. On the **Review** page, review the details about the image creation. Click **Save blueprint** to save the customizations you added to your blueprint. Click **Create image**. The image build starts and takes up to 20 minutes to complete.

## 7.5. PREPARING AND UPLOADING CLOUD IMAGES USING IMAGE BUILDER

Image builder can create custom system images ready for use on various cloud platforms. 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. You can push customized image clouds through the **image builder** application in the RHEL web console, available for a subset of the service providers that we support, such as **AWS** and **Microsoft Azure** clouds. See [Pushing images to AWS Cloud AMI](#) and [Pushing VHD images to Microsoft Azure cloud](#).

### 7.5.1. Preparing to upload AWS AMI images

Before uploading an AWS AMI image, you must configure a system for uploading the 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. Run the following command to set your profile. The terminal prompts you to provide your credentials, region and output format:

```
$ aws configure
AWS Access Key ID [None]:
AWS Secret Access Key [None]:
Default region name [None]:
Default output format [None]:
```

4. Define a name for your bucket and use the following command to create a bucket:

```
$ BUCKET=bucketname
$ aws s3 mb s3://$BUCKET
```

Replace *bucketname* with the actual bucket name. It must be a globally unique name. As a result, your bucket is created.

5. To grant permission to access the S3 bucket, create a **vmimport** S3 Role in the AWS Identity and Access Management (IAM), 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
```

## Additional resources

- [Using high-level \(s3\) commands with the AWS CLI](#)

## 7.5.2. Uploading an AMI image to AWS using the CLI

You can use image builder to build **ami** images and push them directly to Amazon AWS Cloud service provider using the CLI.

### Prerequisites

- You have an **Access Key ID** configured in the [AWS IAM](#) account manager.
- You have a writable [S3 bucket](#) prepared.
- You have a defined blueprint.

### Procedure

1. Using the text editor, create a configuration file with the following content:

```
provider = "aws"

[settings]
accessKeyID = "AWS_ACCESS_KEY_ID"
secretAccessKey = "AWS_SECRET_ACCESS_KEY"
bucket = "AWS_BUCKET"
region = "AWS_REGION"
key = "IMAGE_KEY"
```

Replace values in the fields with your credentials for **accessKeyID**, **secretAccessKey**, **bucket**, and **region**. The **IMAGE\_KEY** value is the name of your VM Image to be uploaded to EC2.

2. Save the file as *CONFIGURATION-FILE.toml* and close the text editor.
3. Start the compose:

```
# composer-cli compose start BLUEPRINT-NAME IMAGE-TYPE IMAGE_KEY
CONFIGURATION-FILE.toml
```

Replace:

- *BLUEPRINT-NAME* with the name of the blueprint you created

- *IMAGE-TYPE* with the **ami** image type.
- *IMAGE\_KEY* with the name of your VM Image to be uploaded to EC2.
- *CONFIGURATION-FILE.toml* with the name of the configuration file of the cloud provider.



#### NOTE

You must have the correct IAM settings for the bucket you are going to send your customized image to. You have to set up a policy to your bucket before you are able to upload images to it.

4. Check the status of the image build and upload it to AWS:

```
# composer-cli compose status
```

After the image upload process is complete, you can see the "FINISHED" status.

### Verification

To confirm that the image upload was successful:

1. Access [EC2](#) on the menu and select the correct region in the AWS console. The image must have the **available** status, to indicate that it was successfully uploaded.
2. On the dashboard, select your image and click **Launch**.

### Additional Resources

- [Required service role to import a VM](#)

### 7.5.3. Pushing images to AWS Cloud AMI

You can push the output image that you create directly to the **Amazon AWS Cloud AMI** service provider.

#### Prerequisites

- You must have **root** or **wheel** group user access to the system.
- You have opened the image builder interface of the RHEL web console in a browser.
- You have create a blueprint. See [Creating an image builder blueprint in the web console interface](#).
- You must have an Access Key ID configured in the [AWS IAM](#) account manager.
- You must have a writable [S3 bucket](#) prepared.

#### Procedure

1. Click the **blueprint name**.
2. Select the tab **Images**.

3. Click **Create Image** to create your customized image.  
A pop-up window opens.
  - a. From the **Type** drop-down menu list, select **Amazon Machine Image Disk (.raw)**.
  - b. Check the **Upload to AWS** check box to upload your image to the AWS Cloud and click **Next**.
  - c. To authenticate your access to AWS, type your **AWS access key ID** and **AWS secret access key** in the corresponding fields. Click **Next**.

**NOTE**

You can view your AWS secret access key only when you create a new Access Key ID. If you do not know your Secret Key, generate a new Access Key ID.

- d. Type the name of the image in the **Image name** field, type the Amazon bucket name in the **Amazon S3 bucket name** field and type the **AWS region** field for the bucket you are going to add your customized image to. Click **Next**.
- e. Review the information and click **Finish**.  
Optionally, you can click **Back** to modify any incorrect detail.

**NOTE**

You must have the correct IAM settings for the bucket you are going to send your customized image. This procedure uses the IAM Import and Export, so you have to set up a **policy** to your bucket before you are able to upload images to it. For more information, see [Required Permissions for IAM Users](#).

4. A small pop-up on the upper right informs you of the saving progress. It also informs that the image creation has been initiated, the progress of this image creation and the subsequent upload to the AWS Cloud.  
After the process is complete, you can see the **Image build complete** status.
5. Click [Service→EC2](#) on the menu and choose the [correct region](#) in the AWS console. The image must have the **Available** status, to indicate that it is uploaded.
6. On the dashboard, select your image and click **Launch**.
7. A new window opens. Choose an instance type according to the resources you need to start your image. Click **Review and Launch**.
8. Review your instance start details. You can edit each section if you need to make any changes. Click **Launch**.
9. Before you start the instance, select a public key to access it.  
You can either use the key pair you already have or you can create a new key pair. Alternatively, you can use **image builder** to add a user to the image with a preset public key. See [Creating a user account with an SSH key](#) for more details.

Follow the next steps to create a new key pair in EC2 and attach it to the new instance.

- a. From the drop-down menu list, select **Create a new key pair**.

- b. Enter the name to the new key pair. It generates a new key pair.
  - c. Click **Download Key Pair** to save the new key pair on your local system.
10. Then, you can click **Launch Instance** to start your instance.  
You can check the status of the instance, which displays as **Initializing**.
11. After the instance status is **running**, the **Connect** button becomes available.
12. Click **Connect**. A pop-up window appears with instructions on how to connect using SSH.
  - a. Select **A standalone SSH client** as the preferred connection method to and open a terminal.
  - b. In the location you store your private key, ensure that your key is publicly viewable for SSH to work. To do so, run the command:  

```
$ chmod 400 <your-instance-name.pem>_
```
  - c. Connect to your instance using its Public DNS:  

```
$ ssh -i "<_your-instance-name.pem_"> ec2-user@<_your-instance-IP-address_>
```
  - d. Type **yes** to confirm that you want to continue connecting.  
As a result, you are connected to your instance using SSH.

### Verification

1. Check if you are able to perform any action while connected to your instance using SSH.

### Additional resources

- [Open a case on Red Hat Customer Portal](#)
- [Connecting to your Linux instance using SSH](#)
- [Open support cases](#)

## 7.5.4. Preparing to upload Microsoft Azure VHD images

You can use image builder to prepare a VHD image that can be uploaded to **Microsoft Azure** cloud.

### Prerequisites

- You must have a usable Microsoft Azure resource group and storage account.
- You have python2 installed because the **AZ CLI** tool depends specifically on python 2.7.

### Procedure

1. Import the Microsoft repository key:

```
# rpm --import https://packages.microsoft.com/keys/microsoft.asc
```

2. Create a local **azure-cli** repository information:

```
# sh -c 'echo -e "[azure-cli]\nname=Azure
CLI\nbaseurl=https://packages.microsoft.com/yumrepos/azure-
cli\nenabled=1\nngpgcheck=1\nngpgkey=https://packages.microsoft.com/keys/microsoft.asc" >
/etc/yum.repos.d/azure-cli.repo'
```

3. Install the Microsoft Azure CLI:

```
# yumdownloader azure-cli
# rpm -ivh --nodeps azure-cli-2.0.64-1.el7.x86_64.rpm
```



#### NOTE

The downloaded version of the Microsoft Azure CLI package may vary depending on the current available version.

4. Run the Microsoft Azure CLI:

```
$ az login
```

The terminal shows the following message **Note, we have launched a browser for you to login. For old experience with device code, use "az login --use-device-code**. Then, the terminal opens a browser with a link to <https://microsoft.com/devicelogin> from where you can login.



#### NOTE

If you are running a remote (SSH) session, the <https://microsoft.com/devicelogin> link will not open in the browser. In this case, you can copy the link to a browser and login to authenticate your remote session. To sign in, use a web browser to open the page <https://microsoft.com/devicelogin> and enter the device code to authenticate.

5. List the keys for the storage account in Microsoft 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 your Microsoft Azure resource group and *storage-account-name* with name of your Microsoft Azure storage account.



#### NOTE

You can list the available resources using the following command:

```
$ az resource list
```

6. Make note of value **key1** in the output of the previous command, and assign it to an environment variable:



```
$ KEY1=value
```

7. 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

- [Microsoft Azure CLI](#).

## 7.5.5. Uploading VHD images to Microsoft Azure cloud

After you have created your customized VHD image, you can upload it to the Microsoft Azure cloud.

### Prerequisites

- Your system must be set up for uploading Microsoft Azure VHD images. See [Preparing to upload Microsoft Azure VHD images](#).
- You must have a Microsoft Azure VHD image created by image builder.
  - In the CLI, use the **vhd** output type.
  - In the GUI, use the **Azure Disk Image (.vhd)** image type.

### Procedure

1. Push the image to Microsoft 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
...
```

2. After the upload to the Microsoft Azure Blob storage completes, create a Microsoft 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 ...
```

### Verification

1. Create an instance either with the Microsoft 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 ...
```



2. Use your private key via SSH to access the resulting instance. Log in as **azure-user**.

### Additional Resources

- [Composing an image for the .vhd format fails.](#)

## 7.5.6. Uploading VMDK images and creating a RHEL virtual machine in vSphere

Upload a .vmdk image to VMware vSphere using the **govc import.vmdk** CLI tool.



### NOTE

Uploading the image via the UI is not supported.

### Prerequisites

- You created a blueprint with username and password customizations.
- You created an **.vmdk** image by using image builder and downloaded it to your host system.
- You installed the **govc import.vmdk** CLI tool.
- You configured the **govc import.vmdk** CLI tool client.
  - you must set the following values in the environment:

```
GOVC_URL
GOVC_DATACENTER
GOVC_FOLDER
GOVC_DATASTORE
GOVC_RESOURCE_POOL
GOVC_NETWORK
```

### Procedure

1. Navigate to the directory where you downloaded your **.vmdk** image.
2. Launch the image on vSphere by following the steps:
  - a. Import the **.vmdk** image in to vSphere:

```
$ govc import.vmdk ./composer-api.vmdk foldername
```

- b. Create the VM in vSphere without powering it on:

```
govc vm.create \
-net.adapter=vmxnet3 \
-m=4096 -c=2 -g=rhel8_64Guest \
-firmware=efi -disk="foldername/composer-api.vmdk" \
-disk.controller=scsi -on=false \
  vmname
```

- c. Power-on the VM:

```
govc vm.power -on vmname
```

- d. Retrieve the VM IP address:

```
HOST=$(govc vm.ip vmname)
```

- e. Use SSH to log in to the VM, using the username and password you specified in your blueprint:

```
$ ssh admin@HOST
```



#### NOTE

If you copied the **.vmdk** image from your local host to the destination using the **govc datastore.upload** command, using the image is not supported. There is no option to use the **import.vmdk** command in the vSphere GUI and as a consequence, the vSphere GUI does not support the direct upload, as a consequence, the **.vmdk** image is not directly usable from the vSphere GUI.

### 7.5.7. Uploading images to GCP with image builder

With image builder you can build a **gce** image, provide credentials for your user or GCP service account, and then upload the **gce** image directly to the GCP environment.

#### 7.5.7.1. Uploading a gce image to GCP using the CLI

Follow the procedure to set up a configuration file with credentials to upload your **gce** image to GCP.

##### Prerequisites

- You have a user or service account Google credentials to upload images to GCP. The account associated with the credentials must have at least the following IAM roles assigned:
  - **roles/storage.admin** - to create and delete storage objects
  - **roles/compute.storageAdmin** - to import a VM image to Compute Engine.
- You have an existing GCP bucket.

##### Procedure

1. Using a text editor, create a **gcp-config.toml** configuration file with the following content:

```
provider = "gcp"

[settings]
bucket = "GCP_BUCKET"
region = "GCP_STORAGE_REGION"
object = "OBJECT_KEY"
credentials = "GCP_CREDENTIALS"
```

Where:

- **GCP\_BUCKET** points to an existing bucket. It is used to store the intermediate storage object of the image which is being uploaded.
- **GCP\_STORAGE\_REGION** is both a regular Google storage region and, a dual or multi region.
- **OBJECT\_KEY** is the name of an intermediate storage object. It must not exist before the upload, and it is deleted when the upload process is done. If the object name does not end with **.tar.gz**, the extension is automatically added to the object name.
- **GCP\_CREDENTIALS** is a Base64-encoded scheme of the credentials JSON file downloaded from GCP. The credentials determine which project the GCP uploads the image to.



#### NOTE

Specifying **GCP\_CREDENTIALS** in the **gcp-config.toml** is optional if you use a different mechanism to authenticate with GCP. For more details on different ways to authenticate with GCP, see [Authentication with GCP](#).

2. Create a compose with an additional image name and cloud provider profile:

```
$ sudo composer-cli compose start BLUEPRINT-NAME gce IMAGE_KEY gcp-config.toml
```

Note: The image build, upload, and cloud registration processes can take up to ten minutes to complete.

#### Verification

- Verify that the image status is FINISHED:

```
$ sudo composer-cli compose status
```

#### Additional resources

- [Identity and Access Management](#).
- [Create storage buckets](#).

#### 7.5.7.2. Authenticating with GCP

You can use several different types of credentials with image builder to authenticate with GCP. If image builder configuration is set to authenticate with GCP using multiple sets of credentials, it uses the credentials in the following order of preference:

1. Credentials specified with the **composer-cli** command in the configuration file.
2. Credentials configured in the **osbuild-composer** worker configuration.
3. Application Default Credentials from the **Google GCP SDK** library, which tries to automatically find a way to authenticate using the following options:
  - a. If the `GOOGLE_APPLICATION_CREDENTIALS` environment variable is set, Application Default Credentials tries to load and use credentials from the file pointed to by the variable.

- b. Application Default Credentials tries to authenticate using the service account attached to the resource that is running the code. For example, Google Compute Engine VM.



#### NOTE

You must use the GCP credentials to determine which GCP project to upload the image to. Therefore, unless you want to upload all of your images to the same GCP project, you always must specify the credentials in the **gcp-config.toml** configuration file with the **composer-cli** command.

#### 7.5.7.2.1. Specifying credentials with the **composer-cli** command

You can specify GCP authentication credentials in the provided upload target configuration **gcp-config.toml**. Use a **Base64**-encoded scheme of the Google account credentials JSON file to save time.

##### Procedure

- In the provided upload target configuration **gcp-config.toml**, set the credentials:

```
provider = "gcp"

[settings]
provider = "gcp"

[settings]
...
credentials = "GCP_CREDENTIALS"
```

- To get the encoded content of the Google account credentials file with the path stored in **GOOGLE\_APPLICATION\_CREDENTIALS** environment variable, run the following command:

```
$ base64 -w 0 "${GOOGLE_APPLICATION_CREDENTIALS}"
```

#### 7.5.7.2.2. Specifying credentials in the **osbuild-composer** worker configuration

You can configure GCP authentication credentials to be used for GCP globally for all image builds. This way, if you want to import images to the same GCP project, you can use the same credentials for all image uploads to GCP.

##### Procedure

- In the **/etc/osbuild-worker/osbuild-worker.toml** worker configuration, set the following credential value:

```
[gcp]
credentials = "PATH_TO_GCP_ACCOUNT_CREDENTIALS"
```

### 7.5.8. Pushing VMDK images to vSphere using the GUI image builder tool

You can build VMware images by using the GUI image builder tool and push the images directly to your vSphere instance, to avoid having to download the image file and push it manually. To create **.vmdk** images using image builder directly to vSphere instances service provider, follow the steps:

## Prerequisites

- You have **root** or **wheel** group user access to the system.
- You have opened the [image builder](#) interface of the RHEL web console in a browser.
- You have created a blueprint. See [Creating an image builder blueprint in the web console interface](#).
- You have a [vSphere Account](#).

## Procedure

1. For the blueprint you created, click the **Images** tab .
2. Click **Create Image** to create your customized image.  
The Image type window opens.
3. In the **Image type** window:
  - a. From the dropdown menu, select the Type: VMware VSphere (.vmdk).
  - b. Check the **Upload to VMware** checkbox to upload your image to the vSphere.
  - c. Optional: Set the size of the image you want to instantiate. The minimal default size is 2GB.
  - d. Click **Next**.
4. In the **Upload to VMware** window, under **Authentication**, enter the following details:
  - a. **Username**: username of the vSphere account.
  - b. **Password**: password of the vSphere account.
5. In the **Upload to VMware** window, under **Destination**, enter the following details:
  - a. **Image name**: a name for the image to be uploaded.
  - b. **Host**: The URL of your VMware vSphere where the image will be uploaded.
  - c. **Cluster**: The name of the cluster where the image will be uploaded.
  - d. **Data center**: The name of the data center where the image will be uploaded.
  - e. **Data store**: The name of the Data store where the image will be uploaded.
  - f. Click **Next**.
6. In the **Review** window, review the details of the image creation and click **Finish**.  
You can click **Back** to modify any incorrect detail.



### NOTE

The image build and upload processes take a few minutes to complete.

After the process is complete, you can see the **Image build complete** status.

## Verification

After the image status upload is completed successfully, you can create a Virtual Machine (VM) from the image you uploaded and login into it. To do so:

1. Access VMware vSphere Client.
2. Search for the image in the Cluster on the vSphere instance you specified.
3. You can create a new VM from the image you uploaded:
  - a. Select the image you uploaded.
  - b. Right-click the selected image.
  - c. Click **New Virtual Machine**.  
A **New Virtual Machine** window opens.

In the **New Virtual Machine** window, provide the following details:

- i. Select **New Virtual Machine**.
  - ii. Select a name and a folder for your VM.
  - iii. Select a computer resource: choose a destination computer resource for this operation.
  - iv. Select storage: For example, select NFS-Node1
  - v. Select compatibility: The image should be BIOS only.
  - vi. Select a guest operating system: For example, select *Linux* and *Red Hat Fedora (64-bit)*.
  - vii. **Customize hardware**: When creating a VM, on the **Device Configuration** button on the upper right, delete the default New Hard Disk and use the drop-down to select an Existing Hard Disk disk image:
  - viii. Ready to complete: Review the details and click **Finish** to create the image.
- d. Navigate to the **VMs** tab.
    - i. From the list, select the VM you created.
    - ii. Click the **Start** button from the panel. A new window appears, showing the VM image loading.
    - iii. Log in with the credentials you created for the blueprint.
    - iv. You can verify if the packages you added to the blueprint are installed. For example:

```
$ rpm -qa | grep firefox
```

## Additional resources

- [Installing the vSphere Client.](#)

## 7.5.9. Pushing VHD images to Microsoft Azure cloud using the GUI image builder tool

You can create **.vhd** images using image builder. Then, you can push the **.vhd** images to a Blob Storage of the Microsoft Azure Cloud service provider.

### Prerequisites

- You have root access to the system.
- You have opened the image builder interface of the RHEL web console in a browser.
- You created a blueprint. See [Creating an image builder blueprint in the web console interface](#) .
- You have a [Microsoft Storage Account](#) created.
- You have a writable [Blob Storage](#) prepared.

### Procedure

1. For the **blueprint name**, click the **Images** tab .
2. Click **Create Image** to create your customized image.  
A pop-up window opens.
  - a. From the "**Type**" drop-down menu list, select the **Azure Disk Image (.vhd)** image.
  - b. Check the **Upload to Microsoft Azure** check box to upload your image to the Microsoft Azure Cloud and click **Next**.
  - c. To authenticate your access to Microsoft Azure, type your "Storage account" and "Storage access key" in the corresponding fields. Click **Next**.  
You can find your [Microsoft Storage account details](#) in the Settings→Access Key menu list.
  - d. Type a "**Image name**" to be used for the image file that will be uploaded and the Blob "Storage container" in which the image file you want to push the image into. Click **Next**.
  - e. Review the information you provided and click **Finish**.  
Optionally, you can click **Back** to modify any incorrect detail.
3. When the image creation process starts, a small pop-up on the upper right side displays with the message: **Image creation has been added to the queue**.  
After the image process creation is complete, click the blueprint you created the image from. In the **Images** tab, you can see the **Image build complete** status for the image you created.
4. To access the image you pushed into **Microsoft Azure Cloud**, access the [Microsoft Azure portal](#).
5. On the search bar, type **Images** and select the first entry under **Services**. You are redirected to the **Image dashboard**.
6. Click **+Add**. You are redirected to the **Create an Image** dashboard.  
Insert the below details:
  - a. **Name**: Choose a name for your new image.

- b. **Resource Group:** Select a **resource group**.
  - c. **Location:** Select the **location** that matches the regions assigned to your storage account. Otherwise you will not be able to select a blob.
  - d. **OS Type:** Set the operating system type to **Linux**.
  - e. **VM Generation:** Keep the VM generation set on **Gen 1**.
  - f. **Storage Blob:** Click **Browse** on the right of **Storage blob input**. Use the dialog to find the image you uploaded earlier.  
Keep the remaining fields as in the default choice.
7. Click **Create** to create the image. After the image is created, you can see the message **Successfully created image** in the upper right corner.
8. Click **Refresh** to see your newly created image and open it.
9. Click **+ Create VM**. You are redirected to the **Create a virtual machine** dashboard.
10. In the **Basic** tab, under **Project Details**, your **Subscription** and the **Resource Group** are already pre-set.  
If you want to create a new **Resource Group**:
  - a. Click **Create new**.  
A pop-up prompts you to create the **Resource Group Name** container.
  - b. Insert a name and click **OK**.  
If you want to keep the already pre-set **Resource Group**:
11. Under **Instance Details**, enter:
  - a. **Virtual machine name**
  - b. **Region**
  - c. **Image:** The image you created is pre-selected by default.
  - d. **Size:** Choose a VM size that better suits your needs.  
Keep the remaining fields default.
12. Under **Administrator account**, enter the below details:
  - a. **Username:** the name of the account administrator.
  - b. **SSH public key source:** from the drop-down menu, select **Generate new key pair**.  
You can either use the key pair you already have or you can create a new key pair.  
Alternatively, you can use **image builder** to add a user to the image with a preset public key.  
See [Creating a user account with SSH key](#) for more details.
  - c. **Key pair name:** insert a name for the key pair.
13. Under **Inbound port rules**, select values for each of the fields:
  - a. **Public inbound ports:** **Allow selected ports**
  - b. **Select inbound ports:** Use the default set **SSH (22)**.



14. Click **Review + Create**. You are redirected to the **Review + create** tab and receive a confirmation that the validation passed.
15. Review the details and click **Create**.  
Optionally, you can click **Previous** to fix previous options selected.
16. A **generates new key pair** window opens. Click **Download private key and create resources**. Save the key file as "*yourKey.pem*".
17. After the deployment is complete, click **Go to resource**.
18. You are redirected to a new window with your VM details. Select the public IP address on the upper right side of the page and copy it to your clipboard.

Now, to create an SSH connection with the VM to connect to the Virtual Machine.

1. Open a terminal.
2. At your prompt, open an SSH connection to your VM. Replace the IP address with the one from your VM, and replace the path to the `.pem` with the path to where the key file was downloaded.

```
# ssh -i ./Downloads/yourKey.pem azureuser@10.111.12.123
```

3. You are required to confirm if you want to continue to connect. Type **yes** to continue.

As a result, the output image you pushed to the Microsoft Azure Storage Blob is ready to be provisioned.

### Additional resources

- [Microsoft Azure Storage Documentation](#) .
- [Create a Microsoft Azure Storage account](#) .
- [Open a case on Red Hat Customer Portal](#) .
- [Help + support](#).
- [Contacting Red Hat](#).

### 7.5.10. Uploading QCOW2 images to OpenStack

With the image builder tool, you can create customized **.qcow2** images that are suitable for uploading to OpenStack cloud deployments, and starting instances there.



#### WARNING

Do not mistake the generic **QCOW2** image type output format you create by using image builder with the OpenStack image type, which is also in the QCOW2 format, but contains further changes specific to OpenStack.

## Prerequisites

- You have created a blueprint.
- created a **QCOW2** image by using image builder. See

## Procedure

1. Start the compose of a **QCOW2** image.

```
# composer-cli compose start blueprint_name openstack
```

2. Check the status of the building.

```
# composer-cli compose status
```

After the image build finishes, you can download the image.

3. Download the **QCOW2** image:

```
# composer-cli compose image UUID
```

4. Access the OpenStack dashboard and click **+Create Image**.
5. On the left menu, select the **Admin** tab.
  - a. From the **System Panel**, click **Image**.  
The **Create An Image** wizard opens.
6. In the **Create An Image** wizard:
  - a. Enter a name for the image
  - b. Click **Browse** to upload the **QCOW2** image.
  - c. From the **Format** dropdown list, select the **QCOW2 - QEMU Emulator**.
  - d. Click **Create Image**.

**Create An Image** ✕

**Name: \***

**Description:**

**Image Source:**

**Image File**  
 96268ffb-2c71-4e97-a85...c25e98

**Format: \***

**Architecture:**

**Minimum Disk (GB):**

**Minimum Ram (MB):**

**Public:**

**Protected:**

**Description:**  
 Specify an image to upload to the Image Service.  
 Currently only images available via an HTTP URL are supported. The image location must be accessible to the Image Service. Compressed image binaries are supported (.zip and .tar.gz.)

**Please note:** The Image Location field MUST be a valid and direct URL to the image binary. URLs that redirect or serve error pages will result in unusable images.

7. On the left menu, select the **Project** tab.
  - a. From the **Compute** menu, select **Instances**.
  - b. Click the **Launch Instance** button.  
The **Launch Instance** wizard opens.
  - c. On the **Details** page, enter a name for the instance. Click **Next**.
  - d. On the **Source** page, select the name of the image you uploaded. Click **Next**.
  - e. On the **Flavor** page, select the machine resources that best fit your needs. Click **Launch**.

### 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.qc

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**

|                       |          |
|-----------------------|----------|
| <b>Name</b>           | m1.small |
| <b>VCPUs</b>          | 1        |
| <b>Root Disk</b>      | 20 GB    |
| <b>Ephemeral Disk</b> | 0 GB     |
| <b>Total Disk</b>     | 20 GB    |
| <b>RAM</b>            | 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

8. You can run the image instance using any mechanism (CLI or OpenStack web UI) from the image. Use your private key via SSH to access the resulting instance. Log in as **cloud-user**.

### 7.5.11. Preparing to upload customized RHEL images to Alibaba

To deploy a customized RHEL image to the **Alibaba Cloud**, first you need to verify the customized image. The image needs a specific configuration to boot successfully, because Alibaba Cloud requests the custom images to meet certain requirements before you use it.



#### NOTE

Image builder generates images that conform to Alibaba's requirements. However, Red Hat recommends also using the Alibaba **image\_check tool** to verify the format compliance of your image.

#### Prerequisites

- You must have created an Alibaba image by using image builder.

#### Procedure

1. Connect to the system containing the image that you want to check by using 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 generates 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.

## Troubleshooting

If any of the **Detection Items** fail, follow the instructions in the terminal to correct it. See link: [Detection items section](#).

## Additional resources

- [Image Compliance Tool](#).

## 7.5.12. Uploading customized RHEL images to Alibaba

You can upload the customized **AMI** image you created by using image builder to the Object Storage Service (OSS).

### Prerequisites

- Your system is set up for uploading Alibaba images. See [Preparing for uploading images to Alibaba](#).
- You have created an **ami** image by using image builder.
- 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. In Bucket menu on the left, select the bucket to which you want to upload an image.
3. In the upper right menu, click the **Files** tab.
4. Click **Upload**. A dialog window opens on the right side. Configure the following:
  - **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. Select the image you want to upload.
7. Click **Open**.

As a result, the customized **AMI** image is uploaded to the OSS Console.

#### Additional resources

- [Upload an object](#).
- [Creating an instance from custom images](#).
- [Importing images](#).

### 7.5.13. Importing images to Alibaba

To import a customized Alibaba RHEL image that you created by using image builder to the Elastic Cloud Console (ECS), follow the steps:

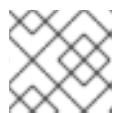
#### Prerequisites

- Your system is set up for uploading Alibaba images. See [Preparing for uploading images to Alibaba](#).
- You have created an **ami** image by using image builder.
- You have a bucket. See [Creating a bucket](#).
- You have an [active Alibaba Account](#).
- You activated [OSS](#).
- You have uploaded the image to Object Storage Service (OSS). See [Uploading images to Alibaba](#).

#### Procedure

1. Log in to the [ECS console](#).
  - i. On the left-side menu, click **Images**.
  - ii. On the upper right side, click **Import Image**. A dialog window 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:
    2. Select **Files** section.
    3. Click the **Details** link on the right for the appropriate image.  
A window appears on the right side of the screen, showing image details. The **OSS** object address is in the **URL** box.
    4. Click **OK**.

**NOTE**

The importing process time can vary depending on the image size.

The customized image is imported to the **ECS** Console.

**Additional resources**

- [Notes for importing images.](#)
- [Creating an instance from custom images.](#)
- [Upload an object.](#)

**7.5.14. Creating an instance of a customized RHEL image using Alibaba**

You can create instances of a customized RHEL image using **Alibaba ECS Console**.

**Prerequisites**

- You have activated [OSS](#) and uploaded your custom image.
- You have successfully imported your image to ECS Console. See [Importing images to Alibaba](#) .

**Procedure**

1. Log in to the [ECS console](#).
2. On the left-side menu, select **Instances**.
3. In the upper-right corner, click **Create Instance**. You are redirected to a new window.
4. Complete 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 from the **Alibaba ECS Console**.

#### Additional resources

- [Creating an instance by using a custom image.](#)
- [Create an instance by using the wizard.](#)



# CHAPTER 8. PERFORMING AN AUTOMATED INSTALLATION USING KICKSTART

## 8.1. KICKSTART INSTALLATION BASICS

The following provides basic information about Kickstart and how to use it to automate installing Red Hat Enterprise Linux.

### 8.1.1. 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. The kickstart used for installation as well as the Anaconda generated output kickstart are stored in **/root** on the target system and that logs from kickstart scriptlet execution are stored in **/var/log/anaconda**.



#### 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](#).

### 8.1.2. 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.



#### NOTE

If you plan to install a Beta release of Red Hat Enterprise Linux, on systems having UEFI Secure Boot enabled, then first disable the UEFI Secure Boot option and then begin the installation.

UEFI Secure Boot requires that the operating system kernel is signed with a recognized private key, which the system's firmware verifies using the corresponding public key. For Red Hat Enterprise Linux Beta releases, the kernel is signed with a Red Hat Beta-specific private key, which the system fails to recognize by default. As a result, the system fails to boot the installation media.

## 8.2. 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.
- Convert the Red Hat Enterprise Linux 7 Kickstart file for Red Hat Enterprise Linux 8 installation. For more information on the conversion tool, see [Kickstart generator lab](#).
- In case of virtual and cloud environment, create a custom system image, using Image Builder.

Note that some highly specific installation options can be configured only by manual editing of the Kickstart file.

### 8.2.1. 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.

#### Prerequisites

- You have a Red Hat Customer Portal account and an active Red Hat subscription.

#### Procedure

1. Open the Kickstart generator lab information page at <https://access.redhat.com/labsinfo/kickstartconfig>.

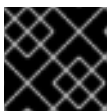
2. Click the **Go to Application** button to the left of heading and wait for the next page to load.
3. Select **Red Hat Enterprise Linux 8** in the drop-down menu and wait for the page to update.
4. Describe the system to be installed using the fields in the form.  
You can use the links on the left 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 saves the file.

## 8.2.2. 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 8 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.



### IMPORTANT

The file contains information about users and passwords.

- 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.

### Additional resources

- [Performing a standard RHEL 8 installation](#)

## 8.2.3. Converting a Kickstart file from previous RHEL installation

You can use the Kickstart Converter tool to convert a RHEL 7 Kickstart file for use in a RHEL 8 or 9 installation or convert a RHEL 8 Kickstart file for use it in RHEL 9. For more information about the tool and how to use it to convert a RHEL Kickstart file, see <https://access.redhat.com/labs/kickstartconvert/>

## 8.2.4. Creating a custom image using Image Builder

You can use Red Hat Image Builder to create a customized system image for virtual and cloud deployments.

For more information about creating customized images, using Image Builder, see [Composing a customized RHEL system image](#) document.

## 8.3. 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.

### 8.3.1. Ports for network-based installation

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

**Table 8.1. Ports for network-based installation**

| Protocol used | Ports to open    |
|---------------|------------------|
| HTTP          | 80               |
| HTTPS         | 443              |
| FTP           | 21               |
| NFS           | 2049, 111, 20048 |
| TFTP          | 69               |

#### Additional resources

- [Securing networks](#)

### 8.3.2. 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 have an administrator-level access to a server with Red Hat Enterprise Linux 8 on the local network.
- The system to be installed can connect to the server.
- The firewall on the server allows 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

- [Preparing to install from the network using PXE](#)

### 8.3.3. 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 have an administrator-level access to a server with Red Hat Enterprise Linux 8 on the local network.
- The system to be installed can connect to the server.
- The firewall on the server allows connections from the system you are installing to.

## Procedure

1. To store the Kickstart file on an HTTP, install the **httpd** package:

```
# yum install httpd
```

To store the Kickstart file on an HTTPS, install **httpd** and **mod\_ssl** packages:

```
# yum install httpd mod_ssl
```



### 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

- [Deploying different types of servers](#)

### 8.3.4. 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 have an administrator-level access to a server with Red Hat Enterprise Linux 8 on the local network.
- The system to be installed can connect to the server.
- The firewall on the server allows 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.

### 8.3.5. 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 have a drive that can be moved to the machine to be installed, such as a USB stick.
- The drive contains a partition that can be read by the installation program. The supported types are **ext2**, **ext3**, **ext4**, **xfs**, and **fat**.
- The drive is connected to the system and its volumes are 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.

### 8.3.6. 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 have a drive that can be moved to the machine to be installed, such as a USB stick.
- The drive contains a partition that can be read by the installation program. The supported types are **ext2**, **ext3**, **ext4**, **xfs**, and **fat**.
- The drive is connected to the system and its volumes are mounted.

#### Procedure

1. List volume information 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.

## 8.4. CREATING INSTALLATION SOURCES FOR KICKSTART INSTALLATIONS

This section describes how to create an installation source for the Boot ISO image using the DVD ISO image that contains the required repositories and software packages.

### 8.4.1. Types of installation source

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

- **DVD:** Burn the DVD ISO image to a DVD. The DVD will be automatically used as the installation source (software package source).
- **Hard drive or USB drive:** Copy the 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 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 DVD ISO image or the installation tree (extracted contents of the DVD ISO image) to a network location and perform the installation over the network using the following protocols:
  - **NFS:** The 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.

### 8.4.2. Ports for network-based installation

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

**Table 8.2. Ports for network-based installation**

| Protocol used | Ports to open    |
|---------------|------------------|
| HTTP          | 80               |
| HTTPS         | 443              |
| FTP           | 21               |
| NFS           | 2049, 111, 20048 |
| TFTP          | 69               |

#### Additional resources

- [Securing networks](#)

### 8.4.3. Creating an 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 an 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. For more information, see [Downloading the installation ISO image](#).
- You have created a bootable CD, DVD, or USB device from the image file. For more information, see [Creating installation media](#).
- You have verified that your firewall allows the system you are installing to access the remote installation source. For more information, see [Ports for network-based installation](#).

#### Procedure

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

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

2. Copy the 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
```

- Replace */exported\_directory/* with the full path to the directory with the ISO image.
- Replace *clients* with one of the following:
  - The host name or IP address of the target system
  - The subnetwork that all target systems can use to access the ISO image
  - To allow any system with network access to the NFS server to use the ISO image, the asterisk sign (\*)

See the **exports(5)** man page for detailed information about the format of this field.

For example, a basic configuration that makes the **/rhel8-install/** directory available as read-only to all clients is:

```
/rhel8-install *
```

4. Save the **/etc/exports** file and exit the text editor.
5. Start the nfs service:

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

If the service was running before you changed the **/etc/exports** file, reload the NFS server 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.

#### 8.4.4. Creating an installation source using HTTP or HTTPS

You can create an installation source for a network-based installation using an installation tree, which is a directory containing extracted contents of the DVD ISO image and a valid **.treeinfo** file. The installation source is accessed over HTTP or HTTPS.

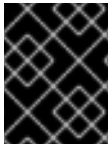
## Prerequisites

- You have an 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. For more information, see [Downloading the installation ISO image](#).
- You have created a bootable CD, DVD, or USB device from the image file. For more information, see [Creating installation media](#).
- You have verified that your firewall allows the system you are installing to access the remote installation source. For more information, see [Ports for network-based installation](#).
- The **httpd** package is installed.
- The **mod\_ssl** package is installed, if you use the **https** installation source.



### WARNING

If your Apache web server configuration enables SSL security, prefer to enable the TLSv1.3 protocol. By default, TLSv1.2 is enabled and you may use the TLSv1 (LEGACY) protocol.



### IMPORTANT

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

## Procedure

1. Copy the DVD ISO image to the HTTP(S) server.
2. Create a suitable directory for mounting the DVD ISO image, for example:

```
# mkdir /mnt/rhel8-install/
```

3. Mount the DVD ISO image to the directory:

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

Replace */image\_directory/image.iso* with the path to the DVD ISO image.

4. Copy the files from the mounted image to the HTTP(S) server root.

```
# 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 other copying methods might skip the **.treeinfo** file which is required for a valid installation source. Entering the **cp** command for entire directories as shown in this

procedure copies **.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 use 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

- [Deploying different types of servers](#)

### 8.4.5. Creating an installation source using FTP

You can create an installation source for a network-based installation using an installation tree, which is a directory containing extracted contents of the DVD ISO image and a valid **.treeinfo** file. The installation source is accessed over FTP.

#### Prerequisites

- You have an 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. For more information, see [Downloading the installation ISO image](#).
- You have created a bootable CD, DVD, or USB device from the image file. For more information, see [Creating installation media](#).
- You have verified that your firewall allows the system you are installing to access the remote installation source. For more information, see [Ports for network-based installation](#).
- The **vsftpd** package is installed.

#### Procedure

1. 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, for example, **10021** and **10031**.

This step might be necessary in network environments featuring various firewall/NAT setups.

- d. Optional: 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.

2. Configure the server firewall.

- a. Enable the firewall:

```
# systemctl enable firewalld
```

- b. Start the firewall:

```
# systemctl start firewalld
```

- c. Configure the firewall to allow the FTP port and port range from the previous step:

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

Replace *<min\_port>* and *<max\_port>* with the port numbers you entered into the **/etc/vsftpd/vsftpd.conf** configuration file.

- d. Reload the firewall to apply the new rules:

```
# firewall-cmd --reload
```

3. Copy the DVD ISO image to the FTP server.
4. Create a suitable directory for mounting the DVD ISO image, for example:

```
# mkdir /mnt/rhel8-install
```

5. Mount the DVD ISO image to the directory:

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

Replace ***/image-directory/image.iso*** with the path to the 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. Entering 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.

## 8.5. 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.

### 8.5.1. 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 and the form of boot prompt depends on your system's architecture. For detailed information, see the [Boot options for RHEL installer](#) guide.

### 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.

For information about editing boot options, see [Editing boot options](#).

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.



### NOTE

If you have installed a Red Hat Enterprise Linux Beta release, on systems having UEFI Secure Boot enabled, then add the Beta public key to the system's Machine Owner Key (MOK) list. For more information about UEFI Secure Boot and Red Hat Enterprise Linux Beta releases, see the [Completing post-installation tasks](#) section of the *Performing a standard RHEL 8 installation* document.

## 8.5.2. 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 64-bit IBM Z).

### Prerequisites

- You have a Kickstart file ready in a location accessible from the system to be installed.

- You have a PXE server that 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.



### NOTE

If you have installed a Red Hat Enterprise Linux Beta release, on systems having UEFI Secure Boot enabled, then add the Beta public key to the system's Machine Owner Key (MOK) list.

For more information about UEFI Secure Boot and Red Hat Enterprise Linux Beta releases, see the [Completing post-installation tasks](#) section of the *Performing a standard RHEL 8 installation* document.

### 8.5.3. 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 have a volume prepared with label **OEMDRV** and the Kickstart file present in its root as **ks.cfg**.
- A drive containing this volume is 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).

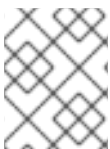
2. At the boot prompt, specify the required boot options.
  - a. If a required repository is in a network location, you may need to configure the network using the **ip=** option. The installer tries to configure all network devices using the DHCP protocol by default without this option.
  - b. In order 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.  
For more information about installation sources, see [Kickstart commands for installation program configuration and flow control](#).
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.

**NOTE**

If you have installed a Red Hat Enterprise Linux Beta release, on systems having UEFI Secure Boot enabled, then add the Beta public key to the system's Machine Owner Key (MOK) list. For more information about UEFI Secure Boot and Red Hat Enterprise Linux Beta releases, see the [Completing post-installation tasks](#) section of the *Performing a standard RHEL 8 installation* document.

## 8.6. 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 8.3. Available tmux windows**

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

## 8.7. 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.

### 8.7.1. 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
```

### 8.7.2. 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. Use the **-v RHEL8** option in the **ksvalidator** command to acknowledge new commands of the RHEL8 class.

#### Procedure

- Run **ksvalidator** on your Kickstart file:

```
$ ksvalidator -v RHEL8 /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)* man page

## 8.8. 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.

### 8.8.1. 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 **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 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 have 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 8 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 (recommended)

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 manual installation using the installer GUI, 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"**

### System Purpose (optional)

Set the System Purpose role, SLA, and usage using the command:

```
subscription-manager syspurpose role --set="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 --
proxy="user:password@hostname:9000"
reboot
%packages
vim
%end
```

3. Save the Kickstart file and start the installation process.

### Additional resources

- [Configuring System Purpose](#)
- [Starting Kickstart installations](#)
- [Red Hat Insights product documentation](#)
- [Understanding Activation Keys](#)
- For information about setting up an HTTP proxy for Subscription Manager, see the **PROXY CONFIGURATION** section in the **subscription-manager** man page.

### 8.8.2. 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 [Register and install using 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: X
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
```

### 8.8.3. 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 [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

- 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.

## 8.9. PERFORMING A REMOTE RHEL INSTALLATION USING VNC

This section describes how to perform a remote RHEL installation using Virtual Network Computing (VNC).

### 8.9.1. 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 64-bit 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.

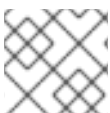
### Connect mode

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.

## 8.9.2. 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.

### 8.9.3. 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

- You have installed a VNC viewer on a remote system as a root user.
- You have set up a network boot server and booted the installation on the target system.

#### 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.

### 8.9.4. 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

- You have installed a VNC viewer on a remote system as a root user.
- You have set up a network boot server to start the installation on the target system.
- 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 [Security hardening](#) and [Securing networks](#).

## 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.

## CHAPTER 9. ADVANCED CONFIGURATION OPTIONS

### 9.1. 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.

#### 9.1.1. Overview

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

- During image creation
- During a GUI installation when using the **Connect to Red Hat** screen to register your system and attach your Red Hat subscription
- During a Kickstart installation when using the **syspurpose Kickstart** command
- 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

- [Composing a customized RHEL system image](#)
- [Performing an advanced RHEL 8 installation](#)
- [Using and Configuring Red Hat Subscription Manager](#)

## 9.1.2. Configuring System Purpose in a Kickstart file

Follow the steps in this procedure to configure System Purpose during the installation. To do so, use the **syspurpose** Kickstart command in the Kickstart configuration file.

Even though System Purpose is an optional feature of the Red Hat Enterprise Linux installation program, we strongly recommend that you configure System Purpose to auto-attach the most appropriate subscription.



### NOTE

You can also enable System Purpose after the installation is complete. To do so use the **syspurpose** command-line tool. The **syspurpose** tool commands are different from the **syspurpose** Kickstart commands.

The following actions are available for the **syspurpose** Kickstart command:

#### 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**

### addon

Any additional layered products or features. To add multiple items specify **--addon** multiple times, once per layered product/feature. This action uses the following format:

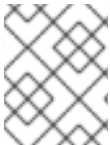
```
syspurpose --addon=
```

## 9.1.3. Additional resources

- [Configuring System Purpose using the \*\*subscription-manager\*\* command-line tool](#)

## 9.2. 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.

### Prerequisites

- 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.

### 9.2.1. 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.

### 9.2.2. 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.

### 9.2.3. Preparing a driver update

This procedure describes how to prepare a driver update on a CD and DVD.

### Prerequisites

- You have received the driver update ISO image from Red Hat, your hardware vendor, or a trusted third-party vendor.
- You have 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.

## 9.2.4. 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

- When you complete the prerequisite steps, the drivers load automatically when the installation program starts and installs during the system's installation process.

## 9.2.5. 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 or 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 [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.

### 9.2.6. Performing a manual driver update

This procedure describes how to perform a manual driver update during installation.

#### Prerequisites

- You have placed the driver update ISO image file on a USB flash drive or a web server and connected 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

- [The `inst.dd` boot option](#)

## 9.2.7. 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.

## 9.3. 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.

### 9.3.1. 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

- [Creating installation sources for Kickstart installations](#)
- [Configuring a TFTP server for BIOS-based clients](#)
- [Configuring a TFTP server for UEFI-based clients](#)
- [Configuring a network server for IBM Power systems](#)
- [Red Hat Satellite product documentation](#)

### 9.3.2. 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. If you already have a DHCP server configured in your network, exclude the **dhcp-server** packages:

```
# yum install tftp-server dhcp-server
```

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. Note that if you already have a DHCP server configured, then perform this step on the DHCP server.

```
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 **pxelinux.0** file from the **SYSLINUX** package in the 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:

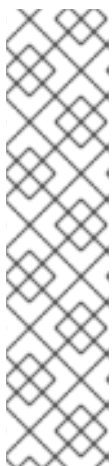
```
# mkdir /var/lib/tftpboot/pxelinux/pxelinux.cfg
```

8. Create a configuration file named **default** and add it 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/vmlinuz
  append initrd=images/RHEL-8/initrd.img ip=dhcp inst.repo=http://10.32.5.1/RHEL-8/x86_64/iso-contents-root/
label vesa
  menu label Install system with ^basic video driver
  kernel images/RHEL-8/vmlinuz
  append initrd=images/RHEL-8/initrd.img ip=dhcp inst.xdriver=vesa nomodeset
inst.repo=http://10.32.5.1/RHEL-8/x86_64/iso-contents-root/
label rescue
  menu label ^Rescue installed system
  kernel images/RHEL-8/vmlinuz
  append initrd=images/RHEL-8/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/`:

```
# mkdir -p /var/lib/tftpboot/pxelinux/images/RHEL-8/
# cp /path_to_x86_64_images/pxeboot/{vmlinuz,initrd.img}
/var/lib/tftpboot/pxelinux/images/RHEL-8/
```

10. On the DHCP server, start and enable the **dhcpd** service. If you have configured a DHCP server on the localhost, then start and enable the **dhcpd** service on the localhost.

```
# systemctl start dhcpd
# systemctl enable dhcpd
```

11. Start and enable the **tftp.socket** service:

```
# systemctl start tftp.socket
# systemctl enable tftp.socket
```

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.

### 9.3.3. 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. If you already have a DHCP server configured in your network, exclude the `dhcp-server` packages.

```
# yum install tftp-server dhcp-server
```

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. Note that if you already have a DHCP server configured, then perform this step on the DHCP server.

```
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.EFI** file from the **shim** package, and the **grubx64.efi** file from the **grub2-efi** package in the 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
```

- Copy the EFI boot images from your boot directory. Replace *ARCH* with shim or grub followed by the architecture, for example, **grubx64**.

```
# mkdir /var/lib/tftpboot/uefi
# cp my_local_directory/boot/efi/EFI/redhat/ARCH.efi /var/lib/tftpboot/uefi/
```

- 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.x/vmlinuz ip=dhcp inst.repo=http://10.32.5.1/RHEL-8.x/x86_64/iso-
  contents-root/
  initrdefi images/RHEL-8.x/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.

- 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.x/**:

```
# mkdir -p /var/lib/tftpboot/images/RHEL-8/
# cp /path_to_x86_64_images/pxeboot/{vmlinuz,initrd.img} /var/lib/tftpboot/images/RHEL-8/
```

- On the DHCP server, start and enable the **dhcpd** service. If you have configured a DHCP server on the localhost, then start and enable the **dhcpd** service on the localhost.

```
# systemctl start dhcpd
# systemctl enable dhcpd
```

- Start and enable the **tftp.socket** service:

```
# systemctl start tftp.socket
# systemctl enable tftp.socket
```

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.

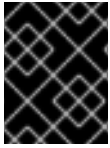
#### Additional resources



- [Using the Shim Program](#)

### 9.3.4. 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. If you already have a DHCP server configured in your network, exclude the `dhcp-server` packages.

```
# yum install tftp-server dhcp-server
```

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/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 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 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. Note that if you already have a DHCP server configured, then perform this step on the DHCP server.

```
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. On the DHCP server, start and enable the **dhcpcd** service. If you have configured a DHCP server on the localhost, then start and enable the **dhcpcd** service on the localhost.

```
# systemctl start dhcpd
# systemctl enable dhcpd
```

10. Start and enable the **tftp.socket** service:

```
# systemctl start tftp.socket
# systemctl enable tftp.socket
```

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.

## 9.4. BOOT OPTIONS

This section describes contains 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.

### 9.4.1. Types of boot options

The two types of boot options are those with an equals "=" sign, and those without an equals "=" sign. Boot options are appended to the boot command line and you can append multiple options separated by 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 example, 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 installation program performs the verification and if the boot option is not present, the verification is skipped.

### 9.4.2. 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.

#### 9.4.2.1. 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.

### 9.4.2.2. Editing predefined boot options using the > prompt

In BIOS-based AMD64 and Intel 64 systems, you can use the > prompt to edit predefined boot options. To display a full set of options, select **Test this media and install RHEL 8** from the boot menu.

#### 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 **Enter** to start the installation.
4. Press **Esc** to cancel editing and return to the boot menu.

### 9.4.2.3. Editing the GRUB2 menu for the UEFI-based systems

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 **e**.
2. On UEFI systems, the kernel command line starts with **linuxefi**. Move the cursor to the end of the **linuxefi** kernel command line.

3. 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.
4. When you finish editing, press **Ctrl+X** to start the installation using the specified options.

### 9.4.3. Installation source boot options

This section describes 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.

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 9.1. Types and format for the **inst.repo=** boot option and installation source**

| Source type                          | Boot option format                                                                 | Source format                                                                                                                                              |
|--------------------------------------|------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CD/DVD drive                         | <b>inst.repo=cdrom:&lt;device&gt;</b>                                              | Installation DVD as a physical disk. <sup>[a]</sup>                                                                                                        |
| Mountable device (HDD and USB stick) | <b>inst.repo=hd:&lt;device&gt;:/&lt;path&gt;</b>                                   | Image file of the installation DVD.                                                                                                                        |
| NFS Server                           | <b>inst.repo=nfs:<br/>[options:]&lt;server&gt;:/&lt;path&gt;</b>                   | Image file of the installation DVD, or an installation tree, which is a complete copy of the directories and files on the installation DVD. <sup>[b]</sup> |
| HTTP Server                          | <b>inst.repo=http://&lt;host&gt;/&lt;path&gt;</b>                                  | Installation tree that is a complete copy of the directories and files on the installation DVD.                                                            |
| HTTPS Server                         | <b>inst.repo=https://&lt;host&gt;/&lt;path&gt;</b>                                 |                                                                                                                                                            |
| FTP Server                           | <b>inst.repo=ftp://&lt;username&gt;:&lt;password&gt;@&lt;host&gt;/&lt;path&gt;</b> |                                                                                                                                                            |
| HMC                                  | <b>inst.repo=hmc</b>                                                               |                                                                                                                                                            |

| Source type                                                                                                                                                                                                    | Boot option format | Source format |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|---------------|
| [a] If <i>device</i> 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, add <b>nfsvers=<i>X</i></b> to <i>options</i> , replacing <i>X</i> with the version number that you want to use. |                    |               |

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 you can use 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.

For more information about unified ISO, see [Unified ISO](#).

**Table 9.2. Installation sources and boot option format**

| Installation source             | Boot option format                                                         | Additional information                                                                                                                                                                                                              |
|---------------------------------|----------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Installable tree at a URL       | <b>inst.addrepo=REPO_NAME,[http,https,ftp]://&lt;host&gt;/&lt;path&gt;</b> | Looks for the installable tree at a given URL.                                                                                                                                                                                      |
| Installable tree at an NFS path | <b>inst.addrepo=REPO_NAME,nfs://&lt;server&gt;:/&lt;path&gt;</b>           | Looks for the installable tree at a given NFS path. A colon is required after the host. The installation program passes everything after <b>nfs://</b> 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 | <b>inst.addrepo=REPO_NAME,file:/// &lt;path&gt;</b>          | 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 <b>/run/install/source/REPO_ISO_PATH</b> . Additionally, you can mount the repository directory in the <b>%pre</b> section in the Kickstart file. The path must be absolute and start with <b>/</b> , for example <b>inst.addrepo=REPO_NAME,file:/// &lt;path&gt;</b> |
| Hard Drive                                       | <b>inst.addrepo=REPO_NAME,hd:&lt;device&gt;:&lt;path&gt;</b> | Mounts the given <i>&lt;device&gt;</i> partition and installs from the ISO that is specified by the <i>&lt;path&gt;</i> . If the <i>&lt;path&gt;</i> is not specified, the installation program looks for a valid installation ISO on the <i>&lt;device&gt;</i> . This installation method requires an ISO with a valid installable tree.                                                                                                                                                                                                                                                                                                                                                                                   |

### inst.stage2=

The **inst.stage2=** boot option specifies the location of the installation program's runtime image. This option expects the path to a directory that contains a valid **.treeinfo** file and reads the runtime image location from the **.treeinfo** file. If the **.treeinfo** file is not available, the installation program attempts to load the image from **images/install.img**.

When you do not specify the **inst.stage2** option, the installation program attempts to use the location specified with the **inst.repo** option.

Use this option when you want to manually specify the installation source in the installation program at a later time. For example, when you want to select the Content Delivery Network (CDN) as an installation source. The installation DVD and Boot ISO already contain a suitable **inst.stage2** option to boot the installation program from the respective ISO.

If you want to specify an installation source, use the **inst.repo=** option instead.



## 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-x-0-0-BaseOS-x86\_64**. If you modify the default label of the file system that contains the runtime image, or if you use a customized procedure to boot the installation system, verify that the **inst.stage2=** boot option is set to the correct value.

### inst.noverifyssl

Use the **inst.noverifyssl** boot option to prevent the installer from verifying SSL certificates for all HTTPS connections with the exception of additional Kickstart repositories, where **--noverifyssl** can be set per repository.

For example, if your remote installation source is using self-signed SSL certificates, the **inst.noverifyssl** boot option enables the installer to complete the installation without verifying the SSL certificates.

#### Example when specifying the source using **inst.stage2=**

```
inst.stage2=https://hostname/path_to_install_image/ inst.noverifyssl
```

#### Example when specifying the source using **inst.repo=**

```
inst.repo=https://hostname/path_to_install_repository/ inst.noverifyssl
```

### inst.stage2.all

Use the **inst.stage2.all** boot option 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. For more information on how to update drivers during installation, see the [Performing an advanced RHEL 8 installation](#) document.

### inst.repo=hmc

This option eliminates the requirement of an external network setup and expands the installation options. 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 the **inst.repo=hmc** option to the kernel parameters. The installation program then enables support element (SE) and hardware management console (HMC) file access, fetches the images for stage2 from the DVD, and provides access to the packages on the DVD for software selection.

### inst.proxy=

The **inst.proxy=** boot option is used when performing an installation from a HTTP, HTTPS, and FTP protocol. For example:

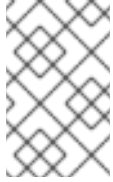
```
[PROTOCOL://][USERNAME[:PASSWORD]@]HOST[:PORT]
```



**inst.nosave=**

Use the **inst.nosave=** boot option to control the installation logs and related files that are not saved to the installed system, for example **input\_ks**, **output\_ks**, **all\_ks**, **logs** and **all**. You can combine multiple values separated by a comma. For example,

```
inst.nosave=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.

**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**.

**inst.memcheck**

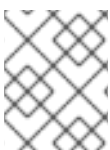
The **inst.memcheck** boot option performs a check to verify that the system has enough RAM to complete the installation. If there isn't enough RAM, the installation process is stopped. The system check is approximate and memory usage during installation depends on the package selection, user interface, for example graphical or text, and other parameters.

**inst.nomemcheck**

The **inst.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.

**9.4.4. Network boot options**

If your scenario requires booting from an image over the network instead of booting from a local image, you can use the following options to customize network booting.

**NOTE**

Initialize the network with the **dracut** tool. For complete list of **dracut** options, see the **dracut.cmdline(7)** man page.

**ip=**

Use the **ip=** boot option to configure one or more network interfaces. To configure multiple interfaces, use one of the following methods;

- use the **ip** option multiple times, once for each interface; to do so, use the **rd.neednet=1** option, and specify a primary boot interface using the **bootdev** option.
- 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.

In the following tables:

- The **ip** parameter specifies the client IP address and **IPv6** requires square brackets, for example 192.0.2.1 or [2001:db8::99].
- The **gateway** parameter is the default gateway. **IPv6** requires square brackets.
- 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 9.3. Boot option formats to configure the network interface**

| Boot option format                                          | Configuration method                                                                                                                                                                                                   |
|-------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>ip=method</b>                                            | Automatic configuration of any interface                                                                                                                                                                               |
| <b>ip=interface:method</b>                                  | Automatic configuration of a specific interface                                                                                                                                                                        |
| <b>ip=ip::gateway:netmask:hostname:interface:none</b>       | Static configuration, for example, IPv4:<br><b>ip=192.0.2.1::192.0.2.254:255.255.255.0:server.example.com:enp1s0:none</b><br><br>IPv6: <b>ip=[2001:db8::1]::<br/>[2001:db8::ffe]:64:server.example.com:enp1s0:none</b> |
| <b>ip=ip::gateway:netmask:hostname:interface:method:mtu</b> | Automatic configuration of a specific interface with an override                                                                                                                                                       |

#### Configuration methods for the automatic interface

The method **automatic configuration of a specific interface with an override** opens 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:

DHCP

**dhcp**

IPv6 DHCP

**dhcp6**

## IPv6 automatic configuration

**auto6**

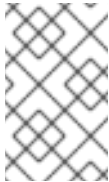
## iSCSI Boot Firmware Table (iBFT)

**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 default value of the **ip** option is **ip=dhcp**.
- To connect to an iSCSI target automatically, activate a network device for accessing the target by using 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**.

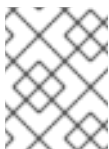
**bootdev=**

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.

### vlan=

Use the **vlan=** option to configure a Virtual LAN (VLAN) device on a specified interface with a given name. The syntax is **vlan=name:interface**. For example:

```
vlan=vlan5:enp0s1
```

This configures a VLAN device named **vlan5** on the **enp0s1** interface. The name can take the following forms:

- VLAN\_PLUS\_VID: **vlan0005**
- VLAN\_PLUS\_VID\_NO\_PAD: **vlan5**
- DEV\_PLUS\_VID: **enp0s1.0005**
- DEV\_PLUS\_VID\_NO\_PAD: **enp0s1.5**

### bond=

Use the **bond=** option to configure a bonding device with the following syntax: **bond=name[:interfaces][:options]**. Replace *name* with the bonding device name, *interfaces* with a comma-separated list of physical (Ethernet) interfaces, and *options* with a comma-separated list of bonding options. For example:

```
bond=bond0:enp0s1,enp0s2:mode=active-backup,tx_queues=32,downdelay=5000
```

For a list of available options, execute the **modinfo** bonding command.

### team=

Use the **team=** option to configure a team device with the following syntax: **team=name:interfaces**. Replace *name* with the desired name of the team device and *interfaces* with a comma-separated list of physical (Ethernet) devices to be used as underlying interfaces in the team device. For example:

```
team=team0:enp0s1,enp0s2
```

### bridge=

Use the **bridge=** option to configure a bridge device with the following syntax: **bridge=name:interfaces**. Replace *name* with the desired name of the bridge device and *interfaces* with a comma-separated list of physical (Ethernet) devices to be used as underlying interfaces in the bridge device. For example:

```
bridge=bridge0:enp0s1,enp0s2
```

## Additional resources

- [Configuring and managing networking](#)

## 9.4.5. Console boot options

This section describes how to configure 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**. When using the **console=** argument, the installation starts with a text UI. If you must use the **console=** option multiple times, the boot message is displayed on all specified console. However, the installation program uses only the last specified console. 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. To view the list of locales, enter the command **locale -a | grep \_** or the **localectl list-locales | grep \_** command.

**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:

- Disable geolocation: **inst.geoloc=0**
- Use the Fedora GeolP API: **inst.geoloc=provider\_fedora\_geoip**
- Use the Hostip.info GeolP API: **inst.geoloc=provider\_hostip**

If you do not specify the **inst.geoloc=** option, the default option is **provider\_fedora\_geoip**.

**inst.keymap=**

Use the **inst.keymap=** option to specify the keyboard layout 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. The graphical 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** you can use the **inst.noninteractive** option with a graphical or text installation. When you use the **inst.noninteractive** option 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 Virtual Network Computing (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. You can use this option by entering the command **vncviewer -listen**.

### inst.xdriver=

Use the **inst.xdriver=** option to specify the name of the X driver 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 the **inst.xdriver=fbdev** option.

### 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. 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 64-bit 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.

## 9.4.6. Debug boot options

This section describes the options you can use when debugging issues.

**inst.rescue**

Use the **inst.rescue** option to run the rescue environment for diagnosing and fixing systems. For example, you can [repair a filesystem in rescue mode](#).

**inst.updates=**

Use the **inst.updates=** option to specify the location of the **updates.img** file that you want to apply during installation. The **updates.img** file can be derived from one of several sources.

Table 9.4. **updates.img** file sources

| Source                            | Description                                                                                                                                                                                                                                      | Example                                                                                                                                                      |
|-----------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Updates from a network            | Specify the network location of <b>updates.img</b> . This does not require any modification to the installation tree. To use this method, edit the kernel command line to include <b>inst.updates</b> .                                          | <b>inst.updates=http://website.com/path/to/updates.img.</b>                                                                                                  |
| Updates from a disk image         | Save an <b>updates.img</b> on a floppy drive or a USB key. This can be done only with an <b>ext2</b> filesystem type of <b>updates.img</b> . To save the contents of the image on your floppy drive, insert the floppy disc and run the command. | <b>dd if=updates.img of=/dev/fd0 bs=72k count=20.</b> To use a USB key or flash media, replace <b>/dev/fd0</b> with the device name of your USB flash drive. |
| Updates from an installation tree | If you are using a CD, hard drive, HTTP, or FTP install, save the <b>updates.img</b> in the installation tree so that all installations can detect the <b>.img</b> file. The file name must be <b>updates.img</b> .                              | For NFS installs, save the file in the <b>images/</b> directory, or in the <b>RHupdates/</b> directory.                                                      |

**inst.loglevel=**

Use the **inst.loglevel=** option to specify the minimum level of messages logged on a terminal. This option applies only to 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**
- **critical**

The default value is **info**, which means that by default, the logging terminal displays messages ranging from **info** to **critical**.

**inst.syslog=**

Sends log messages to the **syslog** process on the specified host when the installation starts. You can use **inst.syslog=** only if the remote **syslog** process is configured to accept incoming connections.

**inst.virtio-log=**

Use the **inst.virtio-log=** option to specify which virtio port (a character device at `/dev/virtio-ports/name`) to use for forwarding logs. The default value is **org.fedoraproject.anaconda.log.0**.

**inst.zram=**

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 using the hard drive. This setup allows the installation program to run with less available memory and improve installation speed. You can configure the **inst.zram=** option using the following values:

- **inst.zram=1** to enable zRAM swap, regardless of system memory size. By default, swap on zRAM is enabled on systems with 2 GiB or less RAM.
- **inst.zram=0** to disable zRAM swap, regardless of system memory size. By default, swap on zRAM is disabled on systems with more than 2 GiB of memory.

**rd.live-ram**

Copies the **stage 2** image in **images/install.img** into RAM. Note that this increases the memory required for installation by the size of the image which is usually between 400 and 800MB.

**inst.nokill**

Prevent the installation program from rebooting when a fatal error occurs, or at the end of the installation process. Use it capture installation logs which would be lost upon reboot.

**inst.noshell**

Prevent a shell on terminal session 2 (tty2) during installation.

**inst.notmux**

Prevent the use of tmux during installation. The output is generated without terminal control characters and is meant for non-interactive uses.

**inst.remotelog=**

Sends all 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.

## 9.4.7. Storage boot options

This section describes the options you can specify to customize booting from a storage device.

**inst.nodmraid**

Disables **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 such as, **dmraid** or **wipefs**.



**inst.nompath**

Disables support for multipath devices. Use this option only if your system has a false-positive that incorrectly identifies a normal block device as a multipath device.

**WARNING**

Use this option with caution. Do not use this option with multipath hardware. Using this option to install to a single path of a multipath device is not supported.

**inst.gpt**

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  $2^{32}$  sectors in size or larger. Disk sectors are typically 512 bytes in size, meaning that this is usually equivalent to 2 TiB. The **inst.gpt** boot option allows a GPT to be written to smaller disks.

**9.4.8. Kickstart boot options**

This section describes the boot options you can add in the Kickstart file to automate an installation.

**inst.ks=**

Defines the location of a Kickstart file to use to automate the installation. You can 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 specified device.

If you use this option without specifying a device, the installation program uses the following value for the 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 9.5. 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 the **inst.ks.all** 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

- [Full list of boot options](#)

### 9.4.9. Advanced installation boot options

This section contains information about advanced installation boot options.

#### inst.kexec

Runs 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. This can potentially create issues for certain device drivers.

#### inst.multilib

Configures the system for multilib packages 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 that depends on the **glibc** package, the **bash** package is installed in multiple variants, while the **glibc** package is installed only in variants that the bash package requires.

**selinux=0**

Disables the use of SELinux in the installation program and the installed system. By default, SELinux operates in permissive mode in the installation program, and in enforcing mode in the installed system.

**NOTE**

The `inst.selinux=0` and `selinux=0` options are not the same: \* `inst.selinux=0`: disable SELinux only in the installation program. \* `selinux=0`: disable the use of SELinux in the installation program and the installed system. Disabling SELinux causes events not to be logged.

**inst.nonibftiscsiboot**

Places the boot loader on iSCSI devices that were not configured in the iSCSI Boot Firmware Table (iBFT).

**9.4.10. 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**.

**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 9.6. Values for the ksdevice boot option

| Value                       | Information                                                        |
|-----------------------------|--------------------------------------------------------------------|
| Not present                 | N/A                                                                |
| <b>ksdevice=link</b>        | Ignored as this option is the same as the default behavior         |
| <b>ksdevice=bootif</b>      | Ignored as this option is the default if <b>BOOTIF=</b> is present |
| <b>ksdevice=ibft</b>        | Replaced with <b>ip=ibft</b> . See <b>ip</b> for details           |
| <b>ksdevice=&lt;MAC&gt;</b> | Replaced with <b>BOOTIF=\${MAC}/-}</b>                             |

| Value                       | Information                  |
|-----------------------------|------------------------------|
| <b>ksdevice=&lt;DEV&gt;</b> | Replaced with <b>bootdev</b> |

### 9.4.11. 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. Use **inst.repo** or specify the appropriate network options.

#### blacklist, nofirewire

The **modprobe** option now 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.

#### repo=nfsiso

Use the **inst.repo=nfs:** option.

#### 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 because many options are 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 because 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 by using **ipv6.disable=1**. This setting is used by the installed system.

**upgradeany**

This option was no longer required because the installation program no longer handles upgrades.

## CHAPTER 10. KICKSTART REFERENCES

# APPENDIX I. KICKSTART SCRIPT FILE FORMAT REFERENCE

This reference describes in detail the kickstart file format.

## I.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.



## NOTE

If user interaction is needed during kickstart installation in text or graphical mode, enter only the windows where updates are mandatory to complete the installation. Entering spokes might lead to resetting the kickstart configuration. Resetting of the configuration applies specifically to the kickstart commands related to storage after entering the Installation Destination window.

## I.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.

### I.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/repodata/\*-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**

### I.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.

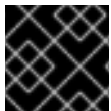
In Red Hat Enterprise Linux 8, modules are present only in the AppStream repository. To list available modules, use the **yum module list** command on an installed Red Hat Enterprise Linux 8 system.

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

- [Installing software](#)
- [Installing, managing, and removing user-space components](#)

### 1.2.3. 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.

User can configure Anaconda to install packages in **multilib** mode during the installation of the system. Use one of the following options to enable **multilib** mode:

1. Configure Kickstart file with the following lines:

```
%packages --multilib --default
%end
```

2. Add the `inst.multilib` boot option during booting the installation image.

### **--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.

## 1.2.4. 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.

**IMPORTANT**

The **--nodefaults** and **--optional** options cannot be used together. You can install only mandatory packages during the installation using **--nodefaults** and install the optional packages on the installed system post installation.

## 1.3. 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

### I.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.



#### IMPORTANT

The **%pre** section of Kickstart is executed at the stage of installation which happens after the installer image (**inst.stage2**) is fetched: it means **after** root switches to the installer environment (the installer image) and **after** the **Anaconda** installer itself starts. Then the configuration in **%pre** is applied and can be used to fetch packages from installation repositories configured, for example, by URL in Kickstart. However, it **cannot** be used to configure network to fetch the image (**inst.stage2**) from network.

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.

#### I.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, see [Introduction to Python](#) in *Configuring basic system settings*.

##### **--erroronfail**

Displays an error and halts the installation if the script fails. The error message will direct you to where the cause of the failure is logged. The installed system might get into an unstable and unbootable state. You can use the **inst.nokill** option to debug the script.

### **--log=**

Logs the script's output into the specified log file. For example:

```
%pre --log=/tmp/ks-pre.log
```

## **1.3.2. %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/sysroot`
- 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.

### **1.3.2.1. %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, see [Introduction to Python](#) in *Configuring basic system settings*.

### **--erroronfail**

Displays an error and halts the installation if the script fails. The error message will direct you to where the cause of the failure is logged. The installed system might get into an unstable and unbootable state. You can use the **inst.nokill** option to debug the script.

### **--log=**

Logs the script's output into the specified log file. For example:

```
%pre-install --log=/mnt/sysroot/root/ks-pre.log
```

## **1.3.3. %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.

Note that during execution of the **%post** section, the installation media must be still inserted.

### **1.3.3.1. %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, 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/sysroot/etc/resolv.conf
%end
```

### **--erroronfail**

Displays an error and halts the installation if the script fails. The error message will direct you to where the cause of the failure is logged. The installed system might get into an unstable and unbootable state. You can use the **inst.nokill** option to debug the script.

### **--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/sysroot/root/ks-post.log
```

### **1.3.3.2. 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
```

### **1.3.3.3. 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.

## I.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 I.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.

## I.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**

Displays an error and halts the installation if the script fails. The error message will direct you to where the cause of the failure is logged. The installed system might get into an unstable and unbootable state. You can use the **inst.nokill** option to debug the script.

### **--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, see [Introduction to Python](#) in *Configuring basic system settings*.

### **--log=**

Logs the script's output into the specified log file.

## **I.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 J. 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.

### J.1. KICKSTART CHANGES

The following sections describe the changes in Kickstart commands and options in Red Hat Enterprise Linux 8.

#### **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.

#### **Kickstart no longer supports Btrfs**

The Btrfs file system is not supported from Red Hat Enterprise Linux 8. As a result, the Graphical User Interface (GUI) and the Kickstart commands no longer support Btrfs.

#### **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.

#### **J.1.1. 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**
- **syspurpose** - use **subscription-manager syspurpose** instead

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.

### J.1.2. 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.

- **device**
- **deviceprobe**
- **dmraid**
- **install** - use the subcommands or methods directly as commands
- **multipath**
- **bootloader --upgrade**
- **ignoredisk --interactive**
- **partition --active**
- **harddrive --biospart**
- **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.

## J.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.

### J.2.1. **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.

### J.2.2. **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 64-bit IBM Z systems with the x3270 terminal.

### J.2.3. **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:

1. Load the driver disk on a hard disk drive, a USB or any similar device.
2. Set the label, for example, `DD`, to this device.
3. Add the following line to your Kickstart file:

```
driverdisk LABEL=DD:/e1000.rpm
```

Replace `DD` with a specific label and replace `e1000.rpm` with a specific name. Use anything supported by the `inst.repo` command instead of `LABEL` to specify your hard disk drive.

### J.2.4. 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.

## Syntax

```
eula [--agreed]
```

## Options

- `--agreed` (required) - Accept the EULA. This option must always be used, otherwise the `eula` command is meaningless.

### J.2.5. 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 root password, time & date, and networking & host name configuration options in addition to the default ones.

### J.2.6. 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.

### J.2.7. 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.

## J.2.8. `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 be formatted with a file system the installation program can mount: **`ext2`**, **`ext3`**, **`ext4`**, **`vfat`**, or **`xfs`**.

### Syntax

```
harddrive OPTIONS
```

### Options

- **`--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.

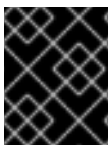
### Example

```
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.

## J.2.9. `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.

## J.2.10. 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.

- To actually run the installation, one of **cdrom**, **harddrive**, **hmc**, **nfs**, **liveimg**, or **url** must be specified.
- 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**.

### J.2.11. 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**.

### J.2.12. 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.

### J.2.13. 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.

## J.2.14. 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/>

## J.2.15. 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.

## J.2.16. 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 64-bit 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.

### J.2.17. 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. Option can be used multiple times, once per activation key, as long as the activation keys used are registered to your subscription.
- **--connect-to-insights** - Connects the target system to Red Hat Insights.
- **--proxy=** - Sets the HTTP proxy.

### J.2.18. 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.

## J.2.19. 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
```

## J.2.20. 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.

## J.2.21. url

The **url** Kickstart command is optional. It is used to install from an installation tree image on a remote server using the FTP, HTTP, or HTTPS protocol. You can only specify one URL.

## Syntax

```
url --url=FROM [OPTIONS]
```

## Mandatory options

- **--url=*FROM*** - Specifies the **HTTP**, **HTTPS**, **FTP**, or **file** location to install from.

## Optional options

- **--mirrorlist=** - Specifies the mirror URL to install from.
- **--proxy=** - Specifies an **HTTP**, **HTTPS**, or **FTP** proxy to use during the installation.
- **--noverifyssl** - Disables SSL verification when connecting to an **HTTPS** server.
- **--metalink=*URL*** - Specifies 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.

### J.2.22. 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

- [Preparing to install from the network using PXE](#)

### J.2.23. %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
```

### J.2.24. %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
```

## J.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.

### J.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.

### J.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.

### J.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.

### J.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.

### J.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
```

### J.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
```

### J.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

- [Installing, managing, and removing user-space components](#)

### J.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.
- **--excludepkgs=** - 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.
- **--includepkgs=** - 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.

### J.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.

### J.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

- [Using SELinux](#)

### J.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
```

### J.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.

### J.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 "ssh_key"
```



## Mandatory options

- **--username=** - The user for which the key will be installed.
- `ssh_key` - The complete SSH key fingerprint. It must be wrapped with quotes.

## J.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.



### NOTE

Red Hat Enterprise Linux 8.6 and later enables you to manage and display system purpose attributes with a single module by making the **role**, **service-level**, **usage**, and **addons** subcommands available under one **subscription-manager syspurpose** module. Previously, system administrators used one of four standalone **syspurpose** commands to manage each attribute. This standalone **syspurpose** command is deprecated starting with RHEL 8.6 and is planned to be removed in RHEL 9. Red Hat will provide bug fixes and support for this feature during the current release lifecycle, but this feature will no longer receive enhancements. Starting with RHEL 9, the single **subscription-manager syspurpose** command and its associated subcommands is the only way to use system purpose.

## 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.



## NOTE

Red Hat Enterprise Linux 8.6 and later enables you to manage and display system purpose attributes with a single module by making the **role**, **service-level**, **usage**, and **addons** subcommands available under one **subscription-manager syspurpose** module. Previously, system administrators used one of four standalone **syspurpose** commands to manage each attribute. This standalone **syspurpose** command is deprecated starting with RHEL 8.6 and is planned to be removed in RHEL 9. Red Hat will provide bug fixes and support for this feature during the current release lifecycle, but this feature will no longer receive enhancements. Starting with RHEL 9, the single **subscription-manager syspurpose** command and its associated subcommands is the only way to use system purpose.

### J.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.

### J.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.

### J.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.

## J.4. KICKSTART COMMANDS FOR NETWORK CONFIGURATION

The Kickstart commands in this list let you configure networking on the system.

### J.4.1. network (optional)

Use the optional **network** Kickstart command to configure network information for the target system and activate the network devices in the installation environment. The device specified in the first **network** command is activated automatically. You can also explicitly require a device to be activated using 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

- **--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 **inst.ks.device=** 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. Verify 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=** - Used to configure the target system's host name. The host name can either be a fully qualified domain name (FQDN) in the format **hostname.domainname**, or a short host name without the domain. 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. When using static IP and host name configuration, it depends on the planned system use case whether to use a short name or FQDN. Red Hat Identity Management configures FQDN during provisioning but some 3rd party software products may require short name. In either case, to ensure availability of both forms in all situations, add an entry for the host in **/etc/hosts** in the format **IP FQDN short-alias**.

The value **localhost** means that no specific static host name for the target system is

The value **localhost** 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.

Host names can only contain alphanumeric characters and - or .. Host name should be equal to or less than 64 characters. Host names cannot start or end with - and .. To be compliant with DNS, each part of a FQDN should be equal to or less than 63 characters and the FQDN total length, including dots, should not exceed 255 characters.

If you only want to configure the target system's host name, use the **--hostname** option in the **network** command and do not include any other option.

If you provide additional options when configuring the host name, the **network** command configures a device using the options specified. If you do not specify which device to configure using the **--device** option, the default **--device link** value is used. Additionally, if you do not specify the protocol using the **--bootproto** option, the device is configured to use DHCP by default.

- **--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 secondary devices 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 secondary devices.

- **--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**. For the list of available and supported modes, see [Configuring and Managing Networking Guide](#).

- **--vlanid=** - Specifies virtual LAN (VLAN) ID number (802.1q tag) for the device created using

- **--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 **vlanID** - for example, **vlan171**.
- **--teamslaves=** - Team device specified by the **--device=** option will be created using secondary devices specified in this option. Secondary devices are separated by commas. A secondary device 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 secondary devices 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 [Network Configuration Setting Specification](#). Also see the [Configuring and managing networking](#) document for general information about network bridging.
- **--bindto=mac** - Bind the device configuration 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 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 8 installation* document.

### J.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.

## J.5. KICKSTART COMMANDS FOR HANDLING STORAGE

The Kickstart commands in this section configure aspects of storage such as devices, disks, partitions, LVM, and filesystems.

### J.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"
```

## J.5.2. autopart

The **autopart** Kickstart command is optional. It automatically creates partitions.

The automatically created partitions are: a root (*/*) partition (1 GiB or larger), a **swap** partition, and an appropriate **/boot** partition for the architecture. On large enough drives (50 GiB 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.
- **--backuppssphrase** - 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 **--backuppasphrase** options.
- Ensure that the disk sector sizes are consistent when using **autopart**, **autopart --type=lvm**, or **autopart=thinp**.

### J.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**:

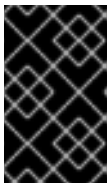
```
%packages
-plymouth
%end
```

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**

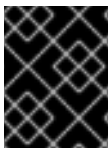
Ensure your system is not at risk of attack before disabling any of the vulnerability mitigation mechanisms. See the [Red Hat vulnerability response article](#) for information about the Meltdown and Spectre vulnerabilities.

- **--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 the device using its disk/by-id/dm-uuid-mpath-WWID name.

**IMPORTANT**

The **--boot-drive=** option is currently being ignored in Red Hat Enterprise Linux installations on 64-bit 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.

**IMPORTANT**

This option is applicable for Power systems only and UEFI systems should not use this option.

- **--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 8 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=xfst --onpart=sda1
```

You can use an entry similar to one of the following:

```
part / --fstype=xfst --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
```

```
part / --fstype=xfst --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.

### J.5.4. zipl

The **zipl** Kickstart command is optional. It specifies the ZIPL configuration for 64-bit IBM Z.

#### Options

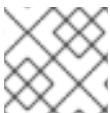
- **--secure-boot** - Enables secure boot if it is supported by the installing system.



#### 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.

### J.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 blocklisting 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. Disks cleared by **clearpart** will have the label created even in case the **--initlabel** is not used.

```
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=xfstyp --onpart=sda1
```

You could use an entry similar to one of the following:

```
part / --fstype=xfstyp --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
```

```
part / --fstype=xfstyp --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.

### J.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.

### J.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==/dev/disk/by-id/dm-uuid-mpath-
```

```
bootloader --location=mbr
```

You must specify only one of the **--drives** or **--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
```

- 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=xfst --onpart=sda1
```

You can use an entry similar to one of the following:

```
part / --fstype=xfst --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
```

```
part / --fstype=xfst --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.

## J.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**.

## J.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.

## J.5.10. logvol

The **logvol** Kickstart command is optional. It creates a logical volume for Logical Volume Management (LVM).

### Syntax

```
logvol mountpoint --vgname=name --name=name [OPTIONS]
```

### Mandatory options

#### **mountpoint**

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 [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**

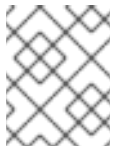
Use an existing logical volume and reformat it.

**--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.

**NOTE**

In the EFI system partition (**/boot/efi**), anaconda hard codes the value and ignores the users specified **--fsoptions** values.

**--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 [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.

### **--backuppssphrase**

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***

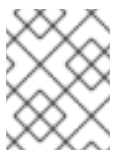
Specify the configuration profile name to use with thin logical volumes. If used, the name will also be included in the metadata for the given logical volume. By default, the available profiles are **default** and **thin-performance** and are defined in the `/etc/lvm/profile/` directory. See the **lvm(8)** 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 **--backupp passphrase** 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

- [Configuring and managing logical volumes](#)

## J.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.

## J.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] [--sectorsize=SECTORSIZE]
```

- **--namespace=** - The device specification by namespace. For example:

```
nvdimm reconfigure --namespace=namespace0.0 --mode=sector --sectorsize=512
```

- **--mode=** - The mode specification. Currently, only the value **sector** is available.

- **--sectorsize=** - Size of a sector for sector mode. For example:

```
nvdimm reconfigure --namespace=namespace0.0 --mode=sector --sectorsize=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]--blockdevs=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.

## J.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 E.4, “Recommended partitioning scheme”](#) for AMD64, Intel 64, and 64-bit ARM systems.

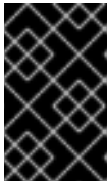
- **raid.id**  
The partition is used for software RAID (see **raid**).
- **pv.id**  
The partition is used for LVM (see **logvol**).
- **biosboot**

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 E.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 GiB of physical memory, the imposed limit is twice the amount of physical memory. For systems with more than 2 GiB, the imposed limit is the size of physical memory plus 2GiB.

- **--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
```



### 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 **part** 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.



### NOTE

In the EFI system partition (**/boot/efi**), anaconda hard codes the value and ignores the users specified **--fsoptions** values.

- **--label=** - assign a label to an individual partition.
- **--recommended** - Determine the size of the partition automatically.  
For details about the recommended scheme, see [Section E.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 **--backupp passphrase** options.

## J.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** or **1.0** if you intend to put the **/boot** and PReP partitions on the RAID device. The **mdadm** metadata versions **1.1** and **1.2** are not supported for the **/boot** and PReP partitions.
- The **PReP** Boot partitions are not required on PowerNV systems.
- **--level=** - RAID level to use (0, 1, 4, 5, 6, or 10).  
See [Section E.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=LABEL=root** or **--device=UUID=93348e56-4631-d0f0-6f5b-45c47f570b88**.

You can use the UUID of the file system on the RAID device or UUID of the RAID device itself. The UUID of the RAID device should be in the **8-4-4-4-12** format. UUID reported by mdadm is in the **8:8:8:8** format which needs to be changed. For example **93348e56:4631d0f0:6f5b45c4:7f570b88** should be changed to **93348e56-4631-d0f0-6f5b-45c47f570b88**.

- **--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.



#### NOTE

In the EFI system partition (**/boot/efi**), anaconda hard codes the value and ignores the users specified **--fsoptions** values.

- **--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.
- **--backupphrase** - 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 **--backupphrase** options.

## J.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.

## J.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.

## J.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.

## J.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 64-bit 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 64-bit 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.

### J.5.19. zfc

The **zfc** Kickstart command is optional. It defines a Fibre channel device.

This option only applies on 64-bit IBM Z. All of the options described below must be specified.

#### Syntax

```
zfc --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**.



#### NOTE

It is sufficient to specify an FCP device bus ID if automatic LUN scanning is available and when installing 8 or later releases. Otherwise all three parameters are required. Automatic LUN scanning is available for FCP devices operating in NPIV mode if it is not disabled through the **zfc.allow\_lun\_scan** module parameter (enabled by default). It provides access to all SCSI devices found in the storage area network attached to the FCP device with the specified bus ID.

#### Example

```
zfc --devnum=0.0.4000 --wwpn=0x5005076300C213e9 --fcplun=0x5022000000000000
zfc --devnum=0.0.4000
```

## J.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.

### J.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 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).

## J.6.2. %addon org\_fedora\_oscaps

The `%addon org_fedora_oscaps` 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 J.1. Sample OpenSCAP Add-on Definition Using SCAP Security Guide

```
%addon org_fedora_oscap
content-type = scap-security-guide
profile = xccdf_org.ssgproject.content_profile_pci-dss
%end
```

- The following is a more complex example which loads a custom profile from a web server:

#### Example J.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

- [Security Hardening](#)
- [OpenSCAP installation program add-on](#)
- [OpenSCAP Portal](#)

## J.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.

### J.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.
- **--notstrict** - 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.
- **--notempty** - 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.

## J.8. KICKSTART COMMANDS FOR SYSTEM RECOVERY

The Kickstart command in this section repairs an installed system.

### J.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
- Blocklist 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. nvdim
  - e. zfc
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 11. 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.

### 11.1. 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.

### 11.2. 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.

### 11.3. 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.

## 11.4. SECURITY CONTROLS

Computer security is often divided into three distinct main 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.

### 11.4.1. 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)

### 11.4.2. 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

### 11.4.3. 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

## 11.5. 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.

### 11.5.1. 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.

### 11.5.2. 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

### 11.5.3. 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.

## 11.6. SECURITY THREATS

### 11.6.1. 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.

## 11.6.2. 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.

### **11.6.3. Threats to workstation and home PC security**

Workstations and home PCs may not be as prone to attack as networks or servers, but because 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 "bot" 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.

## 11.7. COMMON EXPLOITS AND ATTACKS

The following table 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 11.1. 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).                         | <p>Commonly associated with networking hardware such as routers, firewalls, VPNs, and network attached storage (NAS) appliances.</p> <p>Common in many legacy operating systems, especially those that bundle services (such as UNIX and Windows.)</p> <p>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.</p> |
| 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, <b>all</b> 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   | 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>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 <b>rsh</b>, <b>telnet</b>, 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 <b>ssh</b> or SSL/TLS.</p>                                                         |
| Eavesdropping | Collecting data that passes between two active nodes on a network by eavesdropping on the connection between the two nodes.                                                                  | <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 | 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>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 <i>zombies</i>, 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 (RFC 2267) using the <b>nftables</b> packet-filtering framework and Network Intrusion Detection Systems such as <b>snort</b> assist administrators in tracking down and preventing distributed DoS attacks.</p> |

## CHAPTER 12. 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.

### 12.1. 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.

#### 12.1.1. BIOS passwords

The two primary reasons for password protecting the BIOS of a computer are<sup>[1]</sup>:

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.

#### 12.1.2. 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.

### 12.2. DISK PARTITIONING

Red Hat recommends creating separate partitions for the **/boot**, **/**, **/home**, **/tmp**, and **/var/tmp** directories.

### **/boot**

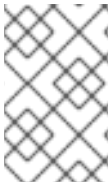
This partition is the first partition that is read by the system during boot up. The boot loader and kernel images that are used to boot your system into Red Hat Enterprise Linux 8 are stored in this partition. This partition should not be encrypted. If this partition is included in / and that partition is encrypted or otherwise becomes unavailable then your system is not 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 is 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.

## **12.3. 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.

## **12.4. 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.

## **12.5. POST-INSTALLATION PROCEDURES**

The following steps are the security-related procedures that should be performed immediately after installation of Red Hat Enterprise Linux 8.

- Update your system. Enter the following command as root:



```
# yum update
```

- 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
```

- 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] Because 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 13. USING SYSTEM-WIDE CRYPTOGRAPHIC POLICIES

The system-wide cryptographic 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.

### 13.1. SYSTEM-WIDE CRYPTOGRAPHIC POLICIES

When 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.

RHEL 8 contains the following predefined policies:

|                |                                                                                                                                                                                                                                                                                                                                                                                                                     |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DEFAULT</b> | 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.                                                                                                                                 |
| <b>LEGACY</b>  | 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 <b>DEFAULT</b> 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. |
| <b>FUTURE</b>  | 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. 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 3072 bits long.                                                                        |
| <b>FIPS</b>    | A policy level that conforms with the FIPS 140-2 requirements. This is used internally by the <b>fips-mode-setup</b> tool, which switches the RHEL system into FIPS mode.                                                                                                                                                                                                                                           |

Red Hat continuously adjusts all policy levels so that all libraries, except when using the LEGACY policy, provide secure defaults. Even though the LEGACY profile does not provide secure defaults, it does not include any algorithms that are easily exploitable. As such, the set of enabled algorithms or acceptable key sizes in any provided policy may change during the lifetime of Red Hat Enterprise Linux.

Such changes reflect new security standards and new security research. If you must ensure interoperability with a specific system for the whole lifetime of Red Hat Enterprise Linux, you should opt-out from cryptographic-policies for components that interact with that system or re-enable specific algorithms using custom policies.



## 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 Red Hat Enterprise Linux 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 <sup>[a]</sup> |
| Symmetric ciphers with keys < 256 bits       | yes           | yes           | yes           | no                |
| SHA-1 and SHA-224 signatures in certificates | yes           | yes           | yes           | no                |

LEGACY

DEFAULT

FIPS

FUTURE

[a] CBC ciphers are disabled for TLS. In a non-TLS scenario, **AES-128-CBC** is disabled but **AES-256-CBC** is enabled. To disable also **AES-256-CBC**, apply a custom subpolicy.

### Additional resources

- **update-crypto-policies(8)** man page

## 13.2. 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 6 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.
- For defining custom cryptographic policies, see the **Custom Policies** section in the **update-crypto-policies(8)** man page and the **Crypto Policy Definition Format** section in the **crypto-policies(7)** man page.

## 13.3. SETTING UP SYSTEM-WIDE CRYPTOGRAPHIC POLICIES IN THE WEB CONSOLE

You can choose from predefined system-wide cryptographic policy levels and switch between them directly in the Red Hat Enterprise Linux web console interface. If you set a custom policy on your system, the web console displays the policy in the **Overview** page as well as the **Change crypto policy** dialog window.

## Prerequisites

- The RHEL 8 web console has been installed. For details, see [Installing and enabling the web console](#).
- You have administrator privileges.

## Procedure

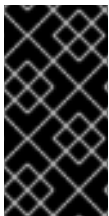
1. Log in to the RHEL web console. For more information, see [Logging in to the web console](#).
2. In the **Configuration** card of the **Overview** page, click your current policy value next to **Crypto policy**.
3. In the **Change crypto policy** dialog window, click on the policy level that you want to start using.
4. Click the **Apply and reboot** button.

## Verification

- Log back in and check that the **Crypto policy** value corresponds to the one you selected.

## 13.4. 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 the 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.



### IMPORTANT

Red Hat recommends installing Red Hat Enterprise Linux 8 with FIPS mode enabled, as opposed to enabling FIPS mode later. Enabling FIPS mode during the installation ensures that the system generates all keys with FIPS-approved algorithms and continuous monitoring tests in place.

## Procedure

1. To switch the system to FIPS mode:

```
# fips-mode-setup --enable
Kernel initramdisks are being regenerated. This might take some time.
Setting system policy to FIPS
Note: System-wide crypto policies are applied on application start-up.
It is recommended to restart the system for the change of policies
to fully take place.
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

1. After the restart, you can check the current state of FIPS mode:

```
# fips-mode-setup --check
FIPS mode is enabled.
```

#### Additional resources

- [fips-mode-setup\(8\)](#) man page
- [Installing a RHEL 8 system with FIPS mode enabled](#)
- [List of RHEL applications using cryptography that is not compliant with FIPS 140-2](#)
- [Security Requirements for Cryptographic Modules](#) on the National Institute of Standards and Technology (NIST) web site.

## 13.5. ENABLING FIPS MODE IN A CONTAINER

To enable the full set of cryptographic module self-checks mandated by the Federal Information Processing Standard Publication 140-2 (FIPS mode), the host system kernel must be running in FIPS mode. Depending on the version of your host system, enabling FIPS mode on containers either is fully automatic or requires only one command.



### NOTE

The **fips-mode-setup** command does not work correctly in containers, and it cannot be used to enable or check FIPS mode in this scenario.

#### Prerequisites

- The host system must be in FIPS mode.

#### Procedure

- **On hosts running RHEL 8.1 and 8.2** Set the FIPS cryptographic policy level in the container using the following command, and ignore the advice to use the **fips-mode-setup** command:

```
$ update-crypto-policies --set FIPS
```

- **On hosts running RHEL 8.4 and later** On systems with FIPS mode enabled, the **podman** utility automatically enables FIPS mode on supported containers.

#### Additional resources

- [Switching the system to FIPS mode](#) .
- [Installing a RHEL 8 system with FIPS mode enabled](#)

## 13.6. LIST OF RHEL APPLICATIONS USING CRYPTOGRAPHY THAT IS NOT COMPLIANT WITH FIPS 140-2

Red Hat recommends to use libraries from the core crypto components set, as they are guaranteed to pass all relevant crypto certifications, such as FIPS 140-2, and also follow the RHEL system-wide crypto policies.

See the [RHEL 8 core crypto components](#) article for an overview of the RHEL 8 core crypto components, the information on 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.

In addition to the following table, in some RHEL 8 Z-stream releases (for example, 8.1.1), the Firefox browser packages have been updated, and they contain a separate copy of the NSS cryptography library. This way, Red Hat wants to avoid the disruption of rebasing such a low-level component in a patch release. As a result, these Firefox packages do not use a FIPS 140-2-validated module.

**Table 13.1. List of RHEL 8 applications using cryptography that is not compliant with FIPS 140-2**

| Application                                                                                                                                                  | Details                                                                                        |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------|
| FreeRADIUS                                                                                                                                                   | The RADIUS protocol uses MD5                                                                   |
| ghostscript                                                                                                                                                  | Custom cryptography implementation (MD5, RC4, SHA-2, AES) to encrypt and decrypt documents     |
| ipxe                                                                                                                                                         | Crypto stack for TLS is compiled in, however, it is unused                                     |
| libica                                                                                                                                                       | Software fallbacks for various algorithms such as RSA and ECDH through CPACF instructions      |
| Ovmf (UEFI firmware), Edk2, shim                                                                                                                             | Full crypto stack (an embedded copy of the OpenSSL library)                                    |
| perl-Digest-HMAC                                                                                                                                             | HMAC, HMAC-SHA1, HMAC-MD5                                                                      |
| perl-Digest-SHA                                                                                                                                              | SHA-1, SHA-224, ...                                                                            |
| pidgin                                                                                                                                                       | DES, RC4                                                                                       |
| qatengine                                                                                                                                                    | Mixed hardware and software implementation of cryptographic primitives (RSA, EC, DH, AES, ...) |
| samba <sup>[a]</sup>                                                                                                                                         | AES, DES, RC4                                                                                  |
| valgrind                                                                                                                                                     | AES, hashes <sup>[b]</sup>                                                                     |
| <p>[a] Starting with RHEL 8.3, samba uses FIPS-compliant cryptography.</p> <p>[b] Re-implements in software hardware-offload operations, such as AES-NI.</p> |                                                                                                |



## 13.7. 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.

### 13.7.1. 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 '@SECLEVEL=0: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.

To opt out of system-wide crypto policies for your OpenSSH client, perform one of the following tasks:

- For a given user, override the global **ssh\_config** with a user-specific configuration in the `~/.ssh/config` file.
- For the entire system, specify the crypto policy in a drop-in configuration file located in the `/etc/ssh/ssh_config.d/` directory, with a two-digit number prefix smaller than 50, so that it lexicographically precedes the **50-redhat.conf** file, and with a **.conf** suffix, for example, **49-crypto-policy-override.conf**.

See the **ssh\_config(5)** man page for more information.

## Libreswan

See the [Configuring IPsec connections that opt out of the system-wide crypto policies](#) in the [Securing networks](#) document for detailed information.

## Additional resources

- **update-crypto-policies(8)** man page

## 13.8. CUSTOMIZING SYSTEM-WIDE CRYPTOGRAPHIC POLICIES WITH SUBPOLICIES

Use this procedure to adjust the set of enabled cryptographic algorithms or protocols.

You can either apply custom subpolicies on top of an existing system-wide cryptographic policy or define such a policy from scratch.

The concept of scoped policies allows enabling different sets of algorithms for different back ends. You can limit each configuration directive to specific protocols, libraries, or services.

Furthermore, directives can use asterisks for specifying multiple values using wildcards.

The **/etc/crypto-policies/state/CURRENT.pol** file lists all settings in the currently applied system-wide cryptographic policy after wildcard expansion. To make your cryptographic policy more strict, consider using values listed in the **/usr/share/crypto-policies/policies/FUTURE.pol** file.

You can find example subpolicies in the **/usr/share/crypto-policies/policies/modules/** directory. The subpolicy files in this directory contain also descriptions in lines that are commented out.



### NOTE

Customization of system-wide cryptographic policies is available from RHEL 8.2. You can use the concept of scoped policies and the option of using wildcards in RHEL 8.5 and newer.

## Procedure

1. Checkout to the **/etc/crypto-policies/policies/modules/** directory:

```
# cd /etc/crypto-policies/policies/modules/
```

2. Create subpolicies for your adjustments, for example:

```
# touch MYCRYPTO-1.pmod  
# touch SCOPES-AND-WILDCARDS.pmod
```



### IMPORTANT

Use upper-case letters in file names of policy modules.

- Open the policy modules in a text editor of your choice and insert options that modify the system-wide cryptographic policy, for example:

```
# vi MYCRYPTO-1.pmod
```

```
min_rsa_size = 3072
hash = SHA2-384 SHA2-512 SHA3-384 SHA3-512
```

```
# vi SCOPES-AND-WILDCARDS.pmod
```

```
# Disable the AES-128 cipher, all modes
cipher = -AES-128-*
```

```
# Disable CHACHA20-POLY1305 for the TLS protocol (OpenSSL, GnuTLS, NSS, and
OpenJDK)
cipher@TLS = -CHACHA20-POLY1305
```

```
# Allow using the FFDHE-1024 group with the SSH protocol (libssh and OpenSSH)
group@SSH = FFDHE-1024+
```

```
# Disable all CBC mode ciphers for the SSH protocol (libssh and OpenSSH)
cipher@SSH = -*-CBC
```

```
# Allow the AES-256-CBC cipher in applications using libssh
cipher@libssh = AES-256-CBC+
```

- Save the changes in the module files.
- Apply your policy adjustments to the **DEFAULT** system-wide cryptographic policy level:

```
# update-crypto-policies --set DEFAULT:MYCRYPTO-1:SCOPES-AND-WILDCARDS
```

- To make your cryptographic settings effective for already running services and applications, restart the system:

```
# reboot
```

## Verification

- Check that the `/etc/crypto-policies/state/CURRENT.pol` file contains your changes, for example:

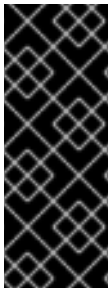
```
$ cat /etc/crypto-policies/state/CURRENT.pol | grep rsa_size
min_rsa_size = 3072
```

## Additional resources

- Custom Policies** section in the **update-crypto-policies(8)** man page
- Crypto Policy Definition Format** section in the **crypto-policies(7)** man page
- [How to customize crypto policies in RHEL 8.2](#) Red Hat blog article

## 13.9. DISABLING SHA-1 BY CUSTOMIZING A SYSTEM-WIDE CRYPTOGRAPHIC POLICY

Because the SHA-1 hash function has an inherently weak design, and advancing cryptanalysis has made it vulnerable to attacks, RHEL 8 does not use SHA-1 by default. Nevertheless, some third-party applications, for example, public signatures, still use SHA-1. To disable the use of SHA-1 in signature algorithms on your system, you can use the **NO-SHA1** policy module.



### IMPORTANT

The **NO-SHA1** policy module disables the SHA-1 hash function only in signatures and not elsewhere. In particular, the **NO-SHA1** module still allows the use of SHA-1 with hash-based message authentication codes (HMAC). This is because HMAC security properties do not rely on the collision resistance of the corresponding hash function, and therefore the recent attacks on SHA-1 have a significantly lower impact on the use of SHA-1 for HMAC.

If your scenario requires disabling a specific key exchange (KEX) algorithm combination, for example, **diffie-hellman-group-exchange-sha1**, but you still want to use both the relevant KEX and the algorithm in other combinations, see [Steps to disable the diffie-hellman-group1-sha1 algorithm in SSH](#) for instructions on opting out of system-wide crypto-policies for SSH and configuring SSH directly.



### NOTE

The module for disabling SHA-1 is available from RHEL 8.3. Customization of system-wide cryptographic policies is available from RHEL 8.2.

### Procedure

1. Apply your policy adjustments to the **DEFAULT** system-wide cryptographic policy level:

```
# update-crypto-policies --set DEFAULT:NO-SHA1
```

2. To make your cryptographic settings effective for already running services and applications, restart the system:

```
# reboot
```

### Additional resources

- **Custom Policies** section in the **update-crypto-policies(8)** man page.
- **Crypto Policy Definition Format** section in the **crypto-policies(7)** man page.
- [How to customize crypto policies in RHEL](#) Red Hat blog article.

## 13.10. 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/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**

- **Custom Policies** section in the **update-crypto-policies(8)** man page and the **Crypto Policy Definition Format** section in the **crypto-policies(7)** man page
- [How to customize crypto policies in RHEL](#) Red Hat blog article

**13.11. ADDITIONAL RESOURCES**

- [System-wide crypto policies in RHEL 8](#) and [Strong crypto defaults in RHEL 8 and deprecation of weak crypto algorithms](#) Knowledgebase articles

## CHAPTER 14. 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 RHEL, 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.

### 14.1. 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.

RHEL 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 the system. 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](#)

### 14.2. USING SSH KEYS STORED ON A SMART CARD

Red Hat Enterprise Linux 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)**, **opensc.conf(5)**, **pcscd(8)**, **ssh(1)**, and **ssh-keygen(1)** man pages

## 14.3. CONFIGURING APPLICATIONS TO AUTHENTICATE USING CERTIFICATES FROM SMART CARDS

Authentication using smart cards in applications may increase security and simplify automation.

- 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**, and **OpenSSL**, use **p11-kit** to implement registering PKCS #11 modules.

### Additional resources

- **p11-kit(8)** man page.

## 14.4. USING HSMS PROTECTING PRIVATE KEYS IN APACHE

The **Apache** HTTP server 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.

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 use 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 the **/usr/share/httpd/manual/mod/mod\_ssl.html** file.

## 14.5. USING HSMS PROTECTING PRIVATE KEYS IN NGINX

The **Nginx** HTTP server 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.

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.

## 14.6. ADDITIONAL RESOURCES

- **pkcs11.conf(5)** man page.

## CHAPTER 15. 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 block-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.

### 15.1. THE SYSTEM-WIDE TRUST STORE

In RHEL, 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. For example, trust anchors belong to the `/usr/share/pki/ca-trust-source/anchors/` or `/etc/pki/ca-trust/source/anchors/` directory.



#### NOTE

In a hierarchical cryptographic system, a trust anchor is an authoritative entity that other parties consider trustworthy. In the X.509 architecture, a root certificate is a trust anchor from which a chain of trust is derived. To enable chain validation, the trusting party must have access to the trust anchor first.

#### Additional resources

- `update-ca-trust(8)` and `trust(1)` man pages

### 15.2. ADDING NEW CERTIFICATES

To acknowledge applications on your system with a new source of trust, add the corresponding certificate to the system-wide store, and use the `update-ca-trust` command.

#### Prerequisites

- The `ca-certificates` package is present on the system.

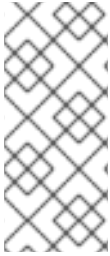
#### Procedure

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

Even though the Firefox browser can use an added certificate without a prior execution of **update-ca-trust**, enter the **update-ca-trust** command after every CA change. Also note that browsers, such as Firefox, Chromium, and GNOME Web cache files, and you might have to clear your browser's cache or restart your browser to load the current system certificate configuration.

### Additional resources

- **update-ca-trust(8)** and **trust(1)** man pages

## 15.3. MANAGING TRUSTED SYSTEM CERTIFICATES

The **trust** command provides a convenient way for managing certificates in the shared system-wide trust store.

- 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
...
```

- 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
...
--purpose=<usage> limit to certificates usable for the purpose
                 server-auth   for authenticating servers
...
```

### Additional resources

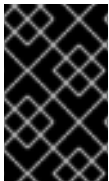
- **update-ca-trust(8)** and **trust(1)** man pages

# CHAPTER 16. SCANNING THE SYSTEM FOR SECURITY COMPLIANCE AND VULNERABILITIES

## 16.1. CONFIGURATION COMPLIANCE TOOLS IN RHEL

Red Hat Enterprise Linux provides tools that enable you to perform 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. You can also use it to generate security reports based on these scans and evaluations.
- **OpenSCAP** - The **OpenSCAP** library, with the accompanying **oscap** command-line utility, is designed to perform configuration and vulnerability scans on a local system, to validate configuration compliance content, and to generate reports and guides based on these scans and evaluations.



### IMPORTANT

You can experience memory-consumption problems while using **OpenSCAP**, which can cause stopping the program prematurely and prevent generating any result files. See the [OpenSCAP memory-consumption problems](#) Knowledgebase article for details.

- **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 enables administrators to write their security content using a scripting language, such as Bash, Python, and Ruby. The SCE extension is provided in the **openscap-engine-sce** package. The SCE itself is not part of the SCAP standard.

To perform automated compliance audits on multiple systems remotely, you can use the OpenSCAP solution for Red Hat Satellite.

### Additional resources

- **oscap(8)**, **scap-workbench(8)**, and **scap-security-guide(8)** man pages
- [Red Hat Security Demos: Creating Customized Security Policy Content to Automate Security Compliance](#)
- [Red Hat Security Demos: Defend Yourself with RHEL Security Technologies](#)
- [Security Compliance Management in the Administering Red Hat Satellite Guide](#) .

## 16.2. 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 although the implementation of the scanner or policy editor is not mandated. This enables organizations to build their security policy (SCAP content) once, no matter how many security vendors they employ.

The Open Vulnerability Assessment Language (OVAL) is the essential and oldest component of SCAP. Unlike other tools and custom scripts, OVAL describes a required state of resources in a declarative manner. OVAL code is never executed directly but using 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.

Because of differences between platforms, versions, and other factors, Red Hat Product Security qualitative severity ratings of vulnerabilities do not directly align with the Common Vulnerability Scoring System (CVSS) baseline ratings provided by third parties. Therefore, we recommend that you use the RHSA OVAL definitions instead of those provided by third parties.

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). Because 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 example, 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.



#### NOTE

The [Red Hat Insights for Red Hat Enterprise Linux compliance service](#) helps IT security and compliance administrators to assess, monitor, and report on the security policy compliance of Red Hat Enterprise Linux systems. You can also create and manage your SCAP security policies entirely within the compliance service UI.

#### Additional resources

- [Red Hat and OVAL compatibility](#)
- [Red Hat and CVE compatibility](#)
- [Notifications and Advisories](#) in the [Product Security Overview](#)
- [Security Data Metrics](#)

## 16.3. VULNERABILITY SCANNING

### 16.3.1. 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 although the implementation of the scanner or policy editor is not mandated. This enables organizations to build their security policy (SCAP content) once, no matter how many security vendors they employ.

The Open Vulnerability Assessment Language (OVAL) is the essential and oldest component of SCAP. Unlike other tools and custom scripts, OVAL describes a required state of resources in a declarative manner. OVAL code is never executed directly but using 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.

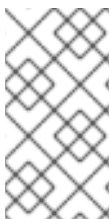
[Red Hat Product Security](#) helps customers evaluate and manage risk by tracking and investigating all security issues affecting Red Hat customers. It provides timely and concise patches and security advisories on the Red Hat Customer Portal. Red Hat creates and supports OVAL patch definitions, providing machine-readable versions of our security advisories.

Because of differences between platforms, versions, and other factors, Red Hat Product Security qualitative severity ratings of vulnerabilities do not directly align with the Common Vulnerability Scoring System (CVSS) baseline ratings provided by third parties. Therefore, we recommend that you use the RHSA OVAL definitions instead of those provided by third parties.

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). Because 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 example, 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.



#### NOTE

The [Red Hat Insights for Red Hat Enterprise Linux compliance service](#) helps IT security and compliance administrators to assess, monitor, and report on the security policy compliance of Red Hat Enterprise Linux systems. You can also create and manage your SCAP security policies entirely within the compliance service UI.

#### Additional resources

- [Red Hat and OVAL compatibility](#)
- [Red Hat and CVE compatibility](#)

- [Notifications and Advisories](#) in the [Product Security Overview](#)
- [Security Data Metrics](#)

### 16.3.2. Scanning the system for vulnerabilities

The **oscap** command-line utility enables you to scan local systems, validate configuration 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 **openscap-scanner** and **bzip2** packages are installed.

#### Procedure

1. Download the latest RHSA OVAL definitions for your system:

```
# wget -O - https://www.redhat.com/security/data/oval/v2/RHEL8/rhel-8.oval.xml.bz2 | bzip2 -  
-decompress > rhel-8.oval.xml
```

2. Scan the system for vulnerabilities and save results to the *vulnerability.html* file:

```
# oscap oval eval --report vulnerability.html rhel-8.oval.xml
```

#### Verification

- Check the results in a browser of your choice, for example:

```
$ firefox vulnerability.html &
```

#### Additional resources

- **oscap(8)** man page
- [Red Hat OVAL definitions](#)
- [OpenSCAP memory consumption problems](#)

### 16.3.3. Scanning remote systems for vulnerabilities

You can check also remote systems for vulnerabilities with the OpenSCAP scanner using the **oscap-ssh** tool over the SSH protocol.

#### Prerequisites

- The **openscap-utils** and **bzip2** packages are installed on the system you use for scanning.
- The **openscap-scanner** package is installed on the remote systems.
- The SSH server is running on the remote systems.



## Procedure

1. Download the latest RHSA OVAL definitions for your system:

```
# wget -O - https://www.redhat.com/security/data/oval/v2/RHEL8/rhel-8.oval.xml.bz2 | bzip2 -  
-decompress > rhel-8.oval.xml
```

2. 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 rhel-8.oval.xml
```

## Additional resources

- [oscap-ssh\(8\)](#)
- [Red Hat OVAL definitions](#)
- [OpenSCAP memory consumption problems](#)

## 16.4. CONFIGURATION COMPLIANCE SCANNING

### 16.4.1. Configuration compliance in RHEL

You can use configuration compliance scanning to conform to a baseline defined by a specific organization. For example, if you work with the US government, you might have to align your systems with the Operating System Protection Profile (OSPP), and if you are a payment processor, you might have to align your systems with the Payment Card Industry Data Security Standard (PCI-DSS). You can also perform configuration compliance scanning to harden your system security.

Red Hat recommends you follow the Security Content Automation Protocol (SCAP) content provided in the SCAP Security Guide package because it is in line with Red Hat best practices for affected components.

The SCAP Security Guide package provides content which conforms to the SCAP 1.2 and SCAP 1.3 standards. The **openscap scanner** utility is compatible with both SCAP 1.2 and SCAP 1.3 content provided in the SCAP Security Guide package.



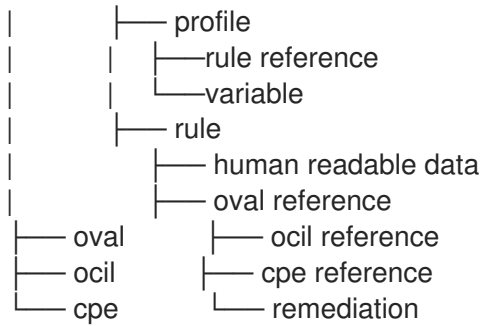
### IMPORTANT

Performing a configuration compliance scanning does not guarantee the system is compliant.

The SCAP Security Guide suite provides profiles for several platforms in a form of data stream documents. A data stream is a file that contains definitions, benchmarks, profiles, and individual rules. Each rule specifies the applicability and requirements for compliance. RHEL provides several profiles for compliance with security policies. In addition to the industry standard, Red Hat data streams also contain information for remediation of failed rules.

### Structure of compliance scanning resources

```
Data stream  
├── xccdf  
└── benchmark
```



A profile is a set of rules based on a security policy, such as OSPP, PCI-DSS, and Health Insurance Portability and Accountability Act (HIPAA). This enables you to audit the system in an automated way for compliance with security standards.

You can modify (tailor) a profile to customize certain rules, for example, password length. For more information on profile tailoring, see [Customizing a security profile with SCAP Workbench](#).

### 16.4.2. Possible results of an OpenSCAP scan

Depending on various properties of your system and the data stream and profile applied to an OpenSCAP scan, each rule may produce a specific result. This is a list of possible results with brief explanations of what they mean.

**Table 16.1. Possible results of an OpenSCAP scan**

| Result         | Explanation                                                                                                                                                                                              |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Pass           | The scan did not find any conflicts with this rule.                                                                                                                                                      |
| Fail           | The scan found a conflict with this rule.                                                                                                                                                                |
| Not checked    | OpenSCAP does not perform an automatic evaluation of this rule. Check whether your system conforms to this rule manually.                                                                                |
| Not applicable | This rule does not apply to the current configuration.                                                                                                                                                   |
| Not selected   | This rule is not part of the profile. OpenSCAP does not evaluate this rule and does not display these rules in the results.                                                                              |
| Error          | The scan encountered an error. For additional information, you can enter the <b>oscap</b> command with the <b>--verbose DEVEL</b> option. Consider opening a <a href="#">bug report</a> .                |
| Unknown        | The scan encountered an unexpected situation. For additional information, you can enter the <b>oscap</b> command with the <b>--verbose DEVEL</b> option. Consider opening a <a href="#">bug report</a> . |

### 16.4.3. Viewing profiles for configuration compliance

Before you decide to use profiles for scanning or remediation, you can list them and check their detailed descriptions using the **oscap info** subcommand.

## 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** subcommand. 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: Health Insurance Portability and Accountability Act (HIPAA)
  Id: xccdf_org.ssgproject.content_profile_hipaa
Title: PCI-DSS v3.2.1 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 HIPPA profile is: **xccdf\_org.ssgproject.content\_profile\_hipaa**, and the value for the **--profile** option is **hipaa**:

```
$ oscap info --profile hipaa /usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
...
Profile
Title: Health Insurance Portability and Accountability Act (HIPAA)

Description: The HIPAA Security Rule establishes U.S. national standards to protect
individuals' electronic personal health information that is created, received, used, or
maintained by a covered entity.
...
```

## Additional resources

- **scap-security-guide(8)** man page

- [OpenSCAP memory consumption problems](#)

#### 16.4.4. Assessing configuration compliance with a specific baseline

To determine whether your system conforms to a specific baseline, follow these steps.

##### Prerequisites

- The **openscap-scanner** and **scap-security-guide** packages are installed
- You know the ID of the profile within the baseline with which the system should comply. To find the ID, see [Viewing Profiles for Configuration Compliance](#).

##### Procedure

1. Evaluate the compliance of the system with the selected profile and save the scan results in the *report.html* HTML file, for example:

```
$ oscap xccdf eval --report report.html --profile hipaa /usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

2. Optional: Scan a remote system with the **machine1** host name, SSH running on port **22**, and the **joesec** user name for compliance and save results to the **remote-report.html** file:

```
$ oscap-ssh joesec@machine1 22 xccdf eval --report remote_report.html --profile hipaa /usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

##### Additional resources

- **scap-security-guide(8)** man page
- **SCAP Security Guide** documentation in the `/usr/share/doc/scap-security-guide/` directory
- `/usr/share/doc/scap-security-guide/guides/ssg-rhel8-guide-index.html` - [Guide to the Secure Configuration of Red Hat Enterprise Linux 8] installed with the **scap-security-guide-doc** package
- [OpenSCAP memory consumption problems](#)

## 16.5. REMEDIATING THE SYSTEM TO ALIGN WITH A SPECIFIC BASELINE

Use this procedure to remediate the RHEL system to align with a specific baseline. This example uses the Health Insurance Portability and Accountability Act (HIPAA) profile.



## WARNING

If not used carefully, running the system evaluation with the **Remediate** option enabled might render the system non-functional. 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 system.

### Procedure

1. Use the **oscaped** command with the **--remediate** option:

```
# oscaped xccdf eval --profile hipaa --remediate /usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

2. Restart your system.

### Verification

1. Evaluate compliance of the system with the HIPAA profile, and save scan results in the **hipaa\_report.html** file:

```
$ oscaped xccdf eval --report hipaa_report.html --profile hipaa /usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

### Additional resources

- **scap-security-guide(8)** and **oscaped(8)** man pages

## 16.6. REMEDIATING THE SYSTEM TO ALIGN WITH A SPECIFIC BASELINE USING AN SSG ANSIBLE PLAYBOOK

Use this procedure to remediate your system with a specific baseline using an Ansible playbook file from the SCAP Security Guide project. This example uses the Health Insurance Portability and Accountability Act (HIPAA) profile.



## WARNING

If not used carefully, running the system evaluation with the **Remediate** option enabled might render the system non-functional. 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.
- The **ansible-core** package is installed. See the [Ansible Installation Guide](#) for more information.



## NOTE

In RHEL 8.6 and later versions, Ansible Engine is replaced by the **ansible-core** package, which contains only built-in modules. Note that many Ansible remediations use modules from the community and Portable Operating System Interface (POSIX) collections, which are not included in the built-in modules. In this case, you can use Bash remediations as a substitute to Ansible remediations. The Red Hat Connector in RHEL 8 includes the necessary Ansible modules to enable the remediation playbooks to function with Ansible Core.

## Procedure

1. Remediate your system to align with HIPAA using Ansible:

```
# ansible-playbook -i localhost, -c local /usr/share/scap-security-guide/ansible/rhel8-playbook-hipaa.yml
```

2. Restart the system.

## Verification

1. Evaluate compliance of the system with the HIPAA profile, and save scan results in the **hipaa\_report.html** file:

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

## Additional resources

- **scap-security-guide(8)** and **oscap(8)** man pages
- [Ansible Documentation](#)

## 16.7. CREATING A REMEDIATION ANSIBLE PLAYBOOK TO ALIGN THE SYSTEM WITH A SPECIFIC BASELINE

You can create an Ansible playbook containing only the remediations that are required to align your system with a specific baseline. This example uses the Health Insurance Portability and Accountability Act (HIPAA) profile. With this procedure, you create a smaller playbook that does not cover already satisfied requirements. By following these steps, you do not modify your system in any way, you only prepare a file for later application.



### NOTE

In RHEL 8.6, Ansible Engine is replaced by the **ansible-core** package, which contains only built-in modules. Note that many Ansible remediations use modules from the community and Portable Operating System Interface (POSIX) collections, which are not included in the built-in modules. In this case, you can use Bash remediations as a substitute for Ansible remediations. The Red Hat Connector in RHEL 8.6 includes the necessary Ansible modules to enable the remediation playbooks to function with Ansible Core.

### Prerequisites

- The **scap-security-guide** package is installed.

### Procedure

1. Scan the system and save the results:

```
# oscap xccdf eval --profile hipaa --results hipaa-results.xml
/usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

2. Generate an Ansible playbook based on the file generated in the previous step:

```
# oscap xccdf generate fix --fix-type ansible --profile hipaa --output hipaa-remediations.yml
hipaa-results.xml
```

3. The **hipaa-remediations.yml** file contains Ansible remediations for rules that failed during the scan performed in step 1. After reviewing this generated file, you can apply it with the **ansible-playbook hipaa-remediations.yml** command.

### Verification

- In a text editor of your choice, review that the **hipaa-remediations.yml** file contains rules that failed in the scan performed in step 1.

### Additional resources

- **scap-security-guide(8)** and **oscap(8)** man pages
- [Ansible Documentation](#)

## 16.8. CREATING A REMEDIATION BASH SCRIPT FOR A LATER APPLICATION

Use this procedure to create a Bash script containing remediations that align your system with a security profile such as HIPAA. Using the following steps, you do not do any modifications to your system, you only prepare a file for later application.

### Prerequisites

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

### Procedure

1. Use the **oscap** command to scan the system and to save the results to an XML file. In the following example, **oscap** evaluates the system against the **hipaa** profile:

```
# oscap xccdf eval --profile hipaa --results hipaa-results.xml
/usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

2. Generate a Bash script based on the results file generated in the previous step:

```
# oscap xccdf generate fix --profile hipaa --fix-type bash --output hipaa-remediations.sh
hipaa-results.xml
```

3. The **hipaa-remediations.sh** file contains remediations for rules that failed during the scan performed in step 1. After reviewing this generated file, you can apply it with the **./hipaa-remediations.sh** command when you are in the same directory as this file.

### Verification

- In a text editor of your choice, review that the **hipaa-remediations.sh** file contains rules that failed in the scan performed in step 1.

### Additional resources

- **scap-security-guide(8)**, **oscap(8)**, and **bash(1)** man pages

## 16.9. SCANNING THE SYSTEM WITH A CUSTOMIZED PROFILE USING SCAP WORKBENCH

**SCAP Workbench**, which is contained in the **scap-workbench** package, 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 **SCAP Workbench** has limited functionality compared with the **oscap** command-line utility. **SCAP Workbench** processes security content in the form of data-stream files.

### 16.9.1. Using SCAP Workbench to scan and remediate the system

To evaluate your system against the selected security policy, use the following procedure.

#### Prerequisites

- The **scap-workbench** package is installed on your system.

#### Procedure

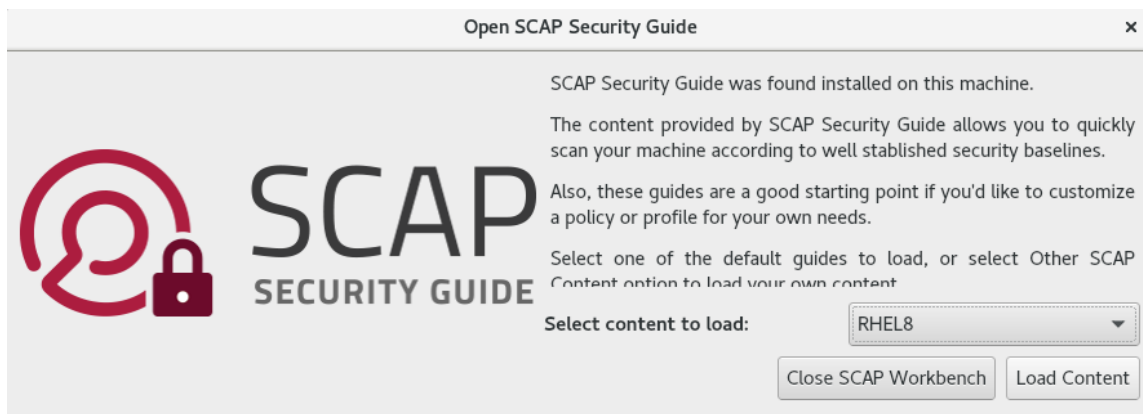


1. To run **SCAP Workbench** from the **GNOME Classic** desktop environment, press the **Super** key to enter the **Activities Overview**, type **scap-workbench**, and then press **Enter**. Alternatively, use:

```
$ scap-workbench &
```

2. Select a security policy using either of 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.



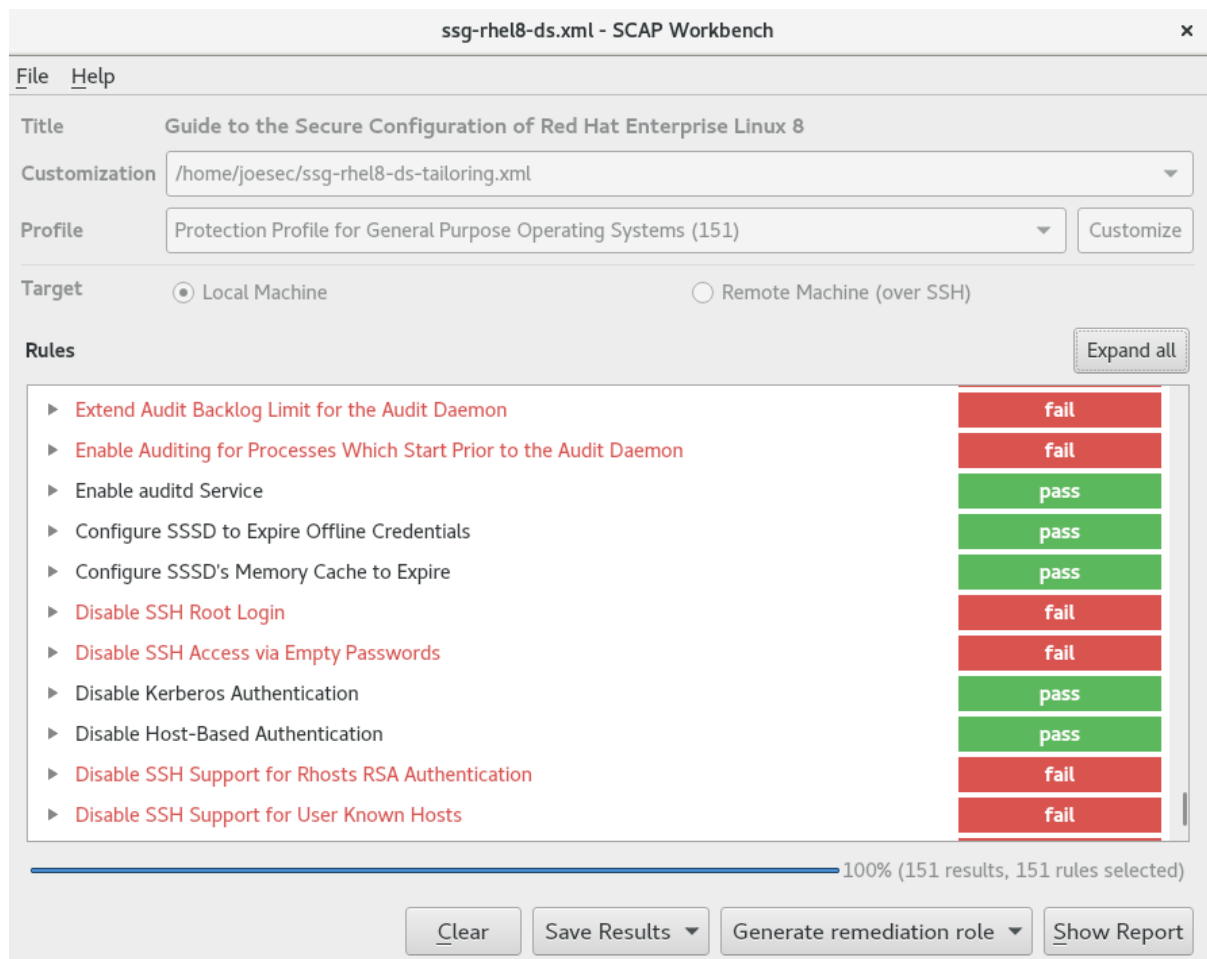
3. You can allow automatic correction of the system configuration by selecting the **Remediate** check box. With this option enabled, **SCAP Workbench** attempts to change the system configuration in accordance with the security rules applied by the policy. This process should fix the related checks that fail during the system scan.



#### WARNING

If not used carefully, running the system evaluation with the **Remediate** option enabled might render the system non-functional. 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.

4. Scan your system with the selected profile by clicking the **Scan** button.



- 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.

### 16.9.2. Customizing a security profile with SCAP Workbench

You can customize a security profile by changing parameters in certain rules (for example, minimum password length), removing rules that you cover in a different way, and selecting additional rules, to implement internal policies. You cannot define new rules by customizing a profile.

The following procedure demonstrates the use of **SCAP Workbench** for customizing (tailoring) a profile. You can also save the tailored profile for use with the **oscap** command-line utility.

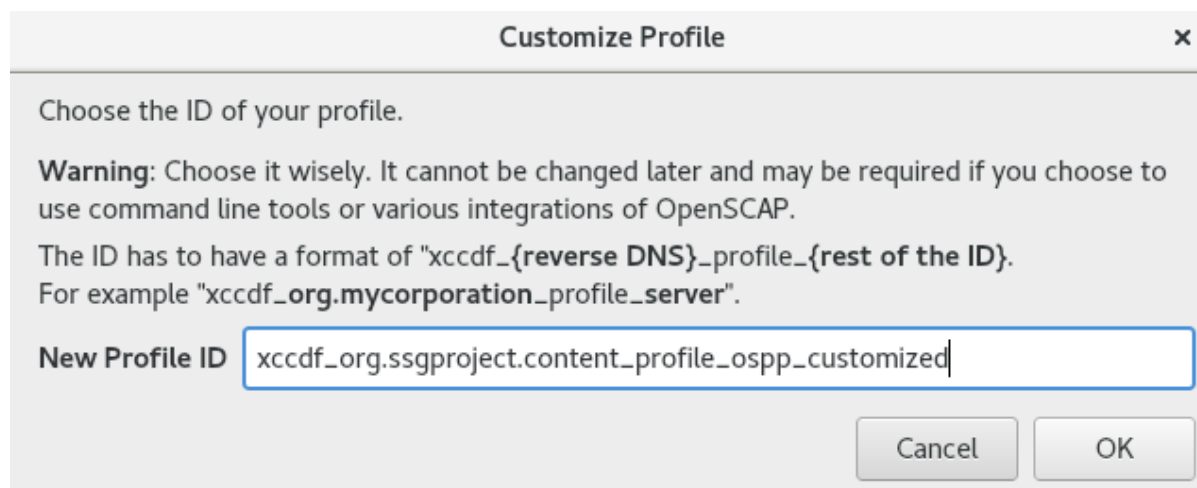
#### Prerequisites

- The **scap-workbench** package is installed on your system.

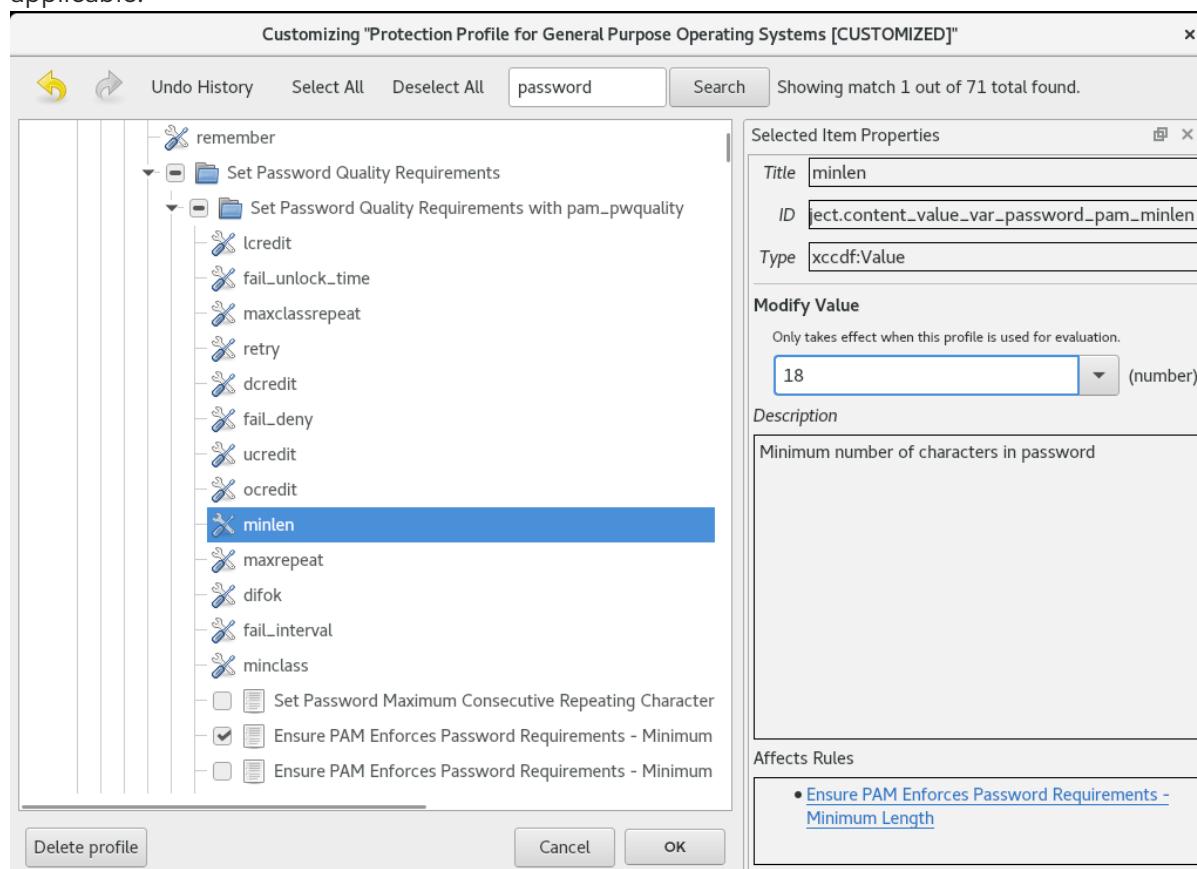
#### Procedure

- 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.
- To adjust the selected security profile according to your needs, click the **Customize** button.

This opens the new Customization window that enables you to modify the currently selected profile without changing the original data stream file. Choose a new profile ID.



3. Find a rule to modify using either the tree structure with rules organized into logical groups or the **Search** field.
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.  
If you select the **Into a directory** option, **SCAP Workbench** saves both the data stream file and the customization file to the specified location. You can use this as a backup solution.

By selecting the **As RPM** option, you can instruct **SCAP Workbench** to create an RPM package containing the data stream file and the customization file. This is useful for distributing the security content to systems that cannot be scanned remotely, and 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.

### 16.9.3. Additional resources

- **scap-workbench(8)** man page
- `/usr/share/doc/scap-workbench/user_manual.html` file provided by the **scap-workbench** package
- [Deploy customized SCAP policies with Satellite 6.x](#) KCS article

## 16.10. SCANNING CONTAINER AND CONTAINER IMAGES FOR VULNERABILITIES

Use this procedure to find security vulnerabilities in a container or a container image.



#### NOTE

The **oscap-podman** command is available from RHEL 8.2. For RHEL 8.1 and 8.0, use the workaround described in the [Using OpenSCAP for scanning containers in RHEL 8](#) Knowledgebase article.

### Prerequisites

- The **openscap-utils** and **bzip2** packages are installed.

### Procedure

1. Download the latest RHSA OVAL definitions for your system:

```
# wget -O - https://www.redhat.com/security/data/oval/v2/RHEL8/rhel-8.oval.xml.bz2 | bzip2 -
-decompress > rhel-8.oval.xml
```

2. Get the ID of a container or a container image, for example:

```
# podman images
REPOSITORY          TAG   IMAGE ID   CREATED   SIZE
registry.access.redhat.com/ubi8/ubi   latest  096cae65a207  7 weeks ago  239 MB
```

3. Scan the container or the container image for vulnerabilities and save results to the *vulnerability.html* file:

```
# oscap-podman 096cae65a207 oval eval --report vulnerability.html rhel-8.oval.xml
```

Note that the **oscap-podman** command requires root privileges, and the ID of a container is the first argument.

### Verification

- Check the results in a browser of your choice, for example:

```
$ firefox vulnerability.html &
```

### Additional resources

- For more information, see the **oscap-podman(8)** and **oscap(8)** man pages.

## 16.11. ASSESSING SECURITY COMPLIANCE OF A CONTAINER OR A CONTAINER IMAGE WITH A SPECIFIC BASELINE

Follow these steps to assess compliance of your container or a container image with a specific security baseline, such as Operating System Protection Profile (OSPP), Payment Card Industry Data Security Standard (PCI-DSS), and Health Insurance Portability and Accountability Act (HIPAA).



### NOTE

The **oscap-podman** command is available from RHEL 8.2. For RHEL 8.1 and 8.0, use the workaround described in the [Using OpenSCAP for scanning containers in RHEL 8](#) Knowledgebase article.

### Prerequisites

- The **openscap-utils** and **scap-security-guide** packages are installed.

### Procedure

1. Get the ID of a container or a container image, for example:

```
# podman images
REPOSITORY          TAG   IMAGE ID   CREATED   SIZE
registry.access.redhat.com/ubi8/ubi latest 096cae65a207 7 weeks ago 239 MB
```

2. Evaluate the compliance of the container image with the HIPAA profile and save scan results into the *report.html* HTML file

```
# oscap-podman 096cae65a207 xccdf eval --report report.html --profile hipaa
/usr/share/xml/scap/ssg/content/ssg-rhel8-ds.xml
```

Replace *096cae65a207* with the ID of your container image and the *hipaa* value with *ospp* or *pci-dss* if you assess security compliance with the OSPP or PCI-DSS baseline. Note that the **oscap-podman** command requires root privileges.

### Verification

- Check the results in a browser of your choice, for example:

```
$ firefox report.html &
```

**NOTE**

The rules marked as *notapplicable* are rules that do not apply to containerized systems. These rules apply only to bare-metal and virtualized systems.

**Additional resources**

- **oscap-podman(8)** and **scap-security-guide(8)** man pages.
- `/usr/share/doc/scap-security-guide/` directory.

**16.12. 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.

**16.12.1. 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.

**16.12.2. Performing integrity checks with AIDE**

## Prerequisites

- **AIDE** is properly installed and its database is initialized. See [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, configure the system to run **AIDE** weekly. Optimally, run **AIDE** 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
```

### 16.12.3. Updating an AIDE database

After verifying the changes of your system such as, package updates or configuration files adjustments, Red Hat recommends updating your baseline **AIDE** database.

## Prerequisites

- **AIDE** is properly installed and its database is initialized. See [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.

### 16.12.4. File-integrity tools: AIDE and IMA

Red Hat Enterprise Linux provides several tools for checking and preserving the integrity of files and directories on your system. The following table helps you decide which tool better fits your scenario.

**Table 16.2. Comparison between AIDE and IMA**

| Question | Advanced Intrusion Detection Environment (AIDE) | Integrity Measurement Architecture (IMA) |
|----------|-------------------------------------------------|------------------------------------------|
|----------|-------------------------------------------------|------------------------------------------|

| Question  | Advanced Intrusion Detection Environment (AIDE)                                                                                                                    | Integrity Measurement Architecture (IMA)                                                                                       |
|-----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| What      | AIDE is a utility that creates a database of files and directories on the system. This database serves for checking file integrity and detect intrusion detection. | IMA detects if a file is altered by checking file measurement (hash values) compared to previously stored extended attributes. |
| How       | AIDE uses rules to compare the integrity state of the files and directories.                                                                                       | IMA uses file hash values to detect the intrusion.                                                                             |
| Why       | Detection - AIDE detects if a file is modified by verifying the rules.                                                                                             | Detection and Prevention - IMA detects and prevents an attack by replacing the extended attribute of a file.                   |
| Usage     | AIDE detects a threat when the file or directory is modified.                                                                                                      | IMA detects a threat when someone tries to alter the entire file.                                                              |
| Extension | AIDE checks the integrity of files and directories on the local system.                                                                                            | IMA ensures security on the local and remote systems.                                                                          |

### 16.12.5. Additional resources

- [aide\(1\)](#) man page
- [Kernel integrity subsystem](#)

## 16.13. 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.

### 16.13.1. 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 uses 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
- Twofish (a 128-bit block cipher)
- Serpent

### Additional resources

- [LUKS Project Home Page](#)
- [LUKS On-Disk Format Specification](#)
- [FIPS PUB 197](#)

## 16.13.2. LUKS versions in RHEL

In RHEL, 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 must 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        | <b>cryptsetup reencrypt</b> |
| LUKS1        | <b>cryptsetup-reencrypt</b> |

## 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.

### 16.13.3. 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.

## 16.13.4. 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. Add an entry for a persistent mapping to **/etc/crypttab**

- a. Find the **luksUUID**:

```
# cryptsetup luksUUID /dev/mapper/sdb1_encrypted
```

This displays the **luksUUID** of the selected device.

- b. Open the **/etc/crypttab** file in a text editor of your choice and add a device in this file:

```
$ vi /etc/crypttab
```

```
/dev/mapper/sdb1_encrypted luks_uuid none
```

- c. Refresh initramfs with **dracut**:

```
$ dracut -f --regenerate-all
```

6. Add an entry for a persistent mounting to the **/etc/fstab** file:

- a. Find the **FS UUID** of the active LUKS block device

```
$ blkid -p /dev/mapper/sdb1_encrypted
```

- b. Open the **/etc/fstab** file in a text editor of your choice and add a device in this file, for example:

```
$ vi /etc/fstab  
fs_uuid /home auto rw,user,auto 0 0
```

7. Start the online encryption:

```
# cryptsetup reencrypt --resume-only /dev/sdb1
```

### Additional resources

- **cryptsetup(8)**, **lvextend(8)**, **resize2fs(8)**, and **parted(8)** man pages

### 16.13.5. 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

- **cryptsetup(8)** man page

### 16.13.6. Encrypting a blank block device using LUKS2

This procedure provides information about encrypting a blank block device using the LUKS2 format.

#### Prerequisites

- A blank block device.

#### Procedure

1. Setup a partition as an encrypted LUKS partition:

```
# cryptsetup luksFormat /dev/sdb1
```

2. Open an encrypted LUKS partition:

```
# cryptsetup open /dev/sdb1 sdb1_encrypted
```

This unlocks the partition and maps it to a new device using the device mapper. This alerts kernel that **device** is an encrypted device and should be addressed through LUKS using the **/dev/mapper/device\_mapped\_name** so as not to overwrite the encrypted data.

3. To write encrypted data to the partition, it must be accessed through the device mapped name. To do this, you must create a file system. For example:

```
# mkfs -t ext4 /dev/mapper/sdb1_encrypted
```

4. Mount the device:

```
# mount /dev/mapper/sdb1_encrypted mount-point
```

#### Additional resources

- **cryptsetup(8)** man page

### 16.13.7. Creating a LUKS encrypted volume using the storage RHEL System Role

You can use the **storage** role to create and configure a volume encrypted with LUKS by running an Ansible playbook.

#### Prerequisites

- Access and permissions to one or more *managed nodes*, which are systems you want to configure with the **crypto\_policies** System Role.

- Access and permissions to a control node, which is a system from which Red Hat Ansible Core configures other systems.

On the control node:

- The **ansible-core** and **rhel-system-roles** packages are installed.



### IMPORTANT

RHEL 8.0–8.5 provided access to a separate Ansible repository that contains Ansible Engine 2.9 for automation based on Ansible. Ansible Engine contains command-line utilities such as **ansible**, **ansible-playbook**, connectors such as **docker** and **podman**, and many plugins and modules. For information on how to obtain and install Ansible Engine, see the [How to download and install Red Hat Ansible Engine](#) Knowledgebase article.

RHEL 8.6 and 9.0 have introduced Ansible Core (provided as the **ansible-core** package), which contains the Ansible command-line utilities, commands, and a small set of built-in Ansible plugins. RHEL provides this package through the AppStream repository, and it has a limited scope of support. For more information, see the [Scope of support for the Ansible Core package included in the RHEL 9 and RHEL 8.6 and later AppStream repositories](#) Knowledgebase article.

- An inventory file which lists the managed nodes.

### Procedure

1. Create a new **playbook.yml** file with the following content:

```
- hosts: all
  vars:
    storage_volumes:
      - name: barefs
        type: disk
        disks:
          - sdb
        fs_type: xfs
        fs_label: label-name
        mount_point: /mnt/data
        encryption: true
        encryption_password: your-password
  roles:
    - rhel-system-roles.storage
```

2. Optional: Verify playbook syntax:

```
# ansible-playbook --syntax-check playbook.yml
```

3. Run the playbook on your inventory file:

```
# ansible-playbook -i inventory.file /path/to/file/playbook.yml
```

### Additional resources

- [Encrypting block devices using LUKS](#)

- `/usr/share/ansible/roles/rhel-system-roles.storage/README.md` file

## 16.14. CONFIGURING AUTOMATED UNLOCKING OF ENCRYPTED VOLUMES USING POLICY-BASED DECRYPTION

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 RHEL consists of the Clevis framework and plug-ins called *pins*. Each pin provides a separate unlocking capability. Currently, the following pins are available:

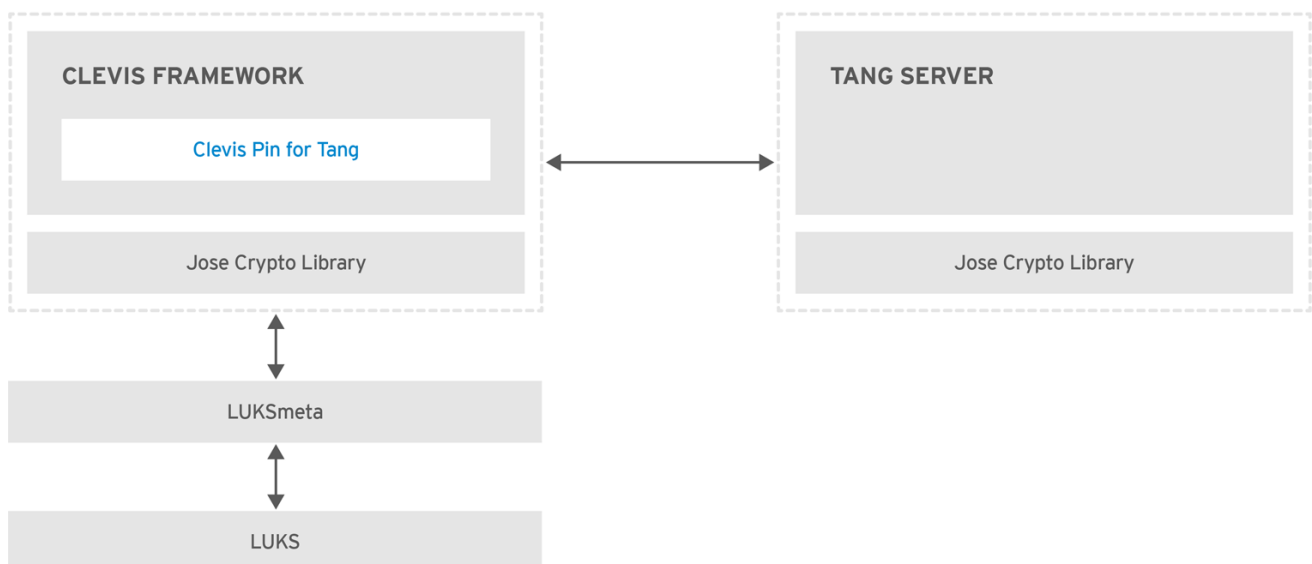
- **tang** - allows unlocking volumes using a network server
- **tpm2** - allows unlocking volumes using a TPM2 policy
- **sss** - allows deploying high-availability systems using the Shamir's Secret Sharing (SSS) cryptographic scheme

### 16.14.1. Network-bound disk encryption

The Network Bound Disc Encryption (NBDE) is a subcategory of Policy-Based Decryption (PBD) that allows binding encrypted volumes to a special network server. The current implementation of the NBDE includes a Clevis pin for the Tang server and the Tang server itself.

In RHEL, NBDE is implemented through the following components and technologies:

**Figure 16.1. 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 RHEL, 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*.

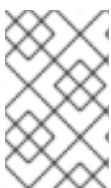
The *Clevis* pin for *Tang* uses one of the public keys to generate a unique, cryptographically-strong encryption key. Once the data is encrypted using this key, the key is discarded. The *Clevis* client should store the state produced by this provisioning operation in a convenient location. This process of encrypting data is the *provisioning step*.

The LUKS version 2 (LUKS2) is the default disk-encryption format in RHEL, 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. *Clevis* can encrypt plain-text files but you have to use the **cryptsetup** tool for encrypting block devices. See the [Encrypting block devices using LUKS](#) for more information.

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.



## NOTE

If the **kdump** kernel crash dumping mechanism is set to save the content of the system memory to a LUKS-encrypted device, you are prompted for entering a password during the second kernel boot.

## Additional resources

- [NBDE \(Network-Bound Disk Encryption\) Technology](#) Knowledgebase article
- **tang(8)**, **clevis(1)**, **jose(1)**, and **clevis-luks-unlockers(7)** man pages
- [How to set up Network-Bound Disk Encryption with multiple LUKS devices \(Clevis + Tang unlocking\)](#) Knowledgebase article

## 16.14.2. 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

- **clevis(1)** man page
- Built-in CLI help after entering the **clevis** command without any argument:

```
$ 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 edit  Edit a binding from a clevis-bound slot in a LUKS device
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 clevis binding
clevis luks report Report tang keys' rotations
clevis luks unbind Unbinds a pin bound to a LUKS volume
clevis luks unlock Unlocks a LUKS volume
```

### 16.14.3. 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 **polycoreutils-python-utils** package and its dependencies are installed.
- The **firewalld** service is running.

#### Procedure

1. To install the **tang** package and its dependencies, enter the following command as **root**:

```
# yum install tang
```

2. 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.

- Reload the changed configuration:

```
# systemctl daemon-reload
```

- Check that your configuration is working:

```
# systemctl show tangd.socket -p Listen
Listen=[::]:7500 (Stream)
```

- Start the **tangd** service:

```
# systemctl restart 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)**, **semanage(8)**, **firewall-cmd(1)**, **jose(1)**, **systemd.unit(5)**, and **systemd.socket(5)** man pages

#### 16.14.4. 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.

Alternatively, you can rotate Tang keys by using the **nbde\_server** RHEL system role. See [Using the nbde\\_server system role for setting up multiple Tang servers](#) for more information.

## 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. Rename all keys in the **/var/db/tang** key database directory 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 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
```

2. Check that you renamed and therefore hid all keys from the Tang server advertisement:

```
# ls -l
total 0
```

3. Generate new keys using the **/usr/libexec/tangd-keygen** command in **/var/db/tang** on the Tang server:

```
# /usr/libexec/tangd-keygen /var/db/tang
# ls /var/db/tang
3ZWS6-cDrCG61UPJS2BMmPU4I54.jwk zyLuX6hijUy_PSeUEFDi7hi38.jwk
```

4. Check that your Tang server advertises the signing key from the new key pair, for example:

```
# tang-show-keys 7500
3ZWS6-cDrCG61UPJS2BMmPU4I54
```

5. 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]
```

- 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
```

- 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

### 16.14.5. Configuring automated unlocking using a Tang key in the web console

Configure automated unlocking of a LUKS-encrypted storage device using a key provided by a Tang server.

#### Prerequisites

- The RHEL 8 web console has been installed. For details, see [Installing the web console](#).
- The **cockpit-storaged** package is installed on your system.
- The **cockpit.socket** service is running at port 9090.
- The **clevis**, **tang**, and **clevis-dracut** packages are installed.
- A Tang server is running.

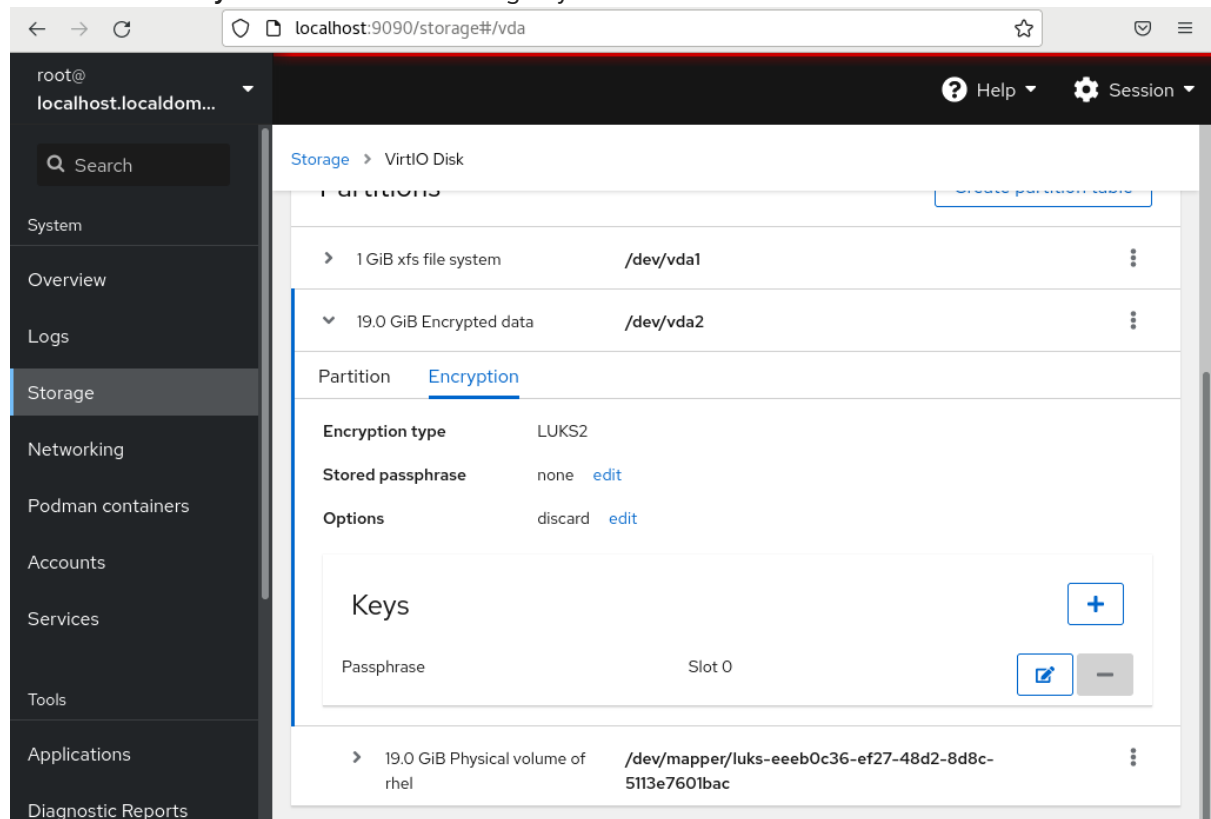
#### Procedure

- Open the RHEL web console by entering the following address in a web browser:

```
https://localhost:9090
```

Replace the *localhost* part by the remote server's host name or IP address when you connect to a remote system.

2. Provide your credentials and click **Storage**. Click **>** to expand details of the encrypted device you want to unlock using the Tang server, and click **Encryption**.
3. Click **+** in the **Keys** section to add a Tang key:



4. Provide the address of your Tang server and a password that unlocks the LUKS-encrypted device. Click **Add** to confirm:

### Add key

**Key source**  Passphrase  Tang keyserver

**Keyserver address**

**Disk passphrase**

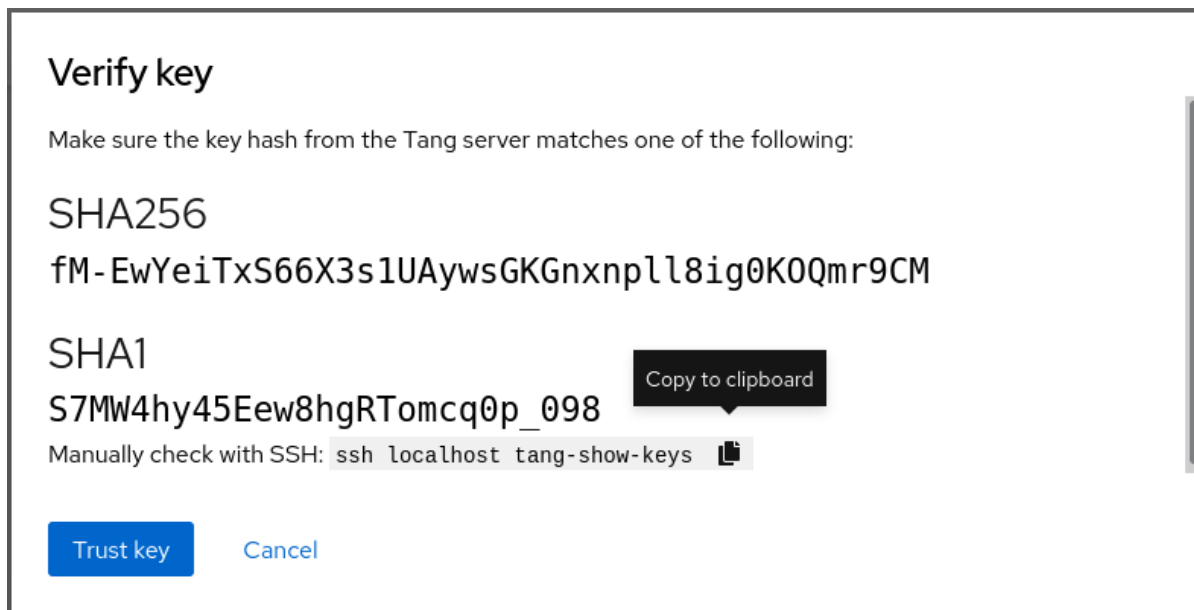
Saving a new passphrase requires unlocking the disk. Please provide a current disk passphrase.

The following dialog window provides a command to verify that the key hash matches.

5. In a terminal on the Tang server, use the **tang-show-keys** command to display the key hash for comparison. In this example, the Tang server is running on the port 7500:

```
# tang-show-keys 7500
fM-EwYeiTxS66X3s1UAYwsGKGnxnp1l8ig0KOQmr9CM
```

- Click **Trust key** when the key hashes in the web console and in the output of previously listed commands are the same:

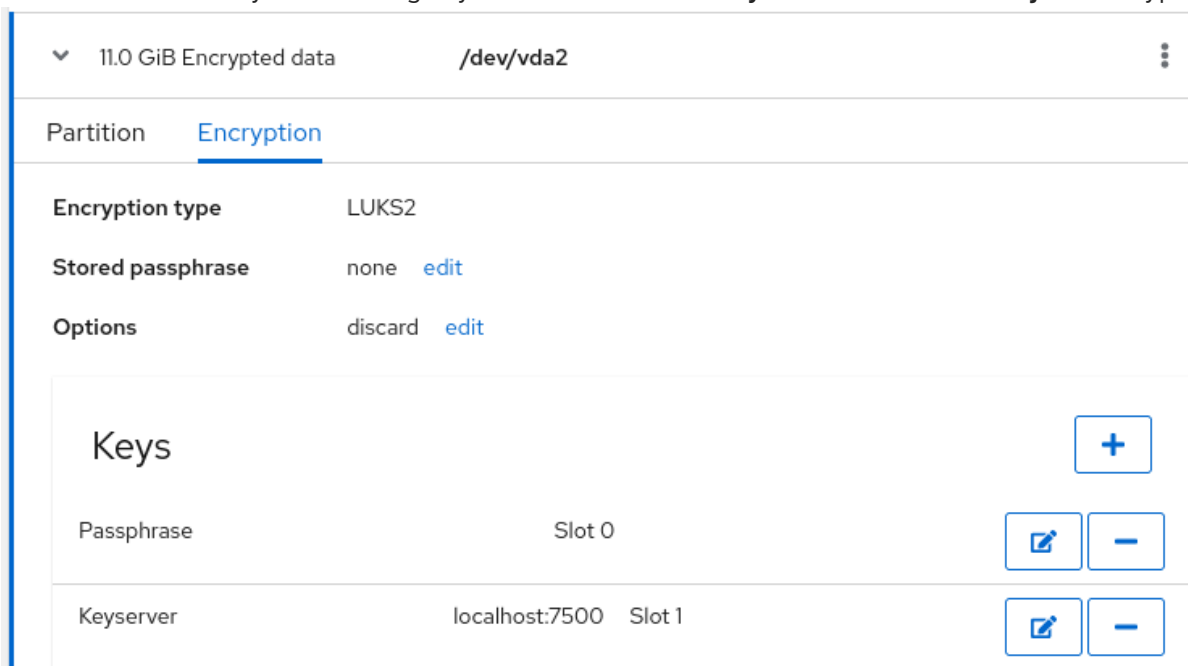


- To enable the early boot system to process the disk binding, click **Terminal** at the bottom of the left navigation bar and enter the following commands:

```
# yum install clevis-dracut
# grubby --update-kernel=ALL --args="rd.neednet=1"
# dracut -fv --regenerate-all
```

## Verification

- Check that the newly added Tang key is now listed in the **Keys** section with the **Keyserver** type:



- Verify that the bindings are available for the early boot, for example:

```
# lsinitrd | grep clevis
clevis
clevis-pin-sss
clevis-pin-tang
clevis-pin-tpm2
-rwxr-xr-x 1 root root 1600 Feb 11 16:30 usr/bin/clevis
-rwxr-xr-x 1 root root 1654 Feb 11 16:30 usr/bin/clevis-decrypt
...
-rwxr-xr-x 2 root root 45 Feb 11 16:30 usr/lib/dracut/hooks/initqueue/settled/60-
clevis-hook.sh
-rwxr-xr-x 1 root root 2257 Feb 11 16:30 usr/libexec/clevis-luks-askpass
```

### Additional resources

- [Configuring automated unlocking of encrypted volumes using policy-based decryption](#)

### 16.14.6. Basic NBDE and TPM2 encryption-client operations

The Clevis framework can encrypt plain-text files and decrypt both ciphertexts in the JSON Web Encryption (JWE) format and LUKS-encrypted block devices. Clevis clients can use either Tang network servers or Trusted Platform Module 2.0 (TPM 2.0) chips for cryptographic operations.

The following commands demonstrate the basic functionality provided by Clevis on examples containing plain-text files. You can also use them for troubleshooting your NBDE or Clevis+TPM deployments.

#### Encryption client bound to a Tang server

- To check that a Clevis encryption client binds 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 JWE 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"}
```

- To decrypt data, use the **clevis decrypt** command and provide the cipher text (JWE):



```
$ clevis decrypt < secret.jwe > output-plain.txt
```

### Encryption client using TPM 2.0

- To encrypt 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":"sha256","key":"rsa"}' < input-plain.txt > secret.jwe
```

- 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 PCR hashes values match the policy used when sealing.

For example, to seal the data to the PCR with index 0 and 7 for the SHA-256 bank:

```
$ clevis encrypt tpm2 '{"pcr_bank":"sha256","pcr_ids":"0,7"}' < input-plain.txt > secret.jwe
```



#### WARNING

Hashes in PCRs can be rewritten, and you no longer can unlock your encrypted volume. For this reason, add a strong passphrase that enable you to unlock the encrypted volume manually even when a value in a PCR changes.

If the system cannot automatically unlock your encrypted volume after an upgrade of the **shim-x64** package, follow the steps in the [Clevis TPM2 no longer decrypts LUKS devices after a restart](#) KCS article.

### Additional resources

- **clevis-encrypt-tang(1)**, **clevis-luks-unlockers(7)**, **clevis(1)**, and **clevis-encrypt-tpm2(1)** man pages
- **clevis**, **clevis decrypt**, and **clevis encrypt tang** commands without any arguments show the built-in CLI help, for example:

```
$ clevis encrypt tang
Usage: clevis encrypt tang CONFIG < PLAINTEXT > JWE
...
```

## 16.14.7. Configuring manual enrollment of LUKS-encrypted volumes

Use the following steps to configure unlocking of LUKS-encrypted volumes with NBDE.

### Prerequisites

- A Tang server is running and available.

### Procedure

1. To automatically unlock an existing LUKS-encrypted volume, install the **clevis-luks** subpackage:

```
# yum install clevis-luks
```

2. 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]
```

3. 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:
```

```
_OsIk0T-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.

The volume can now be unlocked with your existing password as well as with the Clevis policy.

- To enable the early boot system to process the disk binding, use the **dracut** tool on an already installed system:

```
# yum install clevis-dracut
```

In RHEL, Clevis produces a generic **initrd** (initial ramdisk) without host-specific configuration options and does not automatically add parameters such as **rd.neednet=1** to the kernel command line. If your configuration relies on a Tang pin that requires network during early boot, use the **--hostonly-cmdline** argument and **dracut** adds **rd.neednet=1** when it detects a Tang binding:

```
# dracut -fv --regenerate-all --hostonly-cmdline
```

Alternatively, create a `.conf` file in the `/etc/dracut.conf.d/`, and add the **hostonly\_cmdline=yes** option to the file, for example:

```
# echo "hostonly_cmdline=yes" > /etc/dracut.conf.d/clevis.conf
```

**NOTE**

You can also ensure that networking for a Tang pin is available during early boot by using the **grubby** tool on the system where Clevis is installed:

```
# grubby --update-kernel=ALL --args="rd.neednet=1"
```

Then you can use **dracut** without **--hostonly-cmdline**:

```
# dracut -fv --regenerate-all
```

**Verification**

- 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"
```

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"
```

Regenerate the initial RAM disk image:

```
# dracut -fv --regenerate-all
```

**Additional resources**

- **clevis-luks-bind(1)** and **dracut.cmdline(7)** man pages.
- [RHEL Network boot options](#)

**16.14.8. Configuring manual enrollment of LUKS-encrypted volumes using a TPM 2.0 policy**

Use the following steps to configure unlocking of LUKS-encrypted volumes by using a Trusted Platform Module 2.0 (TPM 2.0) policy.

**Prerequisites**

- An accessible TPM 2.0-compatible device.
- A system with the 64-bit Intel or 64-bit AMD architecture.

**Procedure**

1. To automatically unlock an existing LUKS-encrypted volume, install the **clevis-luks** subpackage:

```
# yum install clevis-luks
```

2. 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]
```

3. Bind the volume to a TPM 2.0 device using the **clevis luks bind** command, for example:

```
# clevis luks bind -d /dev/sda2 tpm2 '{"hash":"sha256","key":"rsa"}'
...
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.

Alternatively, if you want to seal data to specific Platform Configuration Registers (PCR) states, add the **pcr\_bank** and **pcr\_ids** values to the **clevis luks bind** command, for example:

```
# clevis luks bind -d /dev/sda2 tpm2
'{"hash":"sha256","key":"rsa","pcr_bank":"sha256","pcr_ids":"0,1"}
```



#### WARNING

Because the data can only be unsealed if PCR hashes values match the policy used when sealing and the hashes can be rewritten, add a strong passphrase that enable you to unlock the encrypted volume manually when a value in a PCR changes.

If the system cannot automatically unlock your encrypted volume after an upgrade of the **shim-x64** package, follow the steps in the [Clevis TPM2 no longer decrypts LUKS devices after a restart](#) KCS article.

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, use the **dracut** tool on an already installed system:

```
# yum install clevis-dracut
# dracut -fv --regenerate-all
```

## Verification

1. To verify that the Clevis JWE object is successfully placed in a LUKS header, use the **clevis luks list** command:

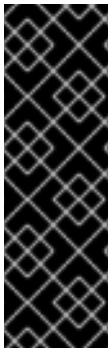
```
# clevis luks list -d /dev/sda2
1: tpm2 '{"hash":"sha256","key":"rsa"}
```

## Additional resources

- **clevis-luks-bind(1)**, **clevis-encrypt-tpm2(1)**, and **dracut.cmdline(7)** man pages

### 16.14.9. 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

1. 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*.

- In case of LUKS2 encryption, remove the token:

```
# cryptsetup token remove --token-id 0 /dev/sda2
```

- 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
```

- Wipe the key slot containing the Clevis passphrase:

```
# cryptsetup luksKillSlot /dev/sda2 1
```

### Additional resources

- clevis-luks-unbind(1)**, **cryptsetup(8)**, and **luksmeta(8)** man pages

## 16.14.10. 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 the enrollment of LUKS-encrypted volumes.

### Procedure

- Instruct Kickstart to partition the disk such that LUKS encryption has enabled for all mount points, other than **/boot**, with a temporary password. The password is temporary for this step of the enrollment process.

```
part /boot --fstype="xfs" --ondisk=vda --size=256
part / --fstype="xfs" --ondisk=vda --grow --encrypted --passphrase=temppass
```

Note that OSPP-compliant systems require a more complex configuration, for example:

```
part /boot --fstype="xfs" --ondisk=vda --size=256
part / --fstype="xfs" --ondisk=vda --size=2048 --encrypted --passphrase=temppass
part /var --fstype="xfs" --ondisk=vda --size=1024 --encrypted --passphrase=temppass
part /tmp --fstype="xfs" --ondisk=vda --size=1024 --encrypted --passphrase=temppass
part /home --fstype="xfs" --ondisk=vda --size=2048 --grow --encrypted --
passphrase=temppass
part /var/log --fstype="xfs" --ondisk=vda --size=1024 --encrypted --passphrase=temppass
part /var/log/audit --fstype="xfs" --ondisk=vda --size=1024 --encrypted --
passphrase=temppass
```

- Install the related Clevis packages by listing them in the **%packages** section:

```
%packages
clevis-dracut
clevis-luks
clevis-systemd
%end
```

- Optionally, to ensure that you can unlock the encrypted volume manually when required, add a strong passphrase before you remove the temporary passphrase. See the [How to add a passphrase, key, or keyfile to an existing LUKS device](#) article for more information.
- Call **clevis luks bind** to perform binding in the **%post** section. Afterward, remove the temporary password:

```
%post
clevis luks bind -y -k - -d /dev/vda2 \
tang '{"url":"http://tang.srv"}' <<< "temppass"
cryptsetup luksRemoveKey /dev/vda2 <<< "temppass"
dracut -fv --regenerate-all
%end
```

If your configuration relies on a Tang pin that requires network during early boot or you use NBDE clients with static IP configurations, you have to modify the **dracut** command as described in [Configuring manual enrollment of LUKS-encrypted volumes](#).

Note that the **-y** option for the **clevis luks bind** command is available from RHEL 8.3. In RHEL 8.2 and older, replace **-y** by **-f** in the **clevis luks bind** command and download the advertisement from the Tang server:

```
%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"
dracut -fv --regenerate-all
%end
```



#### 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](#)

### 16.14.11. 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

- **clevis-luks-unlockers(7)** man page

## 16.14.12. 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.

### 16.14.12.1. 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. When Clevis detects a smaller number of parts than specified in the threshold, it prints an error message.

#### 16.14.12.1.1. 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.

#### 16.14.12.1.2. 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":"0,7"}'}
```

The configuration scheme with the SSS threshold 't' set to '2' is now:

```
{
  "t":2,
  "pins":{
    "tang":[
      {
        "url":"http://tang1.srv"
      }
    ],
    "tpm2":{
      "pcr_ids":"0,7"
    }
  }
}
```

#### Additional resources

- **tang(8)** (section **High Availability**), **clevis(1)** (section **Shamir's Secret Sharing**), and **clevis-encrypt-sss(1)** man pages

### 16.14.13. 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 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 your scenario requires having encrypted root volumes in a cloud, perform the installation process (usually using Kickstart) for each instance of Red Hat Enterprise Linux in the cloud as well. The images cannot be shared without also sharing a LUKS master key.

To deploy automated unlocking in a virtualized environment, 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.

#### Additional resources

- **clevis-luks-bind(1)** man page

### 16.14.14. 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.

### 16.14.15. Deploying Tang as a container

The **tang** container image provides Tang-server decryption capabilities for Clevis clients that run either in OpenShift Container Platform (OCP) clusters or in separate virtual machines.

## Prerequisites

- The **podman** package and its dependencies are installed on the system.
- You have logged in on the **registry.redhat.io** container catalog using the **podman login registry.redhat.io** command. See [Red Hat Container Registry Authentication](#) for more information.
- The Clevis client is installed on systems containing LUKS-encrypted volumes that you want to automatically unlock by using a Tang server.

## Procedure

1. Pull the **tang** container image from the **registry.redhat.io** registry:

```
# podman pull registry.redhat.io/rhel8/tang
```

2. Run the container, specify its port, and specify the path to the Tang keys. The previous example runs the **tang** container, specifies the port `7500`, and indicates a path to the Tang keys of the `/var/db/tang` directory:

```
# podman run -d -p 7500:7500 -v tang-keys:/var/db/tang --name tang registry.redhat.io/rhel8/tang
```

Note that Tang uses port 80 by default but this may collide with other services such as the Apache HTTP server.

3. [Optional] For increased security, rotate the Tang keys periodically. You can use the **tangd-rotate-keys** script, for example:

```
# podman run --rm -v tang-keys:/var/db/tang registry.redhat.io/rhel8/tang tangd-rotate-keys -v -d /var/db/tang
Rotated key 'rZAMKAseaXBe0rcKXL1hCClq-DY.jwk' -> '.rZAMKAseaXBe0rcKXL1hCClq-DY.jwk'
Rotated key 'x1Alpc6WmnCU-CabD8_4q18vDuw.jwk' -> '.x1Alpc6WmnCU-CabD8_4q18vDuw.jwk'
Created new key GrMMX_WfdqomIU_4RyjpcdlXb0E.jwk
Created new key _dTTfn17sZZqVAp80u3ygFDHtjk.jwk
Keys rotated successfully.
```

## Verification

- On a system that contains LUKS-encrypted volumes for automated unlocking by the presence of the Tang server, check that the Clevis client can encrypt and decrypt a plain-text message using Tang:

```
# echo test | clevis encrypt tang '{"url":"http://localhost:7500"}' | clevis decrypt
The advertisement contains the following signing keys:

x1Alpc6WmnCU-CabD8_4q18vDuw
```

```
Do you wish to trust these keys? [ynYN] y
test
```

The previous example command shows the **test** string at the end of its output when a Tang server is available on the *localhost* URL and communicates through port *7500*.

### Additional resources

- **podman(1)**, **clevis(1)**, and **tang(8)** man pages
- For more details on automated unlocking of LUKS-encrypted volumes using Clevis and Tang, see the [Configuring automated unlocking of encrypted volumes using policy-based decryption](#) chapter.

### 16.14.16. Introduction to the `nbde_client` and `nbde_server` System Roles (Clevis and Tang)

RHEL System Roles is a collection of Ansible roles and modules that provide a consistent configuration interface to remotely manage multiple RHEL systems.

RHEL 8.3 introduced Ansible roles for automated deployments of Policy-Based Decryption (PBD) solutions using Clevis and Tang. The **rhel-system-roles** package contains these system roles, related examples, and also the reference documentation.

The **nbde\_client** System Role enables you to deploy multiple Clevis clients in an automated way. Note that the **nbde\_client** role supports only Tang bindings, and you cannot use it for TPM2 bindings at the moment.

The **nbde\_client** role requires volumes that are already encrypted using LUKS. This role supports to bind a LUKS-encrypted volume to one or more Network-Bound (NBDE) servers - Tang servers. You can either preserve the existing volume encryption with a passphrase or remove it. After removing the passphrase, you can unlock the volume only using NBDE. This is useful when a volume is initially encrypted using a temporary key or password that you should remove after you provision the system.

If you provide both a passphrase and a key file, the role uses what you have provided first. If it does not find any of these valid, it attempts to retrieve a passphrase from an existing binding.

PBD defines a binding as a mapping of a device to a slot. This means that you can have multiple bindings for the same device. The default slot is slot 1.

The **nbde\_client** role provides also the **state** variable. Use the **present** value for either creating a new binding or updating an existing one. Contrary to a **clevis luks bind** command, you can use **state: present** also for overwriting an existing binding in its device slot. The **absent** value removes a specified binding.

Using the **nbde\_client** System Role, you can deploy and manage a Tang server as part of an automated disk encryption solution. This role supports the following features:

- Rotating Tang keys
- Deploying and backing up Tang keys

### Additional resources

- For a detailed reference on Network-Bound Disk Encryption (NBDE) role variables, install the

**rhel-system-roles** package, and see the **README.md** and **README.html** files in the `/usr/share/doc/rhel-system-roles/nbde_client/` and `/usr/share/doc/rhel-system-roles/nbde_server/` directories.

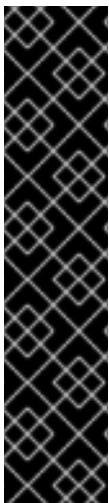
- For example system-roles playbooks, install the **rhel-system-roles** package, and see the `/usr/share/ansible/roles/rhel-system-roles.nbde_server/examples/` directories.
- For more information on RHEL System Roles, see [Introduction to RHEL System Roles](#)

### 16.14.17. Using the `nbde_server` System Role for setting up multiple Tang servers

Follow the steps to prepare and apply an Ansible playbook containing your Tang server settings.

#### Prerequisites

- Access and permissions to one or more *managed nodes*, which are systems you want to configure with the **nbde\_server** System Role.
- Access and permissions to a control node, which is a system from which Red Hat Ansible Core configures other systems.  
On the control node:
  - The **ansible-core** and **rhel-system-roles** packages are installed.



#### IMPORTANT

RHEL 8.0–8.5 provided access to a separate Ansible repository that contains Ansible Engine 2.9 for automation based on Ansible. Ansible Engine contains command-line utilities such as **ansible**, **ansible-playbook**, connectors such as **docker** and **podman**, and many plugins and modules. For information on how to obtain and install Ansible Engine, see the [How to download and install Red Hat Ansible Engine](#) Knowledgebase article.

RHEL 8.6 and 9.0 have introduced Ansible Core (provided as the **ansible-core** package), which contains the Ansible command-line utilities, commands, and a small set of built-in Ansible plugins. RHEL provides this package through the AppStream repository, and it has a limited scope of support. For more information, see the [Scope of support for the Ansible Core package included in the RHEL 9 and RHEL 8.6 and later AppStream repositories](#) Knowledgebase article.

- An inventory file which lists the managed nodes.

#### Procedure

1. Prepare your playbook containing settings for Tang servers. You can either start from the scratch, or use one of the example playbooks from the `/usr/share/ansible/roles/rhel-system-roles.nbde_server/examples/` directory.

```
# cp /usr/share/ansible/roles/rhel-system-roles.nbde_server/examples/simple_deploy.yml
./my-tang-playbook.yml
```

2. Edit the playbook in a text editor of your choice, for example:

```
# vi my-tang-playbook.yml
```

3. Add the required parameters. The following example playbook ensures deploying of your Tang server and a key rotation:

```
---
- hosts: all

  vars:
    nbde_server_rotate_keys: yes

  roles:
    - rhel-system-roles.nbde_server
```

4. Apply the finished playbook:

```
# ansible-playbook -i inventory-file my-tang-playbook.yml
```

Where: \* **inventory-file** is the inventory file. \* **logging-playbook.yml** is the playbook you use.



### IMPORTANT

To ensure that networking for a Tang pin is available during early boot by using the **grubby** tool on the systems where Clevis is installed:

```
# grubby --update-kernel=ALL --args="rd.neednet=1"
```

### Additional resources

- For more information, install the **rhel-system-roles** package, and see the `/usr/share/doc/rhel-system-roles/nbde_server/` and `usr/share/ansible/roles/rhel-system-roles.nbde_server/` directories.

## 16.14.18. Using the `nbde_client` System Role for setting up multiple Clevis clients

Follow the steps to prepare and apply an Ansible playbook containing your Clevis client settings.



### NOTE

The **nbde\_client** System Role supports only Tang bindings. This means that you cannot use it for TPM2 bindings at the moment.

### Prerequisites

- Access and permissions to one or more *managed nodes*, which are systems you want to configure with the **nbde\_client** System Role.
- Access and permissions to a *control node*, which is a system from which Red Hat Ansible Core configures other systems.
- The Ansible Core package is installed on the control machine.
- The **rhel-system-roles** package is installed on the system from which you want to run the playbook.

### Procedure

**Procedure**

1. Prepare your playbook containing settings for Clevis clients. You can either start from the scratch, or use one of the example playbooks from the `/usr/share/ansible/roles/rhel-system-roles.nbde_client/examples/` directory.

```
# cp /usr/share/ansible/roles/rhel-system-roles.nbde_client/examples/high_availability.yml
./my-clevis-playbook.yml
```

2. Edit the playbook in a text editor of your choice, for example:

```
# vi my-clevis-playbook.yml
```

3. Add the required parameters. The following example playbook configures Clevis clients for automated unlocking of two LUKS-encrypted volumes by when at least one of two Tang servers is available:

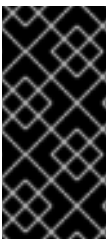
```
---
- hosts: all

vars:
  nbde_client_bindings:
    - device: /dev/rhel/root
      encryption_key_src: /etc/luks/keyfile
  servers:
    - http://server1.example.com
    - http://server2.example.com
  - device: /dev/rhel/swap
    encryption_key_src: /etc/luks/keyfile
  servers:
    - http://server1.example.com
    - http://server2.example.com

roles:
  - rhel-system-roles.nbde_client
```

4. Apply the finished playbook:

```
# ansible-playbook -i host1,host2,host3 my-clevis-playbook.yml
```

**IMPORTANT**

To ensure that networking for a Tang pin is available during early boot by using the **grubby** tool on the system where Clevis is installed:

```
# grubby --update-kernel=ALL --args="rd.neednet=1"
```

**Additional resources**

- For details about the parameters and additional information about the NBDE Client System Role, install the **rhel-system-roles** package, and see the `/usr/share/doc/rhel-system-roles/nbde_client/` and `/usr/share/ansible/roles/rhel-system-roles.nbde_client/` directories.



## CHAPTER 17. USING SELINUX

### 17.1. GETTING STARTED WITH SELINUX

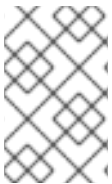
Security Enhanced Linux (SELinux) provides an additional layer of system security. SELinux fundamentally answers the question: *May <subject> do <action> to <object>?*, for example: *May a web server access files in users' home directories?*

#### 17.1.1. Introduction to SELinux

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.

Security Enhanced Linux (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.



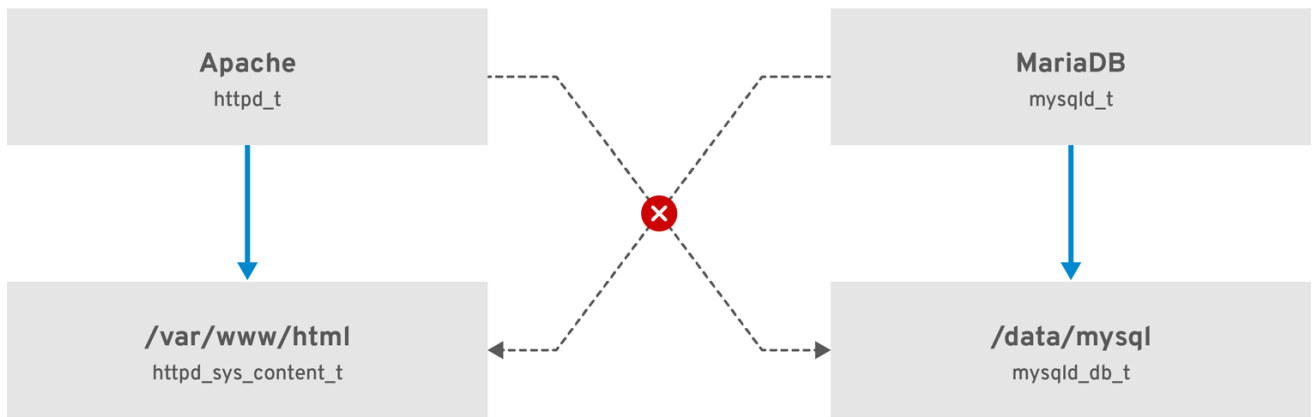
#### NOTE

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 17.1. An example how can SELinux help to run Apache and MariaDB in a secure way.



RHEL\_467048\_0218

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 **mysql\_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

- **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](#)

### 17.1.2. 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.

### 17.1.3. 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**.
- 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 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 RHEL, even if an administrator forgets to limit which servers can perform a zone transfer, the default SELinux policy prevent updates for zone files [2] that use zone transfers, by the BIND **named** daemon itself, and by other processes.
- Without SELinux, an attacker can misuse a vulnerability to path traversal on an Apache web server and access files and directories stored on the file system by using special elements such as **../**. If an attacker attempts an attack on a server running with SELinux in enforcing mode, SELinux denies access to files that the **httpd** process must not access. SELinux cannot block this type of attack completely but it effectively mitigates it.
- SELinux in enforcing mode successfully prevents exploitation of kernel NULL pointer dereference operators on non-SMAP platforms (CVE-2019-9213). Attackers use a vulnerability in the **mmap** function, which does not check mapping of a null page, for placing arbitrary code on this page.
- The **deny\_ptrace** SELinux boolean and SELinux in enforcing mode protect systems from the **PTRACE\_TRACEME** vulnerability (CVE-2019-13272). Such configuration prevents scenarios when an attacker can get **root** privileges.

- The **nfs\_export\_all\_rw** and **nfs\_export\_all\_ro** SELinux booleans provide an easy-to-use tool to prevent misconfigurations of Network File System (NFS) such as accidental sharing **/home** directories.

#### Additional resources

- [SELinux as a security pillar of an operating system - Real-world benefits and examples](#)  
Knowledgebase article

### 17.1.4. SELinux architecture and packages

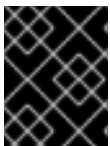
SELinux is a Linux Security Module (LSM) that is built into the Linux kernel. The SELinux subsystem in the kernel is driven by a security policy which is controlled by the administrator and loaded at boot. All security-relevant, kernel-level access operations on the system are intercepted by SELinux 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 RHEL 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.

If a process is sending a **D-Bus** message to another process and if the SELinux policy does not allow the **D-Bus** communication of these two processes, then the system prints a **USER\_AVC** denial message, and the D-Bus communication times out. Note that the D-Bus communication between two processes works bidirectionally.



#### IMPORTANT

To avoid incorrect SELinux labeling and subsequent problems, ensure that you start services using a **systemctl start** command.

RHEL 8 provides the following packages for working with SELinux:

- policies: **selinux-policy-targeted**, **selinux-policy-mls**
- tools: **policycoreutils**, **policycoreutils-gui**, **libselenium-utils**, **policycoreutils-python-utils**, **setools-console**, **checkpolicy**

## 17.1.5. 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.

## 17.2. 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.

### 17.2.1. 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
```



## WARNING

When systems run SELinux in permissive mode, users and processes might 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, SELinux automatically relabels file systems when changing from the disabled state to permissive or enforcing 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.

Before rebooting the system for relabeling, make sure the system will boot in permissive mode, for example by using the **enforcing=0** kernel option. This prevents the system from failing to boot in case the system contains unlabeled files required by **systemd** before launching the **selinux-autorelabel** service. For more information, see [RHBZ#2021835](#).

## 17.2.2. 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 **polycoreutils** 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

1. After the system restarts, confirm that the **getenforce** command returns **Permissive**:

```
$ getenforce
Permissive
```

### 17.2.3. 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

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.

## 17.2.4. Enabling SELinux on systems that previously had it disabled

To avoid problems, such as systems unable to boot or process failures, follow this procedure when enabling SELinux on systems that previously had it disabled.





## WARNING

When systems run SELinux in permissive mode, users and processes might 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, SELinux automatically relabels file systems when changing from the disabled state to permissive or enforcing mode.

Before rebooting the system for relabeling, make sure the system will boot in permissive mode, for example by using the **enforcing=0** kernel option. This prevents the system from failing to boot in case the system contains unlabeled files required by **systemd** before launching the **selinux-autorelabel** service. For more information, see [RHBZ#2021835](#).

## 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. Ensure that files are relabeled upon the next reboot:

```
# fixfiles -F onboot
```

This creates the **/.autorelabel** file containing the **-F** option.



## WARNING

Always switch to permissive mode before entering the **fixfiles -F onboot** command. This prevents the system from failing to boot in case the system contains unlabeled files. For more information, see [RHBZ#2021835](#).

5. If there are no denials, switch to enforcing mode. For more information, see [Changing SELinux modes at boot time](#).

## Verification

1. After the system restarts, confirm that the **getenforce** command returns **Enforcing**:

■

\$ getenforce  
Enforcing



## NOTE

To run custom applications with SELinux in enforcing mode, choose one of the following scenarios:

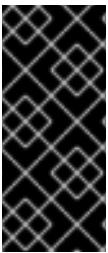
- Run your application in the **unconfined\_service\_t** domain.
- Write a new policy for your application. See the [Writing a custom SELinux policy](#) section for more information.

## Additional resources

- [SELinux states and modes](#) section covers temporary changes in modes.

## 17.2.5. 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

1. After reboot, confirm that the **getenforce** command returns **Disabled**:

```
$ getenforce
Disabled
```

### 17.2.6. Changing SELinux modes at boot time

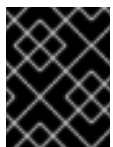
On boot, you can set several kernel parameters to change the way SELinux runs:

#### enforcing=0

Setting this parameter causes the system to start 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 from a series of the same denials is reported. However, in enforcing mode, you might get a denial related to 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
```

## 17.3. 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.

### 17.3.1. Identifying SELinux denials

Follow only the necessary steps from this procedure; in most cases, you need to perform just step 1.

#### Procedure

1. 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.

2. 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
```

3. 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
```

4. 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.

### 17.3.2. 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

- 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.  
After you finish the process, disable full-path auditing:

```
# auditctl -W /etc/shadow -p w -k shadow-write
```

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

- **sealert(8)** man page

### 17.3.3. 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, status=0/SUCCESS)
   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**.

### 17.3.4. 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

- **auditd(8)** and **aureport(8)** man pages

#### 17.3.5. Additional resources

- [Basic SELinux Troubleshooting in CLI](#)
- [What is SELinux trying to tell me? The 4 key causes of SELinux errors](#)

---

[2] Text files that include DNS information, such as hostname to IP address mappings.

## PART III. DESIGN OF NETWORK

## CHAPTER 18. CONFIGURING IP NETWORKING WITH 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 named **ifcfg-name\_pass**, 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.



### IMPORTANT

NetworkManager supports profiles stored in the keyfile format. However, by default, NetworkManager uses the **ifcfg** format when you use the NetworkManager API to create or update profiles.

In a future major RHEL release, the keyfile format will be default. Consider using the keyfile format if you want to manually create and manage configuration files. For details, see [Manually creating NetworkManager profiles in keyfile format](#).

### 18.1. CONFIGURING AN INTERFACE WITH STATIC NETWORK SETTINGS USING IFCFG FILES

If you do not use the NetworkManager utilities and applications, you can manually configure a network interface by creating **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:db8:1::2/64
```

#### Additional resources

- **nm-settings-ifcfg-rh(5)** man page

## 18.2. CONFIGURING AN INTERFACE WITH DYNAMIC NETWORK SETTINGS USING IFCFG FILES

If you do not use the NetworkManager utilities and applications, you can manually configure a network interface by creating **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
```

- 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

You can use only one of these settings. If you specify both **DHCP\_HOSTNAME** and **DHCP\_FQDN**, only **DHCP\_FQDN** is used.

3. 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.

## 18.3. MANAGING SYSTEM-WIDE AND PRIVATE CONNECTION PROFILES WITH IFCFG FILES

By default, all users on a host can use the connections defined in **ifcfg** files. You can limit this behavior to specific users by adding the **USERS** parameter to the **ifcfg** file.

### Prerequisite

- The **ifcfg** file already exists.

### Procedure

1. Edit the **ifcfg** file in the **/etc/sysconfig/network-scripts/** directory that you want to limit to certain users, and add:

```
USERS="username1 username2 ..."
```

2. Reactive the connection:

```
# nmcli connection up connection_name
```

## CHAPTER 19. GETTING STARTED WITH IPVLAN

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.

### 19.1. 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.

### 19.2. 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 losing 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.

## 19.3. 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 I2
```

Note that network interface controller (NIC) is a hardware component which connects a computer to a network.

#### Example 19.1. Creating an IPVLAN device

```
# ip link add link enp0s31f6 name my_ipvlan type ipvlan mode I2
# 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 route add dev <real_NIC_device> <peer_IP_address/32>
```

For **L3S mode**:

```
# ip route add dev 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 20. REUSING THE SAME IP ADDRESS ON DIFFERENT INTERFACES

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.

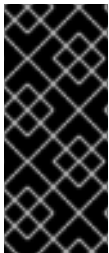
One benefit of VRF over partitioning on layer 2 is that routing scales better considering the number of peers involved.

Red Hat Enterprise Linux uses a virtual **vrf** device for each VRF domain and adds routes to a VRF domain by adding existing network devices to a VRF device. Addresses and routes previously attached to the original device will be moved inside the VRF domain.

Note that each VRF domain is isolated from each other.

### 20.1. PERMANENTLY REUSING THE SAME IP ADDRESS ON DIFFERENT INTERFACES

You can use the virtual routing and forwarding (VRF) feature to permanently use the same IP address on different interfaces in one server.



#### 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 a connection for the VRF device and assign it to a routing table. For example, to create a VRF device named **vrf0** that is assigned to the **1001** routing table:

```
# nmcli connection add type vrf ifname vrf0 con-name vrf0 table 1001 ipv4.method disabled ipv6.method disabled
```

- b. Enable the **vrf0** device:

```
# nmcli connection up vrf0
```

- c. Assign a network device to the VRF just created. For example, to add the **enp1s0** Ethernet device to the **vrf0** VRF device and assign an IP address and the subnet mask to **enp1s0**, enter:

```
# nmcli connection add type ethernet con-name vrf.enp1s0 ifname enp1s0 master
vrf0 ipv4.method manual ipv4.address 192.0.2.1/24
```

- d. Activate the **vrf.enp1s0** connection:

```
# nmcli connection up vrf.enp1s0
```

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 **vrf1** that is assigned to the **1002** routing table, enter:

```
# nmcli connection add type vrf ifname vrf1 con-name vrf1 table 1002 ipv4.method
disabled ipv6.method disabled
```

- b. Activate the **vrf1** device:

```
# nmcli connection up vrf1
```

- c. Assign a network device to the VRF just created. For example, to add the **enp7s0** Ethernet device to the **vrf1** VRF device and assign an IP address and the subnet mask to **enp7s0**, enter:

```
# nmcli connection add type ethernet con-name vrf.enp7s0 ifname enp7s0 master
vrf1 ipv4.method manual ipv4.address 192.0.2.1/24
```

- d. Activate the **vrf.enp7s0** device:

```
# nmcli connection up vrf.enp7s0
```

## 20.2. TEMPORARILY REUSING THE SAME IP ADDRESS ON DIFFERENT INTERFACES

You can use the virtual routing and forwarding (VRF) feature to temporarily use the same IP address on different interfaces in one server. 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.

## 20.3. ADDITIONAL RESOURCES

- `/usr/share/doc/kernel-doc-<kernel_version>/Documentation/networking/vrf.txt` from the `kernel-doc` package

## CHAPTER 21. SECURING NETWORKS

### 21.1. USING SECURE COMMUNICATIONS BETWEEN TWO SYSTEMS WITH OPENSSSH

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.

#### 21.1.1. 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.

A host key authenticates hosts in the SSH protocol. Host keys are cryptographic keys that are generated automatically when OpenSSH is first installed, or when the host boots for the first time.

OpenSSH is an implementation of the SSH protocol supported by 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 RHEL supports only SSH version 2. It has an enhanced key-exchange algorithm that is not vulnerable to exploits known in version 1.

OpenSSH, as one of core cryptographic subsystems of RHEL, uses system-wide crypto policies. This ensures that weak cipher suites and cryptographic algorithms are disabled in the default configuration. To modify the policy, the administrator must either use the **update-crypto-policies** command to adjust the settings or manually opt out of the system-wide crypto policies.

The OpenSSH suite uses two sets of configuration files: one for client programs (that is, **ssh**, **scp**, and **sftp**), and another 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 listed by using the **man -k ssh** command
- [Using system-wide cryptographic policies](#)

## 21.1.2. 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.
```

Ensure that the **Banner** option is not commented out in `/etc/ssh/sshd_config` and its value contains **/etc/issue**:

```
# less /etc/ssh/sshd_config | grep Banner
Banner /etc/issue
```

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 and restart **sshd** to apply the changes:

```
# systemctl daemon-reload
# systemctl restart sshd
```

## Verification

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++IPegSy31v3L/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.

### 21.1.3. Setting an OpenSSH server for key-based authentication



To improve system security, enforce key-based authentication by disabling password authentication on your OpenSSH server.

### 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.

## 21.1.4. 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/joesech/.ssh/id_ecdsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/joesech/.ssh/id_ecdsa.
Your public key has been saved in /home/joesech/.ssh/id_ecdsa.pub.
The key fingerprint is:
SHA256:Q/x+qms4j7PCQ0qFd09iZEFHA+SqwBKRNauU72oZfaCI
joesech@localhost.example.com
The key's randomart image is:
+---[ECDSA 256]---+
|.oo..o=++      |
|.. o .oo .     |
|.. .. o. o     |
|...o.+...     |
|o.oo.o +S .    |
|.=.+ . .o     |
|E.*+ . . .    |
|.=.+ +.. o    |
| . 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.

- To copy the public key to a remote machine:

```

$ ssh-copy-id joesech@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
joesech@ssh-server-example.com's password:
...
Number of key(s) added: 1

```

Now try logging into the machine, with: "ssh 'joesech@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

- Log in to the OpenSSH server without providing any password:

```

$ ssh joesech@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.

### 21.1.5. Using SSH keys stored on a smart card

Red Hat Enterprise Linux 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)**, **opensc.conf(5)**, **pcscd(8)**, **ssh(1)**, and **ssh-keygen(1)** man pages

## 21.1.6. 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**:

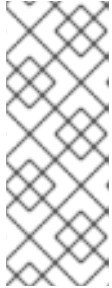
```
PasswordAuthentication no
```

### 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, 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
```



## NOTE

In images with **cloud-init** enabled, the **ssh-keygen** units are automatically disabled. This is because the **ssh-keygen template** service can interfere with the **cloud-init** tool and cause problems with host key generation. To prevent these problems the **etc/systemd/system/sshd-keygen@.service.d/disable-sshd-keygen-if-cloud-init-active.conf** drop-in configuration file disables the **ssh-keygen** units if **cloud-init** is running.

- 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 therefore 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 **policyscoreutils-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 allowlists (directives starting with Allow) is more secure than using blocklists (options starting with Deny) because allowlists 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 [Security hardening](#) title for more information.

### Additional resources

- **sshd\_config(5)**, **ssh-keygen(1)**, **crypto-policies(7)**, and **update-crypto-policies(8)** man pages.

### 21.1.7. Connecting to a remote server using an SSH jump host

Use this procedure for connecting your local system to a remote server through an intermediary server, also called jump host.

#### Prerequisites

- A jump host accepts SSH connections from your local system.
- A remote server accepts SSH connections only from the jump host.

#### Procedure

1. Define the jump host by editing the `~/.ssh/config` file on your local system, for example:

```
Host jump-server1
  HostName jump1.example.com
```

- The **Host** parameter defines a name or alias for the host you can use in **ssh** commands. The value can match the real host name, but can also be any string.
  - The **HostName** parameter sets the actual host name or IP address of the jump host.
2. Add the remote server jump configuration with the **ProxyJump** directive to `~/.ssh/config` file on your local system, for example:

```
Host remote-server
  HostName remote1.example.com
  ProxyJump jump-server1
```

3. Use your local system to 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.

## 21.1.8. Connecting to remote machines with SSH keys using ssh-agent

To avoid entering a passphrase each time you initiate an SSH connection, you can use the **ssh-agent** utility to cache the private SSH key. The private key and the passphrase remain secure.

### Prerequisites

- You have a remote host with SSH daemon running and reachable through the network.
- You know the IP address or hostname and credentials to log in to the remote host.
- You have generated an SSH key pair with a passphrase and transferred the public key to the remote machine.

### Procedure

1. Optional: Verify you can use the key to authenticate to the remote host:

- a. Connect to the remote host using SSH:

```
$ ssh example.user1@198.51.100.1 hostname
```

- b. Enter the passphrase you set while creating the key to grant access to the private key.

```
$ ssh example.user1@198.51.100.1 hostname
host.example.com
```

2. Start the **ssh-agent**.

```
$ eval $(ssh-agent)
Agent pid 20062
```

3. Add the key to **ssh-agent**.



```
$ ssh-add ~/.ssh/id_rsa
Enter passphrase for ~/.ssh/id_rsa:
Identity added: ~/.ssh/id_rsa (example.user0@198.51.100.12)
```

### Verification

- Optional: Log in to the host machine using SSH.

```
$ ssh example.user1@198.51.100.1

Last login: Mon Sep 14 12:56:37 2020
```

Note that you did not have to enter the passphrase.

### 21.1.9. Additional resources

- **sshd(8)**, **ssh(1)**, **scp(1)**, **sftp(1)**, **ssh-keygen(1)**, **ssh-copy-id(1)**, **ssh\_config(5)**, **sshd\_config(5)**, **update-crypto-policies(8)**, and **crypto-policies(7)** man pages.
- [OpenSSH Home Page](#)
- [Configuring SELinux for applications and services with non-standard configurations](#)
- [Controlling network traffic using firewalld](#)

## 21.2. 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.

### 21.2.1. 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. |

| Protocol version | Usage recommendation                                                                                                                                                                                                                                                                      |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 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. In RHEL 8, 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. In RHEL 8, 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](#) .

### 21.2.2. 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.



## WARNING

Algorithms disabled for the **LEGACY** cryptographic policy do not conform to Red Hat's vision of RHEL 8 security, and their security properties are not reliable. Consider moving away from using these algorithms instead of re-enabling them. If you do decide to re-enable them, for example for interoperability with old hardware, treat them as insecure and apply extra protection measures, such as isolating their network interactions to separate network segments. Do not use them across public networks.

If you decide to not follow RHEL system-wide crypto policies or create custom cryptographic policies tailored to your setup, use the following recommendations for preferred protocols, cipher suites, and key lengths on your custom configuration:

### 21.2.2.1. 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 even though 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 the Apache web server.

### 21.2.2.2. 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, cipher 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 prefer 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 prefer AEAD ciphers, such as AES-GCM, over 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 a 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).

### 21.2.2.3. 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.

### 21.2.3. Hardening TLS configuration in applications

In RHEL, [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 ensure that your server application enforces *server-side cipher order*, so that the cipher suite to be used is determined by the order you configure.

#### 21.2.3.1. Configuring the Apache HTTP server to use TLS

The **Apache HTTP Server** can use both **OpenSSL** and **NSS** libraries for its TLS needs. RHEL 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 the `/usr/share/httpd/manual/mod/mod_ssl.html` file. Examples of various settings are described in the `/usr/share/httpd/manual/ssl/ssl_howto.html` file.

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
```

See the [Configuring TLS encryption on an Apache HTTP Server](#) chapter in the [Deploying different types of servers](#) document for more information.

### 21.2.3.2. Configuring the Nginx HTTP and proxy server to use TLS

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
    ...
}
```

See the [Adding TLS encryption to an Nginx web server](#) chapter in the [Deploying different types of servers](#) document for more information.

### 21.2.3.3. Configuring the Dovecot mail server to use TLS

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

- [Deploying different types of servers on RHEL 8](#)
- **config(5)** and **ciphers(1)** man pages.
- [Recommendations for Secure Use of Transport Layer Security \(TLS\) and Datagram Transport Layer Security \(DTLS\)](#).
- [Mozilla SSL Configuration Generator](#).
- [SSL Server Test](#).

## 21.3. CONFIGURING A VPN WITH IPSEC

In RHEL 8, a virtual private network (VPN) can be configured using the **IPsec** protocol, which is supported by the **Libreswan** application.

### 21.3.1. Libreswan as an IPsec VPN implementation

In RHEL, 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 Layer 2 Tunneling Protocol (L2TP) is usually called an L2TP/IPsec VPN, which requires the **xl2tpd** package provided by the **optional** repository.

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 RHEL. Do not use any other VPN technology without understanding the risks of doing so.

In RHEL, 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.

The **leftid** and **rightid** options serve as identification of the respective hosts in the authentication process. See the **ipsec.conf(5)** man page for more information.

### 21.3.2. Authentication methods in Libreswan

Libreswan supports several authentication methods, each of which fits a different scenario.

#### Pre-Shared key (PSK)

*Pre-Shared Key* (PSK) is the simplest authentication method. For security reasons, do not use PSKs shorter than 64 random characters. In FIPS mode, PSKs must comply with a minimum-strength requirement depending on the integrity algorithm used. You can set PSK by using the **authby=secret** connection.

#### Raw RSA keys

*Raw RSA keys* are commonly used for static host-to-host or subnet-to-subnet IPsec configurations. Each host is manually configured with the public RSA keys of all other hosts, and Libreswan sets up an IPsec tunnel between each pair of hosts. This method does not scale well for large numbers of hosts.

You can generate a raw RSA key on a host using the **ipsec newhostkey** command. You can list generated keys by using the **ipsec showhostkey** command. The **leftrsasigkey=** line is required for connection configurations that use CKA ID keys. Use the **authby=rsasig** connection option for raw RSA keys.

#### X.509 certificates

*X.509 certificates* are commonly used for large-scale deployments with hosts that connect to a common IPsec gateway. A central *certificate authority* (CA) signs RSA certificates for hosts or users. This central CA is responsible for relaying trust, including the revocations of individual hosts or users.

For example, you can generate X.509 certificates using the **openssl** command and the NSS **certutil** command. Because Libreswan reads user certificates from the NSS database using the certificates' nickname in the **leftcert=** configuration option, provide a nickname when you create a certificate.

If you use a custom CA certificate, you must import it to the Network Security Services (NSS) database. You can import any certificate in the PKCS #12 format to the Libreswan NSS database by using the **ipsec import** command.



## WARNING

Libreswan requires an Internet Key Exchange (IKE) peer ID as a subject alternative name (SAN) for every peer certificate as described in [section 3.1 of RFC 4945](#). Disabling this check by changing the **require-id-on-certificated=** option can make the system vulnerable to man-in-the-middle attacks.

Use the **authby=rsasig** connection option for authentication based on X.509 certificates using RSA with SHA-1 and SHA-2. You can further limit it for ECDSA digital signatures using SHA-2 by setting **authby=** to **ecdscsa** and RSA Probabilistic Signature Scheme (RSASSA-PSS) digital signatures based authentication with SHA-2 through **authby=rsa-sha2**. The default value is **authby=rsasig,ecdscsa**.

The certificates and the **authby=** signature methods should match. This increases interoperability and preserves authentication in one digital-signature system.

## NULL authentication

*NULL authentication* is used to gain mesh encryption without authentication. It protects against passive attacks but not against active attacks. However, because IKEv2 allows asymmetric authentication methods, NULL authentication can also be used for internet-scale opportunistic IPsec. In this model, clients authenticate the server, but servers do not authenticate the client. This model is similar to secure websites using TLS. Use **authby=null** for NULL authentication.

## Protection against quantum computers

In addition to the previously mentioned authentication methods, you can use the *Post-quantum Pre-shared Key* (PPK) method to protect against possible attacks by quantum computers. Individual clients or groups of clients can use their own PPK by specifying a PPK ID that corresponds to an out-of-band configured pre-shared key.

Using IKEv1 with pre-shared keys provides protection against quantum attackers. The redesign of IKEv2 does not offer this protection natively. Libreswan offers the use of *Post-quantum Pre-shared Key* (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 based on dictionary words. The PPK ID and PPK data are stored in **ipsec.secrets**, for example:

```
@west @east : PPKS "user1" "thestringismeanttobearandomstr"
```

The **PPKS** option refers to static PPKs. This experimental function uses one-time-pad-based Dynamic PPKs. Upon each connection, a new part of the one-time pad is used as the PPK. When used, that part of the dynamic PPK inside the file is overwritten with zeros 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 an unsupported Technology Preview. Use with caution.

### 21.3.3. 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 and create a new database:

```
# systemctl stop ipsec
# rm /etc/ipsec.d/*db
# ipsec initnss
```

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
```

### 21.3.4. Creating a host-to-host VPN

To configure [application]Libreswan to create a host-to-host IPsec VPN between two hosts referred to as *left* and *right* using authentication by raw RSA keys, enter the following commands on both of the hosts:

#### Prerequisites

- Libreswan is installed and the **ipsec** service is started on each node.

#### Procedure

1. Generate a raw RSA key pair on each host:

```
# ipsec newhostkey
```

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. After importing keys, restart the **ipsec** service:

```
# systemctl restart ipsec
```

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
```

### 21.3.5. 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
    leftrsasigkey=0sAQOrlo+hOafUZDICQmXFrje/oZm [...] W2n417C/4urYHQkCvulQ==
    rightid=@east
    right=192.1.2.45
    rightrsasigkey=0sAQO3fwC6nSSGgt64DWiYZzuHbc4 [...] D/v8t5YTQ==
    authby=rsasig
```

### 21.3.6. Configuring a remote access VPN

Road warriors are traveling users with mobile clients and a dynamically assigned IP address. The mobile clients authenticate using X.509 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
leftauthserver=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"
rightauthclient=yes
rightmodecfgclient=yes
authby=rsasig
# optionally, run the client X.509 ID through pam to allow or deny client
# pam-authorize=yes
# load connection, do not 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

```

### 21.3.7. 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
```

### 21.3.8. 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:

```
# fips-mode-setup --enable
```

6. Restart your system to allow the kernel to switch to FIPS mode:

```
# reboot
```

## Verification

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
```

- 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
```

- 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
```

- 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](#).

### 21.3.9. Protecting the IPsec NSS database by a password

By default, the IPsec service creates its Network Security Services (NSS) database with an empty password during the first start. Add password protection by using the following steps.



#### NOTE

In the previous releases of RHEL up to version 6.6, you had to protect the IPsec NSS database with a password to meet the FIPS 140-2 requirements because the NSS cryptographic libraries were certified for the FIPS 140-2 Level 2 standard. In RHEL 8, NIST certified NSS to Level 1 of this standard, and this status does not require password protection for the database.

### Prerequisites

- The `/etc/ipsec.d/` directory contains NSS database files.

### Procedure

1. Enable password protection for the **NSS** database for Libreswan:

```

# certutil -N -d sql:/etc/ipsec.d
Enter Password or Pin for "NSS Certificate DB":
Enter a password which will be used to encrypt your keys.
The password should be at least 8 characters long,
and should contain at least one non-alphabetic character.

Enter new password:

```

2. Create the `/etc/ipsec.d/nsspassword` file containing the password you have set in the previous step, for example:

```

# cat /etc/ipsec.d/nsspassword
NSS Certificate DB:MyStrongPasswordHere

```

Note that the `nsspassword` file use the following syntax:

```

token_1_name:the_password
token_2_name:the_password

```

The default NSS software token is **NSS Certificate DB**. If your system is running in FIPS mode, the name of the token is **NSS FIPS 140-2 Certificate DB**.

3. Depending on your scenario, either start or restart the **ipsec** service after you finish the **nsspassword** file:

```
# systemctl restart ipsec
```

## Verification

1. Check that the **ipsec** service is running after you have added a non-empty password to its NSS database:

```
# systemctl status ipsec
● ipsec.service - Internet Key Exchange (IKE) Protocol Daemon for IPsec
  Loaded: loaded (/usr/lib/systemd/system/ipsec.service; enabled; vendor preset: disable>
  Active: active (running)...
```

2. Optionally, check that the **Journal** log contains entries confirming a successful initialization:

```
# journalctl -u ipsec
...
pluto[6214]: Initializing NSS using read-write database "sql:/etc/ipsec.d"
pluto[6214]: NSS Password from file "/etc/ipsec.d/nsspassword" for token "NSS Certificate
DB" with length 20 passed to NSS
pluto[6214]: NSS crypto library initialized
...
```

## Additional resources

- [certutil\(1\)](#) man page.
- [Government Standards](#) Knowledgebase article.

### 21.3.10. Configuring an IPsec VPN to use TCP

Libreswan supports TCP encapsulation of IKE and IPsec packets as described in RFC 8229. With this feature, you can establish IPsec VPNs on networks that prevent traffic transmitted via UDP and Encapsulating Security Payload (ESP). You can configure VPN servers and clients to use TCP either as a fallback or as the main VPN transport protocol. Because TCP encapsulation has bigger performance costs, use TCP as the main VPN protocol only if UDP is permanently blocked in your scenario.

## Prerequisites

- A [remote-access VPN](#) is already configured.

## Procedure

1. Add the following option to the **/etc/ipsec.conf** file in the **config setup** section:

```
listen-tcp=yes
```

- To use TCP encapsulation as a fallback option when the first attempt over UDP fails, add the following two options to the client's connection definition:

```
enable-tcp=fallback
tcp-remoteport=4500
```

Alternatively, if you know that UDP is permanently blocked, use the following options in the client's connection configuration:

```
enable-tcp=yes
tcp-remoteport=4500
```

### Additional resources

- [IETF RFC 8229: TCP Encapsulation of IKE and IPsec Packets](#) .

### 21.3.11. Configuring automatic detection and usage of ESP hardware offload to accelerate an IPsec connection

Offloading Encapsulating Security Payload (ESP) to the hardware accelerates IPsec connections over Ethernet. By default, Libreswan detects if hardware supports this feature and, as a result, enables ESP hardware offload. In case that the feature was disabled or explicitly enabled, you can switch back to automatic detection.

#### Prerequisites

- The network card supports ESP hardware offload.
- The network driver supports ESP hardware offload.
- The IPsec connection is configured and works.

#### Procedure

- Edit the Libreswan configuration file in the `/etc/ipsec.d/` directory of the connection that should use automatic detection of ESP hardware offload support.
- Ensure the **nic-offload** parameter is not set in the connection's settings.
- If you removed **nic-offload**, restart the **ipsec** service:

```
# systemctl restart ipsec
```

#### Verification

If the network card supports ESP hardware offload support, following these steps to verify the result:

- Display the **tx\_ipsec** and **rx\_ipsec** counters of the Ethernet device the IPsec connection uses:

```
# ethtool -S enp1s0 | egrep "_ipsec"
tx_ipsec: 10
rx_ipsec: 10
```

- Send traffic through the IPsec tunnel. For example, ping a remote IP address:

```
# ping -c 5 remote_ip_address
```

3. Display the **tx\_ipsec** and **rx\_ipsec** counters of the Ethernet device again:

```
# ethtool -S enp1s0 | egrep "_ipsec"  
tx_ipsec: 15  
rx_ipsec: 15
```

If the counter values have increased, ESP hardware offload works.

### Additional resources

- [Configuring a VPN with IPsec](#)

## 21.3.12. Configuring ESP hardware offload on a bond to accelerate an IPsec connection

Offloading Encapsulating Security Payload (ESP) to the hardware accelerates IPsec connections. If you use a network bond for fail-over reasons, the requirements and the procedure to configure ESP hardware offload are different from those using a regular Ethernet device. For example, in this scenario, you enable the offload support on the bond, and the kernel applies the settings to the ports of the bond.

### Prerequisites

- All network cards in the bond support ESP hardware offload.
- The network driver supports ESP hardware offload on a bond device. In RHEL, only the **ixgbe** driver supports this feature.
- The bond is configured and works.
- The bond uses the **active-backup** mode. The bonding driver does not support any other modes for this feature.
- The IPsec connection is configured and works.

### Procedure

1. Enable ESP hardware offload support on the network bond:

```
# nmcli connection modify bond0 ethtool.feature-esp-hw-offload on
```

This command enables ESP hardware offload support on the **bond0** connection.

2. Reactivate the **bond0** connection:

```
# nmcli connection up bond0
```

3. Edit the Libreswan configuration file in the **/etc/ipsec.d/** directory of the connection that should use ESP hardware offload, and append the **nic-offload=yes** statement to the connection entry:

```
conn example  
...  
nic-offload=yes
```

- 4. Restart the **ipsec** service:

```
# systemctl restart ipsec
```

### Verification

1. Display the active port of the bond:

```
# grep "Currently Active Slave" /proc/net/bonding/bond0
Currently Active Slave: enp1s0
```

2. Display the **tx\_ipsec** and **rx\_ipsec** counters of the active port:

```
# ethtool -S enp1s0 | egrep "_ipsec"
tx_ipsec: 10
rx_ipsec: 10
```

3. Send traffic through the IPsec tunnel. For example, ping a remote IP address:

```
# ping -c 5 remote_ip_address
```

4. Display the **tx\_ipsec** and **rx\_ipsec** counters of the active port again:

```
# ethtool -S enp1s0 | egrep "_ipsec"
tx_ipsec: 15
rx_ipsec: 15
```

If the counter values have increased, ESP hardware offload works.

### Additional resources

- [Configuring network bonding](#)
- [Configuring a VPN with IPsec](#) section in the Securing networks document

## 21.3.13. Configuring IPsec connections that opt out of the system-wide crypto policies

### Overriding system-wide crypto-policies for a connection

The RHEL system-wide cryptographic policies create a special connection called **%default**. This connection contains the default values for the **ikev2**, **esp**, and **ike** options. However, you can override the default values by specifying the mentioned option in the connection configuration file.

For example, the following configuration allows connections that use IKEv1 with AES and SHA-1 or SHA-2, and IPsec (ESP) with either AES-GCM or AES-CBC:

```
conn MyExample
...
ikev2=never
```

```
ike=aes-sha2,aes-sha1;modp2048
esp=aes_gcm,aes-sha2,aes-sha1
...
```

Note that AES-GCM is available for IPsec (ESP) and for IKEv2, but not for IKEv1.

### Disabling system-wide crypto policies for all connections

To disable system-wide crypto policies for all IPsec connections, 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 file.

### Additional resources

- [Using system-wide cryptographic policies](#).

## 21.3.14. Troubleshooting IPsec VPN configurations

Problems related to IPsec VPN configurations most commonly occur due to several main reasons. If you are encountering such problems, you can check if the cause of the problem corresponds to any of the following scenarios, and apply the corresponding solution.

### Basic connection troubleshooting

Most problems with VPN connections occur in new deployments, where administrators configured endpoints with mismatched configuration options. Also, a working configuration can suddenly stop working, often due to newly introduced incompatible values. This could be the result of an administrator changing the configuration. Alternatively, an administrator may have installed a firmware update or a package update with different default values for certain options, such as encryption algorithms.

To confirm that an IPsec VPN connection is established:

```
# ipsec trafficstatus
006 #8: "vpn.example.com"[1] 192.0.2.1, type=ESP, add_time=1595296930, inBytes=5999,
outBytes=3231, id='@vpn.example.com', lease=100.64.13.5/32
```

If the output is empty or does not show an entry with the connection name, the tunnel is broken.

To check that the problem is in the connection:

1. Reload the `vpn.example.com` connection:

```
# ipsec auto --add vpn.example.com
002 added connection description "vpn.example.com"
```

2. Next, initiate the VPN connection:

```
# ipsec auto --up vpn.example.com
```

### Firewall-related problems

The most common problem is that a firewall on one of the IPsec endpoints or on a router between the endpoints is dropping all Internet Key Exchange (IKE) packets.

- For IKEv2, an output similar to the following example indicates a problem with a firewall:

```
# ipsec auto --up vpn.example.com
181 "vpn.example.com"[1] 192.0.2.2 #15: initiating IKEv2 IKE SA
181 "vpn.example.com"[1] 192.0.2.2 #15: STATE_PARENT_I1: sent v2I1, expected v2R1
010 "vpn.example.com"[1] 192.0.2.2 #15: STATE_PARENT_I1: retransmission; will wait 0.5
seconds for response
010 "vpn.example.com"[1] 192.0.2.2 #15: STATE_PARENT_I1: retransmission; will wait 1
seconds for response
010 "vpn.example.com"[1] 192.0.2.2 #15: STATE_PARENT_I1: retransmission; will wait 2
seconds for
...
```

- For IKEv1, the output of the initiating command looks like:

```
# ipsec auto --up vpn.example.com
002 "vpn.example.com" #9: initiating Main Mode
102 "vpn.example.com" #9: STATE_MAIN_I1: sent MI1, expecting MR1
010 "vpn.example.com" #9: STATE_MAIN_I1: retransmission; will wait 0.5 seconds for
response
010 "vpn.example.com" #9: STATE_MAIN_I1: retransmission; will wait 1 seconds for
response
010 "vpn.example.com" #9: STATE_MAIN_I1: retransmission; will wait 2 seconds for
response
...
```

Because the IKE protocol, which is used to set up IPsec, is encrypted, you can troubleshoot only a limited subset of problems using the **tcpdump** tool. If a firewall is dropping IKE or IPsec packets, you can try to find the cause using the **tcpdump** utility. However, **tcpdump** cannot diagnose other problems with IPsec VPN connections.

- To capture the negotiation of the VPN and all encrypted data on the **eth0** interface:

```
# tcpdump -i eth0 -n -n esp or udp port 500 or udp port 4500 or tcp port 4500
```

## Mismatched algorithms, protocols, and policies

VPN connections require that the endpoints have matching IKE algorithms, IPsec algorithms, and IP address ranges. If a mismatch occurs, the connection fails. If you identify a mismatch by using one of the following methods, fix it by aligning algorithms, protocols, or policies.

- If the remote endpoint is not running IKE/IPsec, you can see an ICMP packet indicating it. For example:

```
# ipsec auto --up vpn.example.com
...
000 "vpn.example.com"[1] 192.0.2.2 #16: ERROR: asynchronous network error report on
wlp2s0 (192.0.2.2:500), complainant 198.51.100.1: Connection refused [errno 111, origin
ICMP type 3 code 3 (not authenticated)]
...
```

- Example of mismatched IKE algorithms:

```
# ipsec auto --up vpn.example.com
...
003 "vpn.example.com"[1] 193.110.157.148 #3: dropping unexpected IKE_SA_INIT message
containing NO_PROPOSAL_CHOSEN notification; message payloads: N; missing payloads:
SA,KE,Ni
```

- Example of mismatched IPsec algorithms:

```
# ipsec auto --up vpn.example.com
...
182 "vpn.example.com"[1] 193.110.157.148 #5: STATE_PARENT_I2: sent v2I2, expected
v2R2 {auth=IKEv2 cipher=AES_GCM_16_256 integ=n/a prf=HMAC_SHA2_256
group=MODP2048}
002 "vpn.example.com"[1] 193.110.157.148 #6: IKE_AUTH response contained the error
notification NO_PROPOSAL_CHOSEN
```

A mismatched IKE version could also result in the remote endpoint dropping the request without a response. This looks identical to a firewall dropping all IKE packets.

- Example of mismatched IP address ranges for IKEv2 (called Traffic Selectors - TS):

```
# ipsec auto --up vpn.example.com
...
1v2 "vpn.example.com" #1: STATE_PARENT_I2: sent v2I2, expected v2R2 {auth=IKEv2
cipher=AES_GCM_16_256 integ=n/a prf=HMAC_SHA2_512 group=MODP2048}
002 "vpn.example.com" #2: IKE_AUTH response contained the error notification
TS_UNACCEPTABLE
```

- Example of mismatched IP address ranges for IKEv1:

```
# ipsec auto --up vpn.example.com
...
031 "vpn.example.com" #2: STATE_QUICK_I1: 60 second timeout exceeded after 0
retransmits. No acceptable response to our first Quick Mode message: perhaps peer likes
no proposal
```

- When using PreSharedKeys (PSK) in IKEv1, if both sides do not put in the same PSK, the entire IKE message becomes unreadable:

```
# ipsec auto --up vpn.example.com
...
003 "vpn.example.com" #1: received Hash Payload does not match computed value
223 "vpn.example.com" #1: sending notification INVALID_HASH_INFORMATION to
192.0.2.23:500
```

- In IKEv2, the mismatched-PSK error results in an AUTHENTICATION\_FAILED message:

```
# ipsec auto --up vpn.example.com
...
002 "vpn.example.com" #1: IKE SA authentication request rejected by peer:
AUTHENTICATION_FAILED
```

## Maximum transmission unit



Other than firewalls blocking IKE or IPsec packets, the most common cause of networking problems relates to an increased packet size of encrypted packets. Network hardware fragments packets larger than the maximum transmission unit (MTU), for example, 1500 bytes. Often, the fragments are lost and the packets fail to re-assemble. This leads to intermittent failures, when a ping test, which uses small-sized packets, works but other traffic fails. In this case, you can establish an SSH session but the terminal freezes as soon as you use it, for example, by entering the 'ls -al /usr' command on the remote host.

To work around the problem, reduce MTU size by adding the **mtu=1400** option to the tunnel configuration file.

Alternatively, for TCP connections, enable an iptables rule that changes the MSS value:

```
# iptables -I FORWARD -p tcp --tcp-flags SYN,RST SYN -j TCPMSS --clamp-mss-to-pmtu
```

If the previous command does not solve the problem in your scenario, directly specify a lower size in the **set-mss** parameter:

```
# iptables -I FORWARD -p tcp --tcp-flags SYN,RST SYN -j TCPMSS --set-mss 1380
```

## Network address translation (NAT)

When an IPsec host also serves as a NAT router, it could accidentally remap packets. The following example configuration demonstrates the problem:

```
conn myvpn
  left=172.16.0.1
  leftsubnet=10.0.2.0/24
  right=172.16.0.2
  rightsubnet=192.168.0.0/16
  ...
```

The system with address 172.16.0.1 have a NAT rule:

```
iptables -t nat -I POSTROUTING -o eth0 -j MASQUERADE
```

If the system on address 10.0.2.33 sends a packet to 192.168.0.1, then the router translates the source 10.0.2.33 to 172.16.0.1 before it applies the IPsec encryption.

Then, the packet with the source address 10.0.2.33 no longer matches the **conn myvpn** configuration, and IPsec does not encrypt this packet.

To solve this problem, insert rules that exclude NAT for target IPsec subnet ranges on the router, in this example:

```
iptables -t nat -I POSTROUTING -s 10.0.2.0/24 -d 192.168.0.0/16 -j RETURN
```

## Kernel IPsec subsystem bugs

The kernel IPsec subsystem might fail, for example, when a bug causes a desynchronizing of the IKE user space and the IPsec kernel. To check for such problems:

```
$ cat /proc/net/xfrm_stat
XfrmInError      0
XfrmInBufferError 0
```

...

Any non-zero value in the output of the previous command indicates a problem. If you encounter this problem, open a new [support case](#), and attach the output of the previous command along with the corresponding IKE logs.

### Libreswan logs

Libreswan logs using the **syslog** protocol by default. You can use the **journalctl** command to find log entries related to IPsec. Because the corresponding entries to the log are sent by the **pluto** IKE daemon, search for the “pluto” keyword, for example:

```
$ journalctl -b | grep pluto
```

To show a live log for the **ipsec** service:

```
$ journalctl -f -u ipsec
```

If the default level of logging does not reveal your configuration problem, enable debug logs by adding the **plutodebug=all** option to the **config setup** section in the **/etc/ipsec.conf** file.

Note that debug logging produces a lot of entries, and it is possible that either the **journald** or **syslogd** service rate-limits the **syslog** messages. To ensure you have complete logs, redirect the logging to a file. Edit the **/etc/ipsec.conf**, and add the **logfile=/var/log/pluto.log** in the **config setup** section.

### Additional resources

- [Troubleshooting problems using log files](#).
- **tcpdump(8)** and **ipsec.conf(5)** man pages.
- [Using and configuring firewalld](#)

### 21.3.15. Additional resources

- **ipsec(8)**, **ipsec.conf(5)**, **ipsec.secrets(5)**, **ipsec\_auto(8)**, and **ipsec\_rsasigkey(8)** man pages.
- **/usr/share/doc/libreswan-version/** directory.
- [The website of the upstream project](#).
- [The Libreswan Project Wiki](#).
- [All Libreswan man pages](#).
- [NIST Special Publication 800-77: Guide to IPsec VPNs](#).

## 21.4. USING MACSEC TO ENCRYPT LAYER-2 TRAFFIC IN THE SAME PHYSICAL NETWORK

You can use MACsec to secure the communication between two devices (point-to-point). For example, your branch office is connected over a Metro-Ethernet connection with the central office, you can configure MACsec on the two hosts that connect the offices to increase the security.

Media Access Control security (MACsec) is a layer 2 protocol that secures different traffic types over the Ethernet links including:

- dynamic host configuration protocol (DHCP)
- address resolution protocol (ARP)
- Internet Protocol version 4 / 6 (**IPv4 / IPv6**) and
- any traffic over IP such as TCP or UDP

MACsec encrypts and authenticates all traffic in LANs, by default with the GCM-AES-128 algorithm, and uses a pre-shared key to establish the connection between the participant hosts. If you want to change the pre-shared key, you need to update the NM configuration on all hosts in the network that uses MACsec.

A MACsec connection uses an Ethernet device, such as an Ethernet network card, VLAN, or tunnel device, as parent. You can either set an IP configuration only on the MACsec device to communicate with other hosts only using the encrypted connection, or you can also set an IP configuration on the parent device. In the latter case, you can use the parent device to communicate with other hosts using an unencrypted connection and the MACsec device for encrypted connections.

MACsec does not require any special hardware. For example, you can use any switch, except if you want to encrypt traffic only between a host and a switch. In this scenario, the switch must also support MACsec.

In other words, there are 2 common methods to configure MACsec;

- host to host and
- host to switch then switch to other host(s)



### IMPORTANT

You can use MACsec only between hosts that are in the same (physical or virtual) LAN.

#### 21.4.1. Configuring a MACsec connection using nmcli

You can configure Ethernet interfaces to use MACsec using the **nmcli** utility. For example, you can create a MACsec connection between two hosts that are connected over Ethernet.

##### Procedure

1. On the first host on which you configure MACsec:
  - Create the connectivity association key (CAK) and connectivity-association key name (CKN) for the pre-shared key:
    - a. Create a 16-byte hexadecimal CAK:

```
# dd if=/dev/urandom count=16 bs=1 2> /dev/null | hexdump -e '1/2 "%04x"'
50b71a8ef0bd5751ea76de6d6c98c03a
```

- b. Create a 32-byte hexadecimal CKN:

```
# dd if=/dev/urandom count=32 bs=1 2> /dev/null | hexdump -e '1/2 "%04x"'
f2b4297d39da7330910a74abc0449feb45b5c0b9fc23df1430e1898fcf1c4550
```

2. On both hosts you want to connect over a MACsec connection:
3. Create the MACsec connection:

```
# nmcli connection add type macsec con-name macsec0 ifname macsec0
connection.autoconnect yes macsec.parent enp1s0 macsec.mode psk macsec.mka-
cak 50b71a8ef0bd5751ea76de6d6c98c03a macsec.mka-ckn
f2b4297d39da7330910a74abc0449feb45b5c0b9fc23df1430e1898fcf1c4550
```

Use the CAK and CKN generated in the previous step in the **macsec.mka-cak** and **macsec.mka-ckn** parameters. The values must be the same on every host in the MACsec-protected network.

4. Configure the IP settings on the MACsec connection.
  - a. Configure the **IPv4** settings. For example, to set a static **IPv4** address, network mask, default gateway, and DNS server to the **macsec0** connection, enter:

```
# nmcli connection modify macsec0 ipv4.method manual ipv4.addresses
'192.0.2.1/24' ipv4.gateway '192.0.2.254' ipv4.dns '192.0.2.253'
```

- b. Configure the **IPv6** settings. For example, to set a static **IPv6** address, network mask, default gateway, and DNS server to the **macsec0** connection, enter:

```
# nmcli connection modify macsec0 ipv6.method manual ipv6.addresses
'2001:db8:1::1/32' ipv6.gateway '2001:db8:1::fffe' ipv6.dns '2001:db8:1::fffd'
```

5. Activate the connection:

```
# nmcli connection up macsec0
```

## Verification

1. Verify that the traffic is encrypted:

```
# tcpdump -nn -i enp1s0
```

2. Optional: Display the unencrypted traffic:

```
# tcpdump -nn -i macsec0
```

3. Display MACsec statistics:

```
# ip macsec show
```

4. Display individual counters for each type of protection: integrity-only (encrypt off) and encryption (encrypt on)

```
# ip -s macsec show
```

## 21.4.2. Additional resources

- [MACsec: a different solution to encrypt network traffic](#) blog.

## 21.5. 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.

**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.

Note that **firewalld** with **nftables** backend does not support passing custom **nftables** rules to **firewalld**, using the **--direct** option.

### 21.5.1. Getting started with firewalld

The following is an introduction to **firewalld** features, such as services and zones, and how to manage the **firewalld** systemd service.

#### 21.5.1.1. 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 for simple firewall use cases. The utility is easy to use and covers the typical use cases for these scenarios.
- **nftables**: Use the **nftables** utility to set up complex and performance-critical firewalls, such as for a whole network.
- **iptables**: The **iptables** utility on Red Hat Enterprise Linux uses the **nf\_tables** kernel API instead of the **legacy** back end. The **nf\_tables** API provides backward compatibility so that scripts that use **iptables** commands still work on Red Hat Enterprise Linux. For new firewall scripts, Red Hat recommends to use **nftables**.



#### IMPORTANT

To prevent the different firewall services from influencing each other, run only one of them on a RHEL host, and disable the other services.

#### 21.5.1.2. 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

- The **firewalld.zone(5)** man page.

### 21.5.1.3. 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

- The **firewalld.service(5)** man page

### 21.5.1.4. 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
```

### 21.5.1.5. 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
```

### 21.5.1.6. 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. You can use the **firewall-cmd** utility to verify the configuration.

#### 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': 'tcp' not from {'tcp'|'udp'|'sctp'|'dccp'}
```

## 21.5.2. Viewing the current status and settings of firewalld

To monitor the **firewalld** service, you can display the status, allowed services, and settings.

### 21.5.2.1. 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
```



- 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
```

### 21.5.2.2. 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.

You can start the graphical firewall configuration tool using the command-line.

#### Prerequisites

- You installed the **firewall-config** package.

#### Procedure

- To start the graphical firewall configuration tool using the command-line:

```
$ 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.

### 21.5.2.3. 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.

#### Procedure

- 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 dhcpv6-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 dhcpv6-client
...
```

- 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
```

- To see which services are allowed in the current zone:

```
# firewall-cmd --list-services
ssh dhcpv6-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.

### 21.5.3. Controlling network traffic using firewalld

The **firewalld** package installs a large number of predefined service files and you can add more or customize them. You can then use these service definitions to open or close ports for services without knowing the protocol and port numbers they use.

#### 21.5.3.1. 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. For 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, enter:

```
# firewall-cmd --panic-off
```

### Verification

- To see whether panic mode is switched on or off, use:

```
# firewall-cmd --query-panic
```

### 21.5.3.2. 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

- Check that the service is not already allowed:

```
# firewall-cmd --list-services
ssh dhcpv6-client
```

- 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 dhcpv6
dhcpv6-client dns docker-registry ...
```

- Add the service to the allowed services:

```
# firewall-cmd --add-service=<service_name>
```

- Make the new settings persistent:

```
# firewall-cmd --runtime-to-permanent
```

### 21.5.3.3. Controlling traffic with predefined services using GUI

You can control the network traffic with predefined services using graphical user interface.

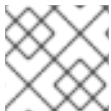
#### Prerequisites

- You installed the **firewall-config** package

## Procedure

1. To enable or disable a predefined or custom service:
  - a. Start the **firewall-config** tool and select the network zone whose services are to be configured.
  - b. Select the **Zones** tab and then the **Services** tab below.
  - c. Select the check box for each type of service you want to trust or clear the check box to block a service in the selected zone.
2. To edit a service:
  - a. Start the **firewall-config** tool.
  - b. Select **Permanent** from the menu labeled **Configuration**. Additional icons and menu buttons appear at the bottom of the **Services** window.
  - c. 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 the **Runtime** mode.

### 21.5.3.4. 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_xml_file> --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.

### 21.5.3.5. Opening ports using GUI

To permit traffic through the firewall to a certain port, you can open the port in the GUI.

#### Prerequisites

- You installed the **firewall-config** package

#### Procedure

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.

### 21.5.3.6. Controlling traffic with protocols using GUI

To permit traffic through the firewall using a certain protocol, you can use the GUI.

#### Prerequisites

- You installed the **firewall-config** package

#### Procedure

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.

### 21.5.3.7. Opening source ports using GUI

To permit traffic through the firewall from a certain port, you can use the GUI.

### Prerequisites

- You installed the **firewall-config** package

### Procedure

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.

## 21.5.4. 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.

### 21.5.4.1. 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
```

The port types are either **tcp**, **udp**, **sctp**, or **dccp**. The type must match the type of network communication.

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.

### 21.5.4.2. 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
```

## 21.5.5. 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.

### 21.5.5.1. Listing zones

You can list zones using the command line.

#### 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
```

### 21.5.5.2. Modifying firewalld settings for a certain zone

The [Controlling traffic with predefined services using cli](#) and [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

- 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
```

### 21.5.5.3. 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

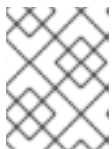
To set up the default zone:

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.

### 21.5.5.4. 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
```

#### 21.5.5.5. Assigning a zone to a connection using nmcli

You can add a **firewalld** zone to a **NetworkManager** connection using the **nmcli** utility.

##### Procedure

1. Assign the zone to the **NetworkManager** connection profile:

```
# nmcli connection modify profile connection.zone zone_name
```

2. Activate the connection:

```
# nmcli connection up profile
```

#### 21.5.5.6. 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

- 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
```

#### 21.5.5.7. 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

1. Create a new zone:

```
# firewall-cmd --permanent --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
```

### 21.5.5.8. 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 **firewalld** 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/firewalld/zones/** and **/etc/firewalld/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 protocol="udp" port="1025-65535"/>
  <port protocol="tcp" port="1025-65535"/>
</zone>
```

To change settings for that zone, add or remove sections to add ports, forward ports, services, and so on.

#### Additional resources

- **firewalld.zone** manual page

### 21.5.5.9. 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 behavior is defined by setting the target of the zone. There are four options:

- **ACCEPT**: Accepts all incoming packets except those disallowed by specific rules.
- **REJECT**: Rejects all incoming packets except those allowed by specific rules. When **firewalld** rejects packets, the source machine is informed about the rejection.
- **DROP**: Drops all incoming packets except those allowed by specific rules. When **firewalld** drops packets, the source machine is not informed about the packet drop.
- **default**: Similar behavior as for **REJECT**, but with special meanings in certain scenarios. For details, see the **Options to Adapt and Query Zones and Policies** section in the **firewall-cmd(1)** man page.

#### Procedure

To set a target for a zone:

1. List the information for the specific zone to see the default target:

```
# firewall-cmd --zone=zone-name --list-all
```

2. Set a new target in the zone:

```
# firewall-cmd --permanent --zone=zone-name --set-target=
<default|ACCEPT|REJECT|DROP>
```

### Additional resources

- **firewall-cmd(1)** man page

## 21.5.6. 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.

### 21.5.6.1. Adding a source

To route incoming traffic into a specific zone, 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.



#### NOTE

In case you add multiple zones with an overlapping network range, they are ordered alphanumerically by zone name and only the first one is considered.

- 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
```

### 21.5.6.2. 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
```

### 21.5.6.3. 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

- To add a source port:

```
# firewall-cmd --zone=zone-name --add-source-port=<port-name>/<tcp|udp|sctp|dccp>
```

### 21.5.6.4. Removing a source port

By removing a source port you disable sorting the traffic based on a port of origin.

#### Procedure

- To remove a source port:

```
# firewall-cmd --zone=zone-name --remove-source-port=<port-name>/<tcp|udp|sctp|dccp>
```

### 21.5.6.5. 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 only HTTP traffic from the **192.0.2.0/24** network while any other traffic is blocked.



## WARNING

When you configure this scenario, use a zone that has the **default** target. Using a zone that has the target set to **ACCEPT** is a security risk, because for traffic from **192.0.2.0/24**, all network connections would be accepted.

## Procedure

1. List all available zones:

```
# firewall-cmd --get-zones
block dmz drop external home internal public trusted work
```

2. Add the IP range to the **internal** zone to route the traffic originating from the source through the zone:

```
# firewall-cmd --zone=internal --add-source=192.0.2.0/24
```

3. Add the **http** service to the **internal** zone:

```
# firewall-cmd --zone=internal --add-service=http
```

4. Make the new settings persistent:

```
# firewall-cmd --runtime-to-permanent
```

## Verification

- Check that the **internal** zone is active and that the service is allowed in it:

```
# firewall-cmd --zone=internal --list-all
internal (active)
target: default
icmp-block-inversion: no
interfaces:
sources: 192.0.2.0/24
services: cockpit dhcpv6-client mdns samba-client ssh http
...
```

## Additional resources

- [firewalld.zones\(5\)](#) man page

## 21.5.7. Filtering forwarded traffic between zones

With a policy object, users can group different identities that require similar permissions in the policy. You can apply policies depending on the direction of the traffic.

The policy objects feature provides forward and output filtering in firewalld. You can use firewalld to filter traffic between different zones to allow access to locally hosted VMs to connect the host.

### 21.5.7.1. The relationship between policy objects and zones

Policy objects allow the user to attach firewalld's primitives' such as services, ports, and rich rules to the policy. You can apply the policy objects to traffic that passes between zones in a stateful and unidirectional manner.

```
# firewall-cmd --permanent --new-policy myOutputPolicy
# firewall-cmd --permanent --policy myOutputPolicy --add-ingress-zone HOST
# firewall-cmd --permanent --policy myOutputPolicy --add-egress-zone ANY
```

**HOST** and **ANY** are the symbolic zones used in the ingress and egress zone lists.

- The **HOST** symbolic zone allows policies for the traffic originating from or has a destination to the host running firewalld.
- The **ANY** symbolic zone applies policy to all the current and future zones. **ANY** symbolic zone acts as a wildcard for all zones.

### 21.5.7.2. Using priorities to sort policies

Multiple policies can apply to the same set of traffic, therefore, priorities should be used to create an order of precedence for the policies that may be applied.

To set a priority to sort the policies:

```
# firewall-cmd --permanent --policy mypolicy --set-priority -500
```

In the above example **-500** is a lower priority value but has higher precedence. Thus, **-500** will execute before **-100**. Higher priority values have precedence over lower values.

The following rules apply to policy priorities:

- Policies with negative priorities apply before rules in zones.
- Policies with positive priorities apply after rules in zones.
- Priority 0 is reserved and hence is unusable.

### 21.5.7.3. Using policy objects to filter traffic between locally hosted Containers and a network physically connected to the host

The policy objects feature allows users to filter their container and virtual machine traffic.

#### Procedure

1. Create a new policy.

```
# firewall-cmd --permanent --new-policy podmanToHost
```

- Block all traffic.

```
# firewall-cmd --permanent --policy podmanToHost --set-target REJECT
# firewall-cmd --permanent --policy podmanToHost --add-service dhcp
# firewall-cmd --permanent --policy podmanToHost --add-service dns
```



#### NOTE

Red Hat recommends that you block all traffic to the host by default and then selectively open the services you need for the host.

- Define the ingress zone to use with the policy.

```
# firewall-cmd --permanent --policy podmanToHost --add-ingress-zone podman
```

- Define the egress zone to use with the policy.

```
# firewall-cmd --permanent --policy podmanToHost --add-egress-zone ANY
```

#### Verification

- Verify information about the policy.

```
# firewall-cmd --info-policy podmanToHost
```

#### 21.5.7.4. Setting the default target of policy objects

You can specify `--set-target` options for policies. The following targets are available:

- ACCEPT** - accepts the packet
- DROP** - drops the unwanted packets
- REJECT** - rejects unwanted packets with an ICMP reply
- CONTINUE** (default) - packets will be subject to rules in following policies and zones.

```
# firewall-cmd --permanent --policy mypolicy --set-target CONTINUE
```

#### Verification

- Verify information about the policy

```
# firewall-cmd --info-policy mypolicy
```

#### 21.5.8. Configuring NAT using firewalld

With **firewalld**, you can configure the following network address translation (NAT) types:

- Masquerading

- Source NAT (SNAT)
- Destination NAT (DNAT)
- Redirect

### 21.5.8.1. NAT types

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 Service Providers do not route private IP ranges, such as **10.0.0.0/8**. If you use private 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.

Masquerading and SNAT are very similar to one another. 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 rewrite the destination address and port of incoming packets. For example, if your web server uses an IP address from a private 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.

#### Redirect

This type is a special case of DNAT that redirects packets to the local machine depending on the chain hook. For example, if a service runs on a different port than its standard port, you can redirect incoming traffic from the standard port to this specific port.

### 21.5.8.2. Configuring IP address masquerading

You can 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, pass the **--permanent** option to the command.



- To disable IP masquerading, enter the following command as **root**:

```
# firewall-cmd --zone=external --remove-masquerade
```

To make this setting permanent, pass the **--permanent** option to the command.

### 21.5.9. Using DNAT to forward HTTPS traffic to a different host

If your web server runs in a DMZ with private IP addresses, you can configure destination network address translation (DNAT) to enable clients on the internet to connect to this web server. In this case, the host name of the web server resolves to the public IP address of the router. When a client establishes a connection to a defined port on the router, the router forwards the packets to the internal web server.

#### Prerequisites

- The DNS server resolves the host name of the web server to the router's IP address.
- You know the following settings:
  - The private IP address and port number that you want to forward
  - The IP protocol to be used
  - The destination IP address and port of the web server where you want to redirect the packets

#### Procedure

- Create a firewall policy:

```
# firewall-cmd --permanent --new-policy ExamplePolicy
```

The policies, as opposed to zones, allow packet filtering for input, output, and forwarded traffic. This is important, because forwarding traffic to endpoints on locally run web servers, containers, or virtual machines requires such capability.

- Configure symbolic zones for the ingress and egress traffic to also enable the router itself to connect to its local IP address and forward this traffic:

```
# firewall-cmd --permanent --policy=ExamplePolicy --add-ingress-zone=HOST
# firewall-cmd --permanent --policy=ExamplePolicy --add-egress-zone=ANY
```

The **--add-ingress-zone=HOST** option refers to packets generated locally, which are transmitted out of the local host. The **--add-egress-zone=ANY** option refers to traffic destined to any zone.

- Add a rich rule that forwards traffic to the web server:

```
# firewall-cmd --permanent --policy=ExamplePolicy --add-rich-rule='rule family="ipv4"
destination address="192.0.2.1" forward-port port="443" protocol="tcp" to-port="443"
to-addr="192.51.100.20"
```

The rich rule forwards TCP traffic from port 443 on the router's IP address 192.0.2.1 to port 443 of the web server's IP 192.51.100.20. The rule uses the **ExamplePolicy** to ensure that the router can also connect to its local IP address.

4. Reload the firewall configuration files:

```
# firewall-cmd --reload
success
```

5. Activate routing of 127.0.0.0/8 in the kernel:

```
# echo "net.ipv4.conf.all.route_localnet=1" > /etc/sysctl.d/90-enable-route-localnet.conf
# sysctl -p /etc/sysctl.d/90-enable-route-localnet.conf
```

## Verification

1. Connect to the router's IP address and port that you have forwarded to the web server:

```
# curl https://192.0.2.1:443
```

2. Optional: Verify that **net.ipv4.conf.all.route\_localnet** is active:

```
# sysctl net.ipv4.conf.all.route_localnet
net.ipv4.conf.all.route_localnet = 1
```

3. Verify that **ExamplePolicy** is active and contains the settings you need. Especially the source IP address and port, protocol to be used, and the destination IP address and port:

```
# firewall-cmd --info-policy=ExamplePolicy
ExamplePolicy (active)
priority: -1
target: CONTINUE
ingress-zones: HOST
egress-zones: ANY
services:
ports:
protocols:
masquerade: no
forward-ports:
source-ports:
icmp-blocks:
rich rules:
rule family="ipv4" destination address="192.0.2.1" forward-port port="443" protocol="tcp" to-port="443" to-addr="192.51.100.20"
```

## Additional resources

- **firewall-cmd(1)**, **firewalld.policies(5)**, **firewalld.richlanguage(5)**, **sysctl(8)**, and **sysctl.conf(5)** man pages
- [Using configuration files in /etc/sysctl.d/ to adjust kernel parameters](#)

## 21.5.10. 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.

### 21.5.10.1. 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/icmptypes/` 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 has 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:
- Set the target of your zone to **DROP**:

```
# firewall-cmd --permanent --set-target=DROP
```

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 --permanent --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 because of the target of your zone changes to **DROP**. The requests that were blocked are not blocked. This means that if you want 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 --permanent --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
```

### 21.5.10.2. 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 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.

### 21.5.11. 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
```



#### 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.

#### 21.5.11.1. 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
```

- Create the **iplist.txt** file that contains a list of IP addresses, for example:

```
192.168.0.2
192.168.0.3
192.168.1.0/24
192.168.2.254
```

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=pass:_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.

## 21.5.12. 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.

### 21.5.12.1. 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.

### 21.5.12.2. Setting the priority of a rich rule

The following is 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

- 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.

## Verification

- 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
  }
}
```

### 21.5.13. 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 allow list 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.

#### 21.5.13.1. Configuring lockdown using CLI

You can enable or disable the lockdown feature using the command line.

##### Procedure

1. 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.

2. To enable lockdown, enter the following command as **root**:

```
# firewall-cmd --lockdown-on
```

3. To disable lockdown, use the following command as **root**:

```
# firewall-cmd --lockdown-off
```

#### 21.5.13.2. Configuring lockdown allowlist options using CLI

The lockdown allowlist can contain commands, security contexts, users and user IDs. If a command entry on the allowlist 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 in the allowlist, enter the following command as **root**:

```
# firewall-cmd --list-lockdown-whitelist-commands
```

- To add a command *command* to the allowlist, 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 allowlist, 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 in the allowlist, 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 in the allowlist, enter the following command as **root**:

```
# firewall-cmd --list-lockdown-whitelist-contexts
```

- To add a context *context* to the allowlist, enter the following command as **root**:

```
# firewall-cmd --add-lockdown-whitelist-context=context
```

- To remove a context *context* from the allowlist, enter the following command as **root**:

```
# firewall-cmd --remove-lockdown-whitelist-context=context
```

- To query whether the context *context* is in the allowlist, 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 in the allowlist, enter the following command as **root**:

```
# firewall-cmd --list-lockdown-whitelist-uids
```

- To add a user ID *uid* to the allowlist, enter the following command as **root**:

```
# firewall-cmd --add-lockdown-whitelist-uid=uid
```

- To remove a user ID *uid* from the allowlist, enter the following command as **root**:

```
# firewall-cmd --remove-lockdown-whitelist-uid=uid
```

- To query whether the user ID *uid* is in the allowlist, 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 in the allowlist, enter the following command as **root**:

```
# firewall-cmd --list-lockdown-whitelist-users
```

- To add a user name *user* to the allowlist, enter the following command as **root**:

```
# firewall-cmd --add-lockdown-whitelist-user=user
```

- To remove a user name *user* from the allowlist, enter the following command as **root**:

```
# firewall-cmd --remove-lockdown-whitelist-user=user
```

- To query whether the user name *user* is in the allowlist, 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.

### 21.5.13.3. Configuring lockdown allowlist options using configuration files

The default allowlist configuration file contains the **NetworkManager** context and the default context of **libvirt**. The user ID 0 is also on the list.

+ The allowlist configuration files are stored in the **/etc/firewalld/** directory.

```
<?xml version="1.0" encoding="utf-8"?>
<whitelist>
  <selinux context="system_u:system_r:NetworkManager_t:s0"/>
  <selinux context="system_u:system_r:virttd_t:s0-s0:c0.c1023"/>
  <user id="0"/>
</whitelist>
```

Following is an example allowlist 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 added in the allowlist 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.

## 21.5.14. Enabling traffic forwarding between different interfaces or sources within a firewalld zone

Intra-zone forwarding is a **firewalld** feature that enables traffic forwarding between interfaces or sources within a **firewalld** zone.

### 21.5.14.1. The difference between intra-zone forwarding and zones with the default target set to ACCEPT

When intra-zone forwarding is enabled, the traffic within a single **firewalld** zone can flow from one interface or source to another interface or source. The zone specifies the trust level of interfaces and sources. If the trust level is the same, communication between interfaces or sources is possible.

Note that, if you enable intra-zone forwarding in the default zone of **firewalld**, it applies only to the interfaces and sources added to the current default zone.

The **trusted** zone of **firewalld** uses a default target set to **ACCEPT**. This zone accepts all forwarded traffic, and intra-zone forwarding is not applicable for it.

As for other default target values, forwarded traffic is dropped by default, which applies to all standard zones except the trusted zone.

### 21.5.14.2. Using intra-zone forwarding to forward traffic between an Ethernet and Wi-Fi network

You can use intra-zone forwarding to forward traffic between interfaces and sources within the same **firewalld** zone. For example, use this feature to forward traffic between an Ethernet network connected to **enp1s0** and a Wi-Fi network connected to **wlp0s20**.

#### Procedure

1. Enable packet forwarding in the kernel:

```
# echo "net.ipv4.ip_forward=1" > /etc/sysctl.d/95-IPv4-forwarding.conf
# sysctl -p /etc/sysctl.d/95-IPv4-forwarding.conf
```

2. Ensure that interfaces between which you want to enable intra-zone forwarding are not assigned to a zone different than the **internal** zone:

```
# firewall-cmd --get-active-zones
```

3. If the interface is currently assigned to a zone other than **internal**, reassign it:

```
# firewall-cmd --zone=internal --change-interface=interface_name --permanent
```

4. Add the **enp1s0** and **wlp0s20** interfaces to the **internal** zone:

```
# firewall-cmd --zone=internal --add-interface=enp1s0 --add-interface=wlp0s20
```

5. Enable intra-zone forwarding:

```
# firewall-cmd --zone=internal --add-forward
```

## Verification

The following verification steps require that the **nmap-ncat** package is installed on both hosts.

1. Log in to a host that is in the same network as the **enp1s0** interface of the host you enabled zone forwarding on.
2. Start an echo service with **ncat** to test connectivity:

```
# ncat -e /usr/bin/cat -l 12345
```

3. Log in to a host that is in the same network as the **wlp0s20** interface.
4. Connect to the echo server running on the host that is in the same network as the **enp1s0**:

```
# ncat <other_host> 12345
```

5. Type something and press **Enter**, and verify the text is sent back.

## Additional resources

- **firewalld.zones(5)** man page

## 21.5.15. Configuring firewalld using System Roles

You can use the **firewall** System Role to configure settings of the **firewalld** service on multiple clients at once. This solution:

- Provides an interface with efficient input settings.
- Keeps all intended **firewalld** parameters in one place.

After you run the **firewall** role on the control node, the System Role applies the **firewalld** parameters to the managed node immediately and makes them persistent across reboots.

### 21.5.15.1. Introduction to the firewall RHEL System Role

RHEL System Roles is a set of contents for the Ansible automation utility. This content together with the Ansible automation utility provides a consistent configuration interface to remotely manage multiple systems.

The **rhel-system-roles.firewall** role from the RHEL System Roles was introduced for automated configurations of the **firewalld** service. The **rhel-system-roles** package contains this System Role, and also the reference documentation.

To apply the **firewalld** parameters on one or more systems in an automated fashion, use the **firewall** System Role variable in a playbook. A playbook is a list of one or more plays that is written in the text-based YAML format.

You can use an inventory file to define a set of systems that you want Ansible to configure.

With the **firewall** role you can configure many different **firewalld** parameters, for example:

- Zones.
- The services for which packets should be allowed.
- Granting, rejection, or dropping of traffic access to ports.
- Forwarding of ports or port ranges for a zone.

#### Additional resources

- **README.md** and **README.html** files in the `/usr/share/doc/rhel-system-roles/firewall/` directory
- [Working with playbooks](#)
- [How to build your inventory](#)

#### 21.5.15.2. Resetting the firewalld settings using the firewall RHEL System Role

With the **firewall** RHEL system role, you can reset the **firewalld** settings to their default state. If you add the **previous:replaced** parameter to the variable list, the System Role removes all existing user-defined settings and resets **firewalld** to the defaults. If you combine the **previous:replaced** parameter with other settings, the **firewall** role removes all existing settings before applying new ones.

Perform this procedure on the Ansible control node.

#### Prerequisites

- [You have prepared the control node and the managed nodes](#)
- You are logged in to the control node as a user who can run playbooks on the managed nodes.
- The account you use to connect to the managed nodes has **sudo** permissions on the them.
- The managed nodes or groups of managed nodes on which you want to run this playbook are listed in the Ansible inventory file.

#### Procedure

1. Create a playbook file, for example `~/reset-firewalld.yml`, with the following content:

```
---
- name: Reset firewalld example
  hosts: managed-node-01.example.com
```

```

tasks:
  - name: Reset firewalld
    include_role:
      name: rhel-system-roles.firewall

vars:
  firewall:
    - previous: replaced

```

2. Run the playbook:

```
# ansible-playbook ~/configuring-a-dmz.yml
```

## Verification

- Run this command as **root** on the managed node to check all the zones:

```
# firewall-cmd --list-all-zones
```

## Additional resources

- [/usr/share/ansible/roles/rhel-system-roles.firewall/README.md](#)
- [ansible-playbook\(1\)](#)
- [firewalld\(1\)](#)

### 21.5.15.3. Forwarding incoming traffic from one local port to a different local port

With the **firewall** role you can remotely configure **firewalld** parameters with persisting effect on multiple managed hosts.

Perform this procedure on the Ansible control node.

## Prerequisites

- [You have prepared the control node and the managed nodes](#)
- You are logged in to the control node as a user who can run playbooks on the managed nodes.
- The account you use to connect to the managed nodes has **sudo** permissions on the them.
- The managed nodes or groups of managed nodes on which you want to run this playbook are listed in the Ansible inventory file.

## Procedure

1. Create a playbook file, for example `~/port_forwarding.yml`, with the following content:

```

---
- name: Configure firewalld
  hosts: managed-node-01.example.com
  tasks:
    - name: Forward incoming traffic on port 8080 to 443

```

```

include_role:
  name: rhel-system-roles.firewall

vars:
  firewall:
    - { forward_port: 8080/tcp;443;, state: enabled, runtime: true, permanent: true }

```

2. Run the playbook:

```
# ansible-playbook ~/port_forwarding.yml
```

## Verification

- On the managed host, display the **firewalld** settings:

```
# firewall-cmd --list-forward-ports
```

## Additional resources

- [/usr/share/ansible/roles/rhel-system-roles.firewall/README.md](#)

### 21.5.15.4. Configuring ports using System Roles

You can use the RHEL **firewall** System Role to open or close ports in the local firewall for incoming traffic and make the new configuration persist across reboots. For example you can configure the default zone to permit incoming traffic for the HTTPS service.

Perform this procedure on the Ansible control node.

## Prerequisites

- [You have prepared the control node and the managed nodes](#)
- You are logged in to the control node as a user who can run playbooks on the managed nodes.
- The account you use to connect to the managed nodes has **sudo** permissions on the them.
- The managed nodes or groups of managed nodes on which you want to run this playbook are listed in the Ansible inventory file.

## Procedure

1. Create a playbook file, for example *~/opening-a-port.yml*, with the following content:

```

---
- name: Configure firewalld
  hosts: managed-node-01.example.com
  tasks:
    - name: Allow incoming HTTPS traffic to the local host
      include_role:
        name: rhel-system-roles.firewall

vars:
  firewall:

```

```
- port: 443/tcp
  service: http
  state: enabled
  runtime: true
  permanent: true
```

The **permanent: true** option makes the new settings persistent across reboots.

2. Run the playbook:

```
# ansible-playbook ~/opening-a-port.yml
```

## Verification

- On the managed node, verify that the **443/tcp** port associated with the **HTTPS** service is open:

```
# firewall-cmd --list-ports
443/tcp
```

## Additional resources

- [/usr/share/ansible/roles/rhel-system-roles.firewall/README.md](#)

### 21.5.15.5. Configuring a DMZ firewalld zone by using the firewalld RHEL System Role

As a system administrator, you can use the **firewall** System Role to configure a **dmz** zone on the **enp1s0** interface to permit **HTTPS** traffic to the zone. In this way, you enable external users to access your web servers.

Perform this procedure on the Ansible control node.

## Prerequisites

- [You have prepared the control node and the managed nodes](#)
- You are logged in to the control node as a user who can run playbooks on the managed nodes.
- The account you use to connect to the managed nodes has **sudo** permissions on the them.
- The managed nodes or groups of managed nodes on which you want to run this playbook are listed in the Ansible inventory file.

## Procedure

1. Create a playbook file, for example **~/configuring-a-dmz.yml**, with the following content:

```
---
- name: Configure firewalld
  hosts: managed-node-01.example.com
  tasks:
  - name: Creating a DMZ with access to HTTPS port and masquerading for hosts in DMZ
    include_role:
      name: rhel-system-roles.firewall
```



```
vars:
  firewall:
    - zone: dmz
      interface: enp1s0
      service: https
      state: enabled
      runtime: true
      permanent: true
```

2. Run the playbook:

```
# ansible-playbook ~/configuring-a-dmz.yml
```

## Verification

- On the managed node, view detailed information about the **dmz** zone:

```
# firewall-cmd --zone=dmz --list-all
dmz (active)
target: default
icmp-block-inversion: no
interfaces: enp1s0
sources:
services: https ssh
ports:
protocols:
forward: no
masquerade: no
forward-ports:
source-ports:
icmp-blocks:
```

## Additional resources

- [/usr/share/ansible/roles/rhel-system-roles.firewall/README.md](#)

### 21.5.16. Additional resources

- [firewalld\(1\)](#) man page
- [firewalld.conf\(5\)](#) man page
- [firewall-cmd\(1\)](#) man page
- [firewall-config\(1\)](#) man page
- [firewall-offline-cmd\(1\)](#) man page
- [firewalld.icmptype\(5\)](#) man page
- [firewalld.ipset\(5\)](#) man page
- [firewalld.service\(5\)](#) man page
- [firewalld.zone\(5\)](#) man page

- **firewalld.direct(5)** man page
- **firewalld.lockdown-whitelist(5)**
- **firewalld.richlanguage(5)**
- **firewalld.zones(5)** man page
- **firewalld.dbus(5)** man page

## 21.6. GETTING STARTED WITH NFTABLES

The **nftables** framework classifies packets and it is the successor to the **iptables**, **ip6tables**, **arptables**, **ebtables**, and **ipset** utilities. It offers numerous improvements in convenience, features, and performance over previous packet-filtering tools, most notably:

- Built-in lookup tables instead of linear processing
- A single framework for both the **IPv4** and **IPv6** protocols
- All rules applied atomically instead of fetching, updating, and storing a complete rule set
- Support for debugging and tracing in the rule set (**nftrace**) and monitoring trace events (in the **nft** tool)
- More consistent and compact syntax, no protocol-specific extensions
- A Netlink API for third-party applications

The **nftables** framework uses tables to store chains. The chains contain individual rules for performing actions. The **nft** utility replaces all tools from the previous packet-filtering frameworks. You can use the **libnftnl** library for low-level interaction with **nftables** Netlink API through the **libmnl** library.

To display the effect of rule set changes, use the **nft list ruleset** command. Because these utilities 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 **iptables** command.

### 21.6.1. Migrating from iptables to nftables

If your firewall configuration still uses **iptables** rules, you can migrate your **iptables** rules to **nftables**.

#### 21.6.1.1. 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 for simple firewall use cases. The utility is easy to use and covers the typical use cases for these scenarios.
- **nftables**: Use the **nftables** utility to set up complex and performance-critical firewalls, such as for a whole network.
- **iptables**: The **iptables** utility on Red Hat Enterprise Linux uses the **nf\_tables** kernel API instead of the **legacy** back end. The **nf\_tables** API provides backward compatibility so that scripts that use **iptables** commands still work on Red Hat Enterprise Linux. For new firewall scripts, Red Hat recommends to use **nftables**.



## IMPORTANT

To prevent the different firewall services from influencing each other, run only one of them on a RHEL host, and disable the other services.

### 21.6.1.2. Converting iptables and ip6tables rule sets to nftables

Use the **iptables-restore-translate** and **ip6tables-restore-translate** utilities to translate **iptables** and **ip6tables** rule sets to **nftables**.

#### Prerequisites

- The **nftables** and **iptables** packages are installed.
- The system has **iptables** and **ip6tables** rules configured.

#### Procedure

1. Write the **iptables** and **ip6tables** rules to a file:

```
# iptables-save >/root/iptables.dump
# ip6tables-save >/root/ip6tables.dump
```

2. Convert the dump files to **nftables** instructions:

```
# iptables-restore-translate -f /root/iptables.dump > /etc/nftables/ruleset-migrated-
from-iptables.nft
# ip6tables-restore-translate -f /root/ip6tables.dump > /etc/nftables/ruleset-migrated-
from-ip6tables.nft
```

3. Review and, if needed, manually update the generated **nftables** rules.
4. To enable the **nftables** service to load the generated files, add the following to the **/etc/sysconfig/nftables.conf** file:

```
include "/etc/nftables/ruleset-migrated-from-iptables.nft"
include "/etc/nftables/ruleset-migrated-from-ip6tables.nft"
```

5. Stop and disable the **iptables** service:

```
# systemctl disable --now iptables
```

If you used a custom script to load the **iptables** rules, ensure that the script no longer starts automatically and reboot to flush all tables.

6. Enable and start the **nftables** service:

```
# systemctl enable --now nftables
```

#### Verification

- Display the **nftables** rule set:

## # nft list ruleset

### Additional resources

- [Automatically loading nftables rules when the system boots](#)

### 21.6.1.3. Converting single iptables and ip6tables rules to nftables

Red Hat Enterprise Linux provides the **iptables-translate** and **ip6tables-translate** utilities to convert an **iptables** or **ip6tables** rule into the equivalent one for **nftables**.

#### Prerequisites

- The **nftables** package is installed.

#### Procedure

- Use the **iptables-translate** or **ip6tables-translate** utility instead of **iptables** or **ip6tables** to display the corresponding **nftables** rule, for example:

```
# iptables-translate -A INPUT -s 192.0.2.0/24 -j ACCEPT
nft add rule ip filter INPUT ip saddr 192.0.2.0/24 counter accept
```

Note that some extensions lack translation support. In these cases, the utility 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
```

#### Additional resources

- **iptables-translate --help**

### 21.6.1.4. Comparison of common iptables and nftables commands

The following is a comparison of common **iptables** and **nftables** commands:

- Listing all rules:

iptables	nftables
<b>iptables-save</b>	<b>nft list ruleset</b>

- Listing a certain table and chain:

iptables	nftables
<b>iptables -L</b>	<b>nft list table ip filter</b>
<b>iptables -L INPUT</b>	<b>nft list chain ip filter INPUT</b>

iptables	nftables
<b>iptables -t nat -L PREROUTING</b>	<b>nft list chain ip nat PREROUTING</b>

The **nft** command does not pre-create tables and chains. They exist only if a user created them manually.

Listing rules generated by firewalld:

```
# nft list table inet firewalld
# nft list table ip firewalld
# nft list table ip6 firewalld
```

### 21.6.1.5. Additional resources

- [iptables: The two variants and their relationship with nftables](#)

## 21.6.2. Writing and executing nftables scripts

The major benefit of using the **nftables** framework is that 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, with the **nftables** script environment, you can:

- Add comments
- Define variables
- Include other rule-set files

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.

### 21.6.2.1. Supported nftables script formats

You can write scripts in the **nftables** scripting environment in the following formats:

- 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
}
}

```

- The same syntax as for **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

```

### 21.6.2.2. Running nftables scripts

You can run an **nftables** script either by passing it to the **nft** utility or by executing the script directly.

#### Procedure

- To run an **nftables** script by passing it to the **nft** utility, enter:

```
# nft -f /etc/nftables/<example_firewall_script>.nft
```

- To run an **nftables** script directly:
  - a. For the single time that you perform this:
    - i. Ensure that the script starts with the following shebang sequence:

```
#!/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.**

- ii. Optional: Set the owner of the script to **root**:

```
# chown root /etc/nftables/<example_firewall_script>.nft
```

- iii. Make the script executable for the owner:

```
# chmod u+x /etc/nftables/<example_firewall_script>.nft
```

- b. Run the script:

```
# /etc/nftables/<example_firewall_script>.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

- **chown(1)** man page
- **chmod(1)** man page
- [Automatically loading nftables rules when the system boots](#)

### 21.6.2.3. Using comments in nftables scripts

The **nftables** scripting environment interprets everything to the right of a **#** character to the end of a line as a comment.

Comments can start at the beginning of a line, or next to a command:

```
...
# Flush the rule set
flush ruleset

add table inet example_table # Create a table
...
```

### 21.6.2.4. Using variables in 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 entering 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

Curly braces have special semantics when you use them in a rule because they indicate that the variable represents a set.

#### Additional resources

- [Using sets in nftables commands](#)
- [Using verdict maps in nftables commands](#)

#### 21.6.2.5. Including files in nftables scripts

In the **nftables** scripting environment, you can 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 21.1. Including files from the default search directory

To include a file from the default search directory:

```
include "example.nft"
```

##### Example 21.2. Including all \*.nft files from a directory

To include all files ending with **\*.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

- The **Include files** section in the **nft(8)** man page

#### 21.6.2.6. 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.

#### Prerequisites

- The **nftables** scripts are stored in the **/etc/nftables/** directory.



## Procedure

1. Edit the `/etc/sysconfig/nftables.conf` file.
  - If you modified the `*.nft` scripts that were created in `/etc/nftables/` with the installation of the `nftables` package, uncomment the `include` statement for these scripts.
  - If you wrote new scripts, 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. Optional: Start the `nftables` service to load the firewall rules without rebooting the system:

```
# systemctl start nftables
```

3. Enable the `nftables` service.

```
# systemctl enable nftables
```

## Additional resources

- [Supported nftables script formats](#)

### 21.6.3. Creating and managing nftables tables, chains, and rules

You can display `nftables` rule sets and manage them.

#### 21.6.3.1. Basics of nftables tables

A table in `nftables` is a namespace that contains a collection of chains, rules, sets, and other objects.

Each table must have an address family assigned. The address family defines the packet types that this 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 pass through a bridge device.
- **netdev**: Matches packets from ingress.

If you want to add a table, the format to use depends on your firewall script:

- In scripts in native syntax, use:

```
table <table_address_family> <table_name> {
}
```

- In shell scripts, use:

```
nft add table <table_address_family> <table_name>
```

### 21.6.3.2. Basics of nftables chains

Tables consist of chains which in turn are containers for rules. The following two rule types exist:

- **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 to better organize rules.

If you want to add a base chain to a table, the format to use depends on your firewall script:

- In scripts in native syntax, use:

```
table <table_address_family> <table_name> {
  chain <chain_name> {
    type <type> hook <hook> priority <priority>
    policy <policy> ;
  }
}
```

- In shell scripts, use:

```
nft add chain <table_address_family> <table_name> <chain_name> { type <type> hook
<hook> priority <priority> \; policy <policy> \; }
```

To avoid that the shell interprets the semicolons as the end of the command, place the `\` escape character in front of the semicolons.

Both examples create **base chains**. To create a **regular chain**, do not set any parameters in the curly brackets.

#### Chain types

The following are the chain types and an overview with which address families and hooks you can use them:

Type	Address families	Hooks	Description
<b>filter</b>	all	all	Standard chain type
<b>nat</b>	ip, ip6, inet	<b>prerouting, input, output, postrouting</b>	Chains of this type perform native address translation based on connection tracking entries. Only the first packet traverses this chain type.
<b>route</b>	ip, ip6	<b>output</b>	Accepted packets that traverse this chain type cause a new route lookup if relevant parts of the IP header have changed.

#### Chain priorities

The priority parameter specifies the order in which packets traverse chains with the same hook value. You can set this parameter to an integer value or use a standard priority name.

The following matrix is an overview of the standard priority names and their numeric values, and with which address families and hooks you can use them:

Textual value	Numeric value	Address families	Hooks
<b>raw</b>	<b>-300</b>	<b>ip, ip6, inet</b>	all
<b>mangle</b>	<b>-150</b>	<b>ip, ip6, inet</b>	all
<b>dstnat</b>	<b>-100</b>	<b>ip, ip6, inet</b>	<b>prerouting</b>
	<b>-300</b>	<b>bridge</b>	<b>prerouting</b>
<b>filter</b>	<b>0</b>	<b>ip, ip6, inet, arp, netdev</b>	all
	<b>-200</b>	<b>bridge</b>	all
<b>security</b>	<b>50</b>	<b>ip, ip6, inet</b>	all
<b>srcnat</b>	<b>100</b>	<b>ip, ip6, inet</b>	<b>postrouting</b>
	<b>300</b>	<b>bridge</b>	<b>postrouting</b>
<b>out</b>	<b>100</b>	<b>bridge</b>	<b>output</b>

### Chain policies

The chain policy defines whether **nftables** should accept or drop packets if rules in this chain do not specify any action. You can set one of the following policies in a chain:

- **accept** (default)
- **drop**

### 21.6.3.3. Basics of nftables rules

Rules define actions to perform on packets that pass a chain that contains this rule. If the rule also contains matching expressions, **nftables** performs the actions only if all previous expressions apply.

If you want to add a rule to a chain, the format to use depends on your firewall script:

- In scripts in native syntax, use:

```
table <table_address_family> <table_name> {
  chain <chain_name> {
    type <type> hook <hook> priority <priority> ; policy <policy> ;
    <rule>
  }
}
```

- In shell scripts, use:

■

```
nft add rule <table_address_family> <table_name> <chain_name> <rule>
```

This shell command appends the new rule at the end of the chain. If you prefer to add a rule at the beginning of the chain, use the **nft insert** command instead of **nft add**.

#### 21.6.3.4. Managing tables, chains, and rules using nft commands

To manage an **nftables** firewall on the command line or in shell scripts, use the **nft** utility.



#### IMPORTANT

The commands in this procedure do not represent a typical workflow and are not optimized. This procedure only demonstrates how to use **nft** commands to manage tables, chains, and rules in general.

#### Procedure

1. Create a table named **nftables\_svc** with the **inet** address family so that the table can process both IPv4 and IPv6 packets:

```
# nft add table inet nftables_svc
```

2. Add a base chain named **INPUT**, that processes incoming network traffic, to the **inet nftables\_svc** table:

```
# nft add chain inet nftables_svc INPUT { type filter hook input priority filter \; policy accept \; }
```

To avoid that the shell interprets the semicolons as the end of the command, escape the semicolons using the **\** character.

3. Add rules to the **INPUT** chain. For example, allow incoming TCP traffic on port 22 and 443, and, as the last rule of the **INPUT** chain, reject other incoming traffic with an Internet Control Message Protocol (ICMP) port unreachable message:

```
# nft add rule inet nftables_svc INPUT tcp dport 22 accept  
# nft add rule inet nftables_svc INPUT tcp dport 443 accept  
# nft add rule inet nftables_svc INPUT reject with icmpx type port-unreachable
```

If you enter the **nft add rule** commands as shown, **nft** adds the rules in the same order to the chain as you run the commands.

4. Display the current rule set including handles:

```
# nft -a list table inet nftables_svc  
table inet nftables_svc { # handle 13  
  chain INPUT { # handle 1  
    type filter hook input priority filter; policy accept;  
    tcp dport 22 accept # handle 2  
    tcp dport 443 accept # handle 3  
    reject # handle 4  
  }  
}
```

5. Insert a rule before the existing rule with handle 3. For example, to insert a rule that allows TCP traffic on port 636, enter:

```
# nft insert rule inet nftables_svc INPUT position 3 tcp dport 636 accept
```

6. Append a rule after the existing rule with handle 3. For example, to insert a rule that allows TCP traffic on port 80, enter:

```
# nft add rule inet nftables_svc INPUT position 3 tcp dport 80 accept
```

7. Display the rule set again with handles. Verify that the later added rules have been added to the specified positions:

```
# nft -a list table inet nftables_svc
table inet nftables_svc { # handle 13
  chain INPUT { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport 22 accept # handle 2
    tcp dport 636 accept # handle 5
    tcp dport 443 accept # handle 3
    tcp dport 80 accept # handle 6
    reject # handle 4
  }
}
```

8. Remove the rule with handle 6:

```
# nft delete rule inet nftables_svc INPUT handle 6
```

To remove a rule, you must specify the handle.

9. Display the rule set, and verify that the removed rule is no longer present:

```
# nft -a list table inet nftables_svc
table inet nftables_svc { # handle 13
  chain INPUT { # handle 1
    type filter hook input priority filter; policy accept;
    tcp dport 22 accept # handle 2
    tcp dport 636 accept # handle 5
    tcp dport 443 accept # handle 3
    reject # handle 4
  }
}
```

10. Remove all remaining rules from the **INPUT** chain:

```
# nft flush chain inet nftables_svc INPUT
```

11. Display the rule set, and verify that the **INPUT** chain is empty:

```
# nft list table inet nftables_svc
table inet nftables_svc {
  chain INPUT {
```

```

    type filter hook input priority filter; policy accept
  }
}

```

12. Delete the **INPUT** chain:

```
# nft delete chain inet nftables_svc INPUT
```

You can also use this command to delete chains that still contain rules.

13. Display the rule set, and verify that the **INPUT** chain has been deleted:

```
# nft list table inet nftables_svc
table inet nftables_svc {
}

```

14. Delete the **nftables\_svc** table:

```
# nft delete table inet nftables_svc
```

You can also use this command to delete tables that still contain chains.



#### NOTE

To delete the entire rule set, use the **nft flush ruleset** command instead of manually deleting all rules, chains, and tables in separate commands.

### Additional resources

**nft(8)** man page

## 21.6.4. Configuring NAT using nftables

With **nftables**, you can configure the following network address translation (NAT) types:

- Masquerading
- Source NAT (SNAT)
- Destination NAT (DNAT)
- Redirect



#### IMPORTANT

You can only use real interface names in **iifname** and **oifname** parameters, and alternative names (**altname**) are not supported.

### 21.6.4.1. NAT types

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 Service Providers do not route private IP ranges, such as **10.0.0.0/8**. If you use private 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.

Masquerading and SNAT are very similar to one another. 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 rewrite the destination address and port of incoming packets. For example, if your web server uses an IP address from a private 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.

### Redirect

This type is a special case of DNAT that redirects packets to the local machine depending on the chain hook. For example, if a service runs on a different port than its standard port, you can redirect incoming traffic from the standard port to this specific port.

#### 21.6.4.2. 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.

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 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 prevent the shell from interpreting 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
```

### 21.6.4.3. 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 router then replaces the source IP of outgoing packets.

#### Procedure

1. Create a table:

```
# nft add table nat
```

2. Add the **prerouting** and **postrouting** chains to the table:

```
# 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 prevent the shell from interpreting 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

- [Forwarding incoming packets on a specific local port to a different host](#)

### 21.6.4.4. Configuring destination NAT using nftables

Destination NAT (DNAT) enables you to redirect traffic on a router to a host that is not directly accessible from the Internet.

For example, with DNAT the router redirects incoming traffic sent to port **80** and **443** to a web server with the IP address **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 prevent the shell from interpreting the negative priority value as an option of the **nft** command.

3. Add a rule to the **prerouting** chain that redirects incoming traffic to port **80** and **443** on the **ens3** interface of the router to the web server with the IP address **192.0.2.1**:

```
# 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 for packets returning from the web server to the sender:
  - a. If the **ens3** interface uses a 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
```

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
```

### Additional resources

- [NAT types](#)

#### 21.6.4.5. Configuring a redirect using nftables

The **redirect** feature is a special case of destination network address translation (DNAT) that redirects packets to the local machine depending on the chain hook.

For example, you can redirect incoming and forwarded traffic sent to port **22** of the local host to port **2222**.

#### Procedure

1. Create a table:

```
# nft add table nat
```

2. Add the **prerouting** chain to the table:

```
# nft -- add chain nat prerouting { type nat hook prerouting priority -100 \; }
```

Note that you must pass the `--` option to the `nft` command to prevent the shell from interpreting the negative priority value as an option of the `nft` command.

3. Add a rule to the `prerouting` chain that redirects incoming traffic on port `22` to port `2222`:

```
# nft add rule nat prerouting tcp dport 22 redirect to 2222
```

### Additional resources

- [NAT types](#)

## 21.6.5. 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.

### 21.6.5.1. Using anonymous sets in nftables

An anonymous set contains comma-separated values enclosed in curly brackets, such as `{ 22, 80, 443 }`, that you use directly in a rule. You can use anonymous sets also for IP addresses and 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 [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. Optional: 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 filter; policy accept;
    tcp dport { ssh, http, https } accept
  }
}
```

### 21.6.5.2. 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::1** or **2001:db8:1::1/64**.
- **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 prevent the shell from interpreting the semicolons as the end of the command, you must escape the semicolons with a backslash.

2. Optional: 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.

### 21.6.5.3. Additional resources

- The **Sets** section in the **nft(8)** man page

### 21.6.6. 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.

#### 21.6.6.1. Using anonymous maps in nftables

An anonymous 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 an anonymous map is that if you want to change the map, you must replace the rule. For a dynamic solution, use named maps as described in [Using named maps in nftables](#).

For example, you can use an anonymous 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 a new 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 an anonymous map to **incoming\_traffic**:

■

```
# nft add rule inet example_table incoming_traffic ip protocol vmap { tcp : jump
tcp_packets, udp : jump udp_packets }
```

The anonymous 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 filter; 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.

### 21.6.6.2. Using named maps in nftables

The **nftables** framework supports named maps. You can use these maps in multiple rules within a table. Another benefit over anonymous maps is that you can update a named map without replacing the rules that use it.

When you create a named 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::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.

For example, you can allow or drop incoming packets based on their source IP address. Using a named map, you require only a single rule to configure this scenario while the IP addresses and actions are dynamically stored in the map.

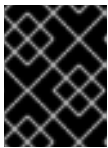
## Procedure

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 prevent the shell from interpreting 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. Optional: 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. Optional: Remove an entry from the map:

```
# nft delete element ip example_table example_map { 192.0.2.1 }
```

8. Optional: 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 filter; policy accept;
    ip saddr vmap @example_map
}
}

```

### 21.6.6.3. Additional resources

- The **Maps** section in the **nft(8)** man page

### 21.6.7. Example: Protecting a LAN and DMZ using an nftables script

Use the **nftables** framework on a RHEL router to write and install a firewall script that protects the network clients in an internal LAN and a web server in a DMZ from unauthorized access from the Internet and from other networks.



#### IMPORTANT

This example is only for demonstration purposes and describes a scenario with specific requirements.

Firewall scripts highly depend on the network infrastructure and security requirements. Use this example to learn the concepts of **nftables** firewalls when you write scripts for your own environment.

#### 21.6.7.1. Network conditions

The network in this example has the following conditions:

- The router is connected to the following networks:
  - The Internet through interface **enp1s0**
  - The internal LAN through interface **enp7s0**
  - The DMZ through **enp8s0**
- The Internet interface of the router has both a static IPv4 address (**203.0.113.1**) and IPv6 address (**2001:db8:a::1**) assigned.
- The clients in the internal LAN use only private IPv4 addresses from the range **10.0.0.0/24**. Consequently, traffic from the LAN to the Internet requires source network address translation (SNAT).
- The administrator PCs in the internal LAN use the IP addresses **10.0.0.100** and **10.0.0.200**.
- The DMZ uses public IP addresses from the ranges **198.51.100.0/24** and **2001:db8:b::/56**.
- The web server in the DMZ uses the IP addresses **198.51.100.5** and **2001:db8:b::5**.
- The router acts as a caching DNS server for hosts in the LAN and DMZ.

#### 21.6.7.2. Security requirements to the firewall script

The following are the requirements to the **nftables** firewall in the example network:

- The router must be able to:
  - Recursively resolve DNS queries.
  - Perform all connections on the loopback interface.
- Clients in the internal LAN must be able to:
  - Query the caching DNS server running on the router.
  - Access the HTTPS server in the DMZ.
  - Access any HTTPS server on the Internet.
- The PCs of the administrators must be able to access the router and every server in the DMZ using SSH.
- The web server in the DMZ must be able to:
  - Query the caching DNS server running on the router.
  - Access HTTPS servers on the Internet to download updates.
- Hosts on the Internet must be able to:
  - Access the HTTPS servers in the DMZ.
- Additionally, the following security requirements exist:
  - Connection attempts that are not explicitly allowed should be dropped.
  - Dropped packets should be logged.

### 21.6.7.3. Configuring logging of dropped packets to a file

By default, **systemd** logs kernel messages, such as for dropped packets, to the journal. Additionally, you can configure the **rsyslog** service to log such entries to a separate file. To ensure that the log file does not grow infinitely, configure a rotation policy.

#### Prerequisites

- The **rsyslog** package is installed.
- The **rsyslog** service is running.

#### Procedure

1. Create the **/etc/rsyslog.d/nftables.conf** file with the following content:

```
:msg, startswith, "nft drop" -/var/log/nftables.log
& stop
```

Using this configuration, the **rsyslog** service logs dropped packets to the **/var/log/nftables.log** file instead of **/var/log/messages**.



- Restart the **rsyslog** service:

```
# systemctl restart rsyslog
```

- Create the **/etc/logrotate.d/nftables** file with the following content to rotate **/var/log/nftables.log** if the size exceeds 10 MB:

```
/var/log/nftables.log {
    size +10M
    maxage 30
    sharedscripts
    postrotate
        /usr/bin/systemctl kill -s HUP rsyslog.service >/dev/null 2>&1 || true
    endsript
}
```

The **maxage 30** setting defines that **logrotate** removes rotated logs older than 30 days during the next rotation operation.

#### Additional resources

- **rsyslog.conf(5)** man page
- **logrotate(8)** man page

#### 21.6.7.4. Writing and activating the nftables script

This example is an **nftables** firewall script that runs on a RHEL router and protects the clients in an internal LAN and a web server in a DMZ. For details about the network and the requirements for the firewall used in the example, see [Network conditions](#) and [Security requirements to the firewall script](#).



#### WARNING

This **nftables** firewall script is only for demonstration purposes. Do not use it without adapting it to your environments and security requirements.

#### Prerequisites

- The network is configured as described in [Network conditions](#).

#### Procedure

- Create the **/etc/nftables/firewall.nft** script with the following content:

```
# Remove all rules
flush ruleset

# Table for both IPv4 and IPv6 rules
table inet nftables_svc {
```

```
# Define variables for the interface name
define INET_DEV = enp1s0
define LAN_DEV = enp7s0
define DMZ_DEV = enp8s0

# Set with the IPv4 addresses of admin PCs
set admin_pc_ipv4 {
    type ipv4_addr
    elements = { 10.0.0.100, 10.0.0.200 }
}

# Chain for incoming traffic. Default policy: drop
chain INPUT {
    type filter hook input priority filter
    policy drop

    # Accept packets in established and related state, drop invalid packets
    ct state vmap { established:accept, related:accept, invalid:drop }

    # Accept incoming traffic on loopback interface
    iifname lo accept

    # Allow request from LAN and DMZ to local DNS server
    iifname { $LAN_DEV, $DMZ_DEV } meta l4proto { tcp, udp } th dport 53 accept

    # Allow admins PCs to access the router using SSH
    iifname $LAN_DEV ip saddr @admin_pc_ipv4 tcp dport 22 accept

    # Last action: Log blocked packets
    # (packets that were not accepted in previous rules in this chain)
    log prefix "nft drop IN : "
}

# Chain for outgoing traffic. Default policy: drop
chain OUTPUT {
    type filter hook output priority filter
    policy drop

    # Accept packets in established and related state, drop invalid packets
    ct state vmap { established:accept, related:accept, invalid:drop }

    # Accept outgoing traffic on loopback interface
    oifname lo accept

    # Allow local DNS server to recursively resolve queries
    oifname $INET_DEV meta l4proto { tcp, udp } th dport 53 accept

    # Last action: Log blocked packets
    log prefix "nft drop OUT: "
}
```

```

# Chain for forwarding traffic. Default policy: drop
chain FORWARD {
    type filter hook forward priority filter
    policy drop

    # Accept packets in established and related state, drop invalid packets
    ct state vmap { established:accept, related:accept, invalid:drop }

    # IPv4 access from LAN and Internet to the HTTPS server in the DMZ
    iifname { $LAN_DEV, $INET_DEV } oifname $DMZ_DEV ip daddr 198.51.100.5 tcp dport
443 accept

    # IPv6 access from Internet to the HTTPS server in the DMZ
    iifname $INET_DEV oifname $DMZ_DEV ip6 daddr 2001:db8:b::5 tcp dport 443 accept

    # Access from LAN and DMZ to HTTPS servers on the Internet
    iifname { $LAN_DEV, $DMZ_DEV } oifname $INET_DEV tcp dport 443 accept

    # Last action: Log blocked packets
    log prefix "nft drop FWD: "
}

# Postrouting chain to handle SNAT
chain postrouting {
    type nat hook postrouting priority srcnat; policy accept;

    # SNAT for IPv4 traffic from LAN to Internet
    iifname $LAN_DEV oifname $INET_DEV snat ip to 203.0.113.1
}
}

```

2. Include the `/etc/nftables/firewall.nft` script in the `/etc/sysconfig/nftables.conf` file:

```
include "/etc/nftables/firewall.nft"
```

3. Enable IPv4 forwarding:

```
# echo "net.ipv4.ip_forward=1" > /etc/sysctl.d/95-IPv4-forwarding.conf
# sysctl -p /etc/sysctl.d/95-IPv4-forwarding.conf
```

4. Enable and start the `nftables` service:

```
# systemctl enable --now nftables
```

## Verification

1. Optional: Verify the `nftables` rule set:

```
# nft list ruleset
...
```

2. Try to perform an access that the firewall prevents. For example, try to access the router using SSH from the DMZ:

```
# ssh router.example.com
```

```
ssh: connect to host router.example.com port 22: Network is unreachable
```

3. Depending on your logging settings, search:

- The **systemd** journal for the blocked packets:

```
# journalctl -k -g "nft drop"
```

```
Oct 14 17:27:18 router kernel: nft drop IN : IN=enp8s0 OUT= MAC=...
```

```
SRC=198.51.100.5 DST=198.51.100.1 ... PROTO=TCP SPT=40464 DPT=22 ... SYN ...
```

- The **/var/log/nftables.log** file for the blocked packets:

```
Oct 14 17:27:18 router kernel: nft drop IN : IN=enp8s0 OUT= MAC=...
```

```
SRC=198.51.100.5 DST=198.51.100.1 ... PROTO=TCP SPT=40464 DPT=22 ... SYN ...
```

## 21.6.8. 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.

### 21.6.8.1. Forwarding incoming packets to a different local port

You can use **nftables** to forward packets. For example, you can forward incoming IPv4 packets on port **8022** to port **22** on the local system.

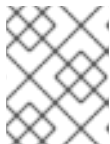
#### Procedure

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 \; }
```



#### NOTE

Pass the **--** option to the **nft** command to prevent the shell from interpreting 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 **8022** to the local port **22**:

```
# nft add rule ip nat prerouting tcp dport 8022 redirect to :22
```

### 21.6.8.2. 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.

For example, you can 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.

#### Prerequisites

- 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 prevent the shell from interpreting 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 nat postrouting 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
```

### 21.6.9. Using nftables to limit the amount of connections

You can use **nftables** to limit the number of connections or to block IP addresses that attempt to establish a given amount of connections to prevent them from using too many system resources.

#### 21.6.9.1. Limiting the number of connections using nftables

The **ct count** parameter of the **nft** utility enables administrators to limit the number of connections.

## Prerequisites

- The base **example\_chain** in **example\_table** exists.

## Procedure

1. Create a dynamic set for IPv4 addresses:

```
# nft add set inet example_table example_meter { type ipv4_addr\; flags dynamic \;}
```

2. 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
```

3. Optional: Display the set created in the previous step:

```
# nft list set inet example_table example_meter
table inet 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.

### 21.6.9.2. Blocking IP addresses that attempt more than ten new incoming TCP connections within one minute

You can temporarily block hosts that are establishing more than ten IPv4 TCP connections within one minute.

## 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 rule that drops all packets from source addresses that attempt to establish more than ten TCP connections within one minute:

```
# nft add rule ip filter input ip protocol tcp ct state new, untracked meter ratemeter { ip saddr timeout 5m limit rate over 10/minute } drop
```

The **timeout 5m** parameter defines that **nftables** automatically removes entries after five minutes to prevent that the meter fills up with stale entries.

## Verification

- To display the meter's content, enter:

```
# nft list meter ip filter ratemeter
table ip filter {
  meter ratemeter {
    type ipv4_addr
    size 65535
    flags dynamic,timeout
    elements = { 192.0.2.1 limit rate over 10/minute timeout 5m expires 4m58s224ms }
  }
}
```

## 21.6.10. Debugging nftables rules

The **nftables** framework provides different options for administrators to debug rules and if packets match them.

### 21.6.10.1. Creating a rule with a counter

To identify if a rule is matched, you can use a counter.

- For more information on a procedure that adds a counter to an existing rule, see [Adding a counter to an existing rule](#) in **Configuring and managing networking**

## 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 filter; policy accept;
    tcp dport ssh counter packets 6872 bytes 105448565 accept
  }
}
```

### 21.6.10.2. Adding a counter to an existing rule

To identify if a rule is matched, you can use a counter.

- For more information on a procedure that adds a new rule with a counter, see [Creating a rule with the counter](#) in **Configuring and managing networking**

### 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 filter; 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 filter; policy accept;
    tcp dport ssh counter packets 6872 bytes 105448565 accept
  }
}
```

### 21.6.10.3. 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. You can enable tracing for a rule and use it to monitor 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 filter; policy accept;
tcp dport ssh accept # handle 4
}
}

```

2. Add the tracing feature by replacing the rule but with the **meta nfttrace 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 nfttrace
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 nfttrace 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.

## 21.6.11. Backing up and restoring the nftables rule set

You can backup **nftables** rules to a file and later restoring them. Also, administrators can use a file with the rules to, for example, transfer the rules to a different server.

### 21.6.11.1. Backing up the nftables rule set to a file

You can use the **nft** utility to back up the **nftables** rule set to a file.

#### Procedure

- To backup **nftables** rules:
  - In a format produced by **nft list ruleset** format:

```

# nft list ruleset > file.nft

```

- In JSON format:

```

# nft -j list ruleset > file.json

```

### 21.6.11.2. Restoring the nftables rule set from a file

You can restore the **nftables** rule set from a file.

#### Procedure

- To restore **nftables** rules:
  - If the file to restore is in the format produced by **nft list ruleset** or contains **nft** commands directly:

```
# nft -f file.nft
```

- If the file to restore is in JSON format:

```
# nft -j -f file.json
```

### 21.6.12. Additional resources

- [Using nftables in Red Hat Enterprise Linux 8](#)
- [What comes after iptables? Its successor, of course: nftables](#)
- [Firewalld: The Future is nftables](#)

## PART IV. DESIGN OF HARD DISK

## CHAPTER 22. 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.

The following sections describe the file systems that Red Hat Enterprise Linux 8 includes by default, and recommendations on the most suitable file system for your application.

### 22.1. 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 22.1. 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.

## 22.2. 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

## 22.3. 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

- Reblink-based file copies
- 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.

## 22.4. 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.

## 22.5. 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 **Eoverflow** 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.

## 22.6. 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 22.2. Summary of local file system recommendations**

Scenario	Recommended file system
No special use case	XFS



Scenario	Recommended file system
Large server	XFS
Large storage devices	XFS
Large files	XFS
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

## 22.7. 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.

## 22.8. 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.

## 22.9. 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.

## 22.10. 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 23. MOUNTING NFS SHARES

As a system administrator, you can mount remote NFS shares on your system to access shared data.

### 23.1. 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.

### 23.2. 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.

## 23.3. 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, which is provided by the **quota-rpc** package, has to be started by user when the **nfs-server** is started.

### **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**

- [Configuring an NFSv4 only server without \*\*rpcbind\*\*](#).

## **23.4. 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.

### **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.

## 23.5. INSTALLING NFS

This procedure installs all packages necessary to mount or export NFS shares.

### Procedure

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

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

## 23.6. 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

- **showmount(8)** man page

## 23.7. MOUNTING AN NFS SHARE WITH MOUNT

Mount an NFS share exported from a server by using the **mount** utility.



## WARNING

You can experience conflicts in your NFSv4 **clientid** and their sudden expiration if your NFS clients have the same short hostname. To avoid any possible sudden expiration of your NFSv4 **clientid**, you must use either unique hostnames for NFS clients or configure identifier on each container, depending on what system you are using. For more information, see the [NFSv4 clientid was expired suddenly due to use same hostname on several NFS clients](#) Knowledgebase article.

## 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 want to mount.

### */remote/export*

The file system or directory being exported from the server, that is, the directory you want to mount.

### */local/directory*

The client location where */remote/export* is mounted.

## Additional resources

- [Common NFS mount options](#).
- [NFS host name formats](#).
- [Mounting a file system with mount](#).
- **mount(8)** man page
- **exports(5)** man page

## 23.8. COMMON NFS MOUNT OPTIONS

The following are the commonly used options when mounting NFS shares. You can use these options with manual **mount** commands, the **/etc/fstab** settings, and **autofs**.

### **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=flavors**

Security flavors to use for accessing files on the mounted export. The *flavors* value is a colon-separated list of one or more security flavors.

By default, the client attempts to find a security flavor that both the client and the server support. If the server does not support any of the selected flavors, the mount operation fails.

Available flavors:

- **sec=sys** uses local UNIX UIDs and GIDs. These use **AUTH\_SYS** to authenticate NFS operations.
- **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**

- **mount(8)** man page
- **nfs(5)** man page

## **23.9. ADDITIONAL RESOURCES**

- [The Linux NFS wiki](#)
- [Mounting NFS shares persistently.](#)
- [Mounting NFS shares on demand.](#)

## CHAPTER 24. EXPORTING NFS SHARES

As a system administrator, you can use the NFS server to share a directory on your system over network.

### 24.1. 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.

### 24.2. 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.

## 24.3. 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.

## 24.4. 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, which is provided by the **quota-rpc** package, has to be started by user when the **nfs-server** is started.

### **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**

- [Configuring an NFSv4 only server without \*\*rpcbind\*\*](#).

## **24.5. 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.

### **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.

## 24.6. 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

### 24.6.1. 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 24.1. 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 **nobody**. 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.

## 24.6.2. 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 **/var/lib/nfs/etab**. 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 **exports** utility, it displays a list of currently exported file systems.

#### Additional resources

- [NFS host name formats](#) .

## 24.7. NFS AND RPCBIND

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.

The Network File System Version 3 (NFSv3) requires the **rpcbind** 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

- **rpc.mountd(8)** man page
- **rpc.statd(8)** man page

## 24.8. INSTALLING NFS

This procedure installs all packages necessary to mount or export NFS shares.

#### Procedure

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

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

## 24.9. 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 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

- [Configuring an NFSv4-only server.](#)

## 24.10. 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

1. To make sure the proper NFS RPC-based services are enabled for **rpcbind**, use the following command:

```
# rpcinfo -p
```

### Example 24.2. 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

- [Configuring an NFSv4-only server.](#)

## 24.11. 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. The following sections describe how to configure NFS versions to work behind a firewall if you want to support:

- NFSv3  
This includes any servers that support NFSv3:
  - NFSv3-only servers
  - Servers that support both NFSv3 and NFSv4
- NFSv4-only

### 24.11.1. Configuring the NFSv3-enabled server to run behind a firewall

The following procedure describes how to configure servers that support NFSv3 to run behind a firewall. This includes NFSv3-only servers and servers that support both NFSv3 and NFSv4.

#### Procedure

1. 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
```

2. Specify the ports to be used by the RPC service **nlockmgr** in the **/etc/nfs.conf** file as follows:

```
[lockd]

port=tcp-port-number
udp-port=udp-port-number
```

Alternatively, you can specify **nlm\_tcpport** and **nlm\_udpport** in the **/etc/modprobe.d/lockd.conf** file.

3. Open the specified ports in the firewall by running the following commands on the NFS server:

```

firewall-cmd --permanent --add-port=<lockd-tcp-port>/tcp
firewall-cmd --permanent --add-port=<lockd-udp-port>/udp

```

4. Add static ports for **rpc.statd** by editing the **[statd]** section of the **/etc/nfs.conf** file as follows:

```

[statd]

port=port-number

```

5. Open the added ports in the firewall by running the following commands on the NFS server:

```

firewall-cmd --permanent --add-port=<statd-tcp-port>/tcp
firewall-cmd --permanent --add-port=<statd-udp-port>/udp

```

6. Reload the firewall configuration:

```

firewall-cmd --reload

```

7. Restart the **rpc-statd** service first, and then restart the **nfs-server** service:

```

# systemctl restart rpc-statd.service
# systemctl restart nfs-server.service

```

Alternatively, if you specified the **lockd** ports in the **/etc/modprobe.d/lockd.conf** file:

- a. Update the current values of **/proc/sys/fs/nfs/nlm\_tcpport** and **/proc/sys/fs/nfs/nlm\_udpport**:

```

# sysctl -w fs.nfs.nlm_tcpport=<tcp-port>
# sysctl -w fs.nfs.nlm_udpport=<udp-port>

```

- b. Restart the **rpc-statd** and **nfs-server** services:

```

# systemctl restart rpc-statd.service
# systemctl restart nfs-server.service

```

## 24.11.2. Configuring the NFSv4-only server to run behind a firewall

The following procedure describes how to configure the NFSv4-only server to run behind a firewall.

### Procedure

1. To allow clients to access NFS shares behind a firewall, configure the firewall by running the following command on the NFS server:

```

firewall-cmd --permanent --add-service nfs

```

2. Reload the firewall configuration:

```

firewall-cmd --reload

```

3. Restart the **nfs-server**:

```
# systemctl restart nfs-server
```

### 24.11.3. Configuring an NFSv3 client to run behind a firewall

The procedure to configure an NFSv3 client to run behind a firewall is similar to the procedure to configure an NFSv3 server to run behind a firewall.

If the machine you are configuring is both an NFS client and an NFS server, follow the procedure described in [Configuring the NFSv3-enabled server to run behind a firewall](#).

The following procedure describes how to configure a machine that is an NFS client only to run behind a firewall.

#### Procedure

1. To allow the NFS server to perform callbacks to the NFS client when the client is behind a firewall, add the **rpc-bind** service to the firewall by running the following command on the NFS client:

```
firewall-cmd --permanent --add-service rpc-bind
```

2. Specify the ports to be used by the RPC service **nlockmgr** in the **/etc/nfs.conf** file as follows:

```
[lockd]
port=port-number
udp-port=udp-port-number
```

Alternatively, you can specify **nlm\_tcpport** and **nlm\_udpport** in the **/etc/modprobe.d/lockd.conf** file.

3. Open the specified ports in the firewall by running the following commands on the NFS client:

```
firewall-cmd --permanent --add-port=<lockd-tcp-port>/tcp
firewall-cmd --permanent --add-port=<lockd-udp-port>/udp
```

4. Add static ports for **rpc.statd** by editing the **[statd]** section of the **/etc/nfs.conf** file as follows:

```
[statd]
port=port-number
```

5. Open the added ports in the firewall by running the following commands on the NFS client:

```
firewall-cmd --permanent --add-port=<statd-tcp-port>/tcp
firewall-cmd --permanent --add-port=<statd-udp-port>/udp
```

6. Reload the firewall configuration:

```
firewall-cmd --reload
```

7. Restart the **rpc-statd** service:

```
# systemctl restart rpc-statd.service
```

Alternatively, if you specified the **lockd** ports in the `/etc/modprobe.d/lockd.conf` file:

- a. Update the current values of `/proc/sys/fs/nfs/nlm_tcpport` and `/proc/sys/fs/nfs/nlm_udpport`:

```
# sysctl -w fs.nfs.nlm_tcpport=<tcp-port>
# sysctl -w fs.nfs.nlm_udpport=<udp-port>
```

- b. Restart the **rpc-statd** service:

```
# systemctl restart rpc-statd.service
```

#### 24.11.4. Configuring an NFSv4 client to run behind a firewall

Perform this procedure only if the client is using NFSv4.0. In that case, it is necessary to open a port for NFSv4.0 callbacks.

This procedure is not needed for NFSv4.1 or higher because in the later protocol versions the server performs callbacks on the same connection that was initiated by the client.

##### Procedure

1. 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 as follows:

```
# echo "fs.nfs.nfs_callback_tcpport = <callback-port>" >/etc/sysctl.d/90-nfs-callback-
port.conf
# sysctl -p /etc/sysctl.d/90-nfs-callback-port.conf
```

2. Open the specified port in the firewall by running the following command on the NFS client:

```
firewall-cmd --permanent --add-port=<callback-port>/tcp
```

3. Reload the firewall configuration:

```
firewall-cmd --reload
```

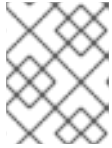
## 24.12. EXPORTING RPC QUOTA THROUGH A FIREWALL

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.

- 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.
- 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.
- Restart **rpc-rquotad** for the changes in the **/etc/sysconfig/rpc-rquotad** file to take effect:

```
# systemctl restart rpc-rquotad
```

## 24.13. ENABLING NFS OVER RDMA (NFSORDMA)

In Red Hat Enterprise Linux 8, Remote direct memory access (RDMA) service on RDMA-capable hardware provides Network File System (NFS) protocol support for high-speed file transfer over the network.

### Procedure

- Install the **rdma-core** package:

```
# yum install rdma-core
```

- Verify the lines with **xprtrdma** and **svcrdma** are commented out in the **/etc/rdma/modules/rdma.conf** file:

```
# NFS over RDMA client support
xprtrdma
# NFS over RDMA server support
svcrdma
```

- On the NFS server, create directory **/mnt/nfsordma** and export it to **/etc/exports**:

```
# mkdir /mnt/nfsordma
# echo "/mnt/nfsordma *(fsid=0,rw,async,insecure,no_root_squash)" >> /etc/exports
```

- On the NFS client, mount the nfs-share with server IP address, for example, **172.31.0.186**:

```
# mount -o rdma,port=20049 172.31.0.186:/mnt/nfs-share /mnt/nfs
```

- Restart the **nfs-server** service:

```
# systemctl restart nfs-server
```

### Additional resources

- [The RFC 5667 standard](#)

## 24.14. ADDITIONAL RESOURCES

- [The Linux NFS wiki](#)



## CHAPTER 25. 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. Therefore, 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

## 25.1. SUPPORTED SMB PROTOCOL VERSIONS

The **cifs.ko** kernel module supports the following SMB protocol versions:

- SMB 1

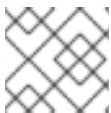


### WARNING

The SMB1 protocol is deprecated due to known security issues, and is only **safe to use on a private network**. The main reason that SMB1 is still provided as a supported option is that currently it is the only SMB protocol version that supports UNIX extensions. If you do not need to use UNIX extensions on SMB, Red Hat strongly recommends using SMB2 or later.

- 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.

## 25.2. 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.

## 25.3. 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 [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 the **OPTIONS** section in the **mount.cifs(8)** man page and [Frequently used mount options](#).

### Example 25.1. Mounting a share using an encrypted SMB 3.0 connection

To mount the `\\server\example\` share as the **DOMAINAdministrator** user over an encrypted SMB 3.0 connection into the `/mnt/` directory:

```
# mount -t cifs -o username=DOMAINAdministrator,seal,vers=3.0 //server/example /mnt/
Password for DOMAINAdministrator@//server_name/share_name: password
```

## 25.4. 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 [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 the **OPTIONS** section in the **mount.cifs(8)** man page and [Frequently used mount options](#).

To verify that the share mounts successfully, enter:

```
# mount /mnt/
```

## 25.5. 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.

## 25.6. 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 [Mounting a share with the multiuser option](#).

The following list gives frequently used mount options:

Option	Description
<code>credentials=<i>file_name</i></code>	Sets the path to the credentials file. See <a href="#">Authenticating to an SMB share using a credentials file</a> .
<code>dir_mode=<i>mode</i></code>	Sets the directory mode if the server does not support CIFS UNIX extensions.
<code>file_mode=<i>mode</i></code>	Sets the file mode if the server does not support CIFS UNIX extensions.
<code>password=<i>password</i></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 the example in <a href="#">Manually mounting an SMB share</a> .

Option	Description
<code>sec=security_mode</code>	<p>Sets the security mode, such as <b>ntlmsspi</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>	<p>Sets the user name used to authenticate to the SMB server. Alternatively, specify a credentials file using the <b>credentials</b> option.</p>
<code>vers=SMB_protocol_version</code>	<p>Sets the SMB protocol version used for the communication with the server.</p>

For a complete list, see the **OPTIONS** section in the **mount.cifs(8)** man page.

## CHAPTER 26. 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.

### 26.1. 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 take 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.

### 26.2. FILE SYSTEM AND DEVICE IDENTIFIERS

This sections 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.

## 26.3. DEVICE NAMES MANAGED BY THE UDEV MECHANISM IN /DEV/DISK/

The **udev** mechanism is used for all types of devices in Linux, and is not limited only for storage devices. It provides different kinds of persistent naming attributes in the **/dev/disk/** directory. 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.

### 26.3.1. 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.

## 26.3.2. 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 26.1. 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
<code>/dev/disk/by-id/ata-SAMSUNG_MZNLN256MHQ-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 26.2. 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.

The Path attribute fails if any part of the hardware path (for example, the PCI ID, target port, or LUN number) changes. The Path attribute is therefore unreliable. However, the Path attribute may be useful in one of the following scenarios:

- You need to identify a disk that you are planning to replace later.
- You plan to install a storage service on a disk in a specific location.

## 26.4. THE WORLD WIDE IDENTIFIER WITH DM MULTIPATH

You can configure Device Mapper (DM) Multipath to map between the World Wide Identifier (WWID) and non-persistent device names.

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 26.3. WWID mappings in a multipath configuration

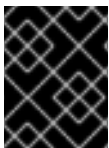
An example output of the **multipath -l** command:

```
3600508b400105df70000e00000ac0000 dm-2 vendor,product
```

```
[size=20G][features=1 queue_if_no_path][hwhandler=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.

## 26.5. 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 **udev** 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.
- The device names managed by the **udev** mechanism in **/dev/disk/** may change between major releases, requiring you to update the links.

## 26.6. 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 26.4. 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 26.5. 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 26.6. 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-
QIWtEHtXGobe5bewlIUdivKOz5ofkgFhP0RMFsNyySVihqEI2cWWbR7MjXJoID6g
symbolic link to ../../dm-1
/dev/disk/by-id/dm-uuid-LVM-
QIWtEHtXGobe5bewlIUdivKOz5ofkgFhXqH2M45hD2H9nAf2qfWSrIRLhzfMyOKd
symbolic link to ../../dm-0
/dev/disk/by-id/lvm-pv-uuid-atlr2Y-vuMo-ueoH-CpMG-4JuH-AhEF-wu4QQm
symbolic link to ../../sda2
```

## 26.7. 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 **swlabel** utility:

```
# swlabel --uuid new-uuid --label new-label swap-device  
# udevadm settle
```

## CHAPTER 27. GETTING STARTED WITH PARTITIONS

Use disk partitioning to divide a disk into one or more logical areas which enables work on each partition separately. The hard disk stores information about the location and size of each disk partition in the partition table. Using the table, each partition then appears as a logical disk to the operating system. You can then read and write on those individual disks.

For an overview of the advantages and disadvantages to using partitions on block devices, see [What are the advantages and disadvantages to using partitioning on LUNs, either directly or with LVM in between?](#).

### 27.1. CREATING A PARTITION TABLE ON A DISK WITH PARTED

Use the **parted** utility to format a block device with a partition table more easily.



#### WARNING

Formatting a block device with a partition table deletes all data stored on the device.

#### Procedure

1. Start the interactive **parted** shell:

```
# parted block-device
```

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 following steps.

3. Create the new partition table:

```
# (parted) mklabel table-type
```

- Replace *table-type* with with the intended partition table type:
  - **msdos** for MBR
  - **gpt** for GPT

#### Example 27.1. Creating a GUID Partition Table (GPT) table

To create a GPT table on the disk, use:

```
# (parted) mklabel gpt
```

The changes start applying after you enter this command.

4. View the partition table to confirm that it is created:

```
# (parted) print
```

5. Exit the **parted** shell:

```
# (parted) quit
```

### Additional resources

- **parted(8)** man page.

## 27.2. VIEWING THE PARTITION TABLE WITH PARTED

Display the partition table of a block device to see the partition layout and details about individual partitions. You can view the partition table on a block device using the **parted** utility.

### Procedure

1. Start the **parted** utility. For example, the following output lists the device **/dev/sda**:

```
# parted /dev/sda
```

2. View the partition table:

```
# (parted) print

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
```

3. Optional: Switch to the device you want to examine next:

```
# (parted) select block-device
```

For a detailed description of the print command output, see the following:

#### **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**.

**Additional resources**

- **parted(8)** man page.

## 27.3. CREATING A PARTITION WITH PARTED

As a system administrator, you can create new partitions on a disk by using the **parted** utility.

**NOTE**

The required partitions are **swap**, **/boot/**, and **/ (root)**.

**Prerequisites**

- A partition table on the disk.
- If the partition you want to create is larger than 2TiB, format the disk with the **GUID Partition Table (GPT)**.

**Procedure**

1. Start the **parted** utility:

```
# parted block-device
```

2. View the current partition table to determine if there is enough free space:

```
# (parted) print
```

- Resize the partition in case there is not enough free space.
- 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**. 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 **xfs**, **ext2**, **ext3**, **ext4**, **fat16**, **fat32**, **hfs**, **hfs+**, **linux-swap**, **ntfs**, or **reiserfs**. The *fs-type* parameter is optional. Note that the **parted** utility 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 is in megabytes.

#### Example 27.2. Creating a small primary partition

To create a primary partition from 1024MiB until 2048MiB on an MBR table, use:

```
# (parted) mkpart primary 1024MiB 2048MiB
```

The changes start applying after you enter the command.

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. Register the new device node:

```
# udevadm settle
```

7. Verify that the kernel recognizes the new partition:

```
# cat /proc/partitions
```

#### Additional resources

- **parted(8)** man page.
- [Creating a partition table on a disk with parted](#) .
- [Resizing a partition with parted](#)

## 27.4. SETTING A PARTITION TYPE WITH FDISK



You can set a partition type or flag, using the **fdisk** utility.

### Prerequisites

- A partition on the disk.

### Procedure

1. Start the interactive **fdisk** shell:

```
# fdisk block-device
```

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. Optional: View the list in 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
```

## 27.5. RESIZING A PARTITION WITH PARTED

Using the **parted** utility, extend a partition to utilize unused disk space, or shrink a partition to use its capacity for different purposes.

### Prerequisites

- Back up the data before shrinking a partition.

- If the partition you want to create is larger than 2TiB, format the disk with the **GUID Partition Table (GPT)**.
- If you want to shrink the partition, first shrink the file system so that it is not larger than the resized partition.

**NOTE**

XFS does not support shrinking.

**Procedure**

1. Start the **parted** utility:

```
# parted block-device
```

2. 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.

3. Resize the partition:

```
# (parted) resizepart 1 2GiB
```

- Replace *1* with the minor number of the partition that you are resizing.
- Replace *2* 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 is in megabytes.

4. View the partition table to confirm that the resized partition is in the partition table with the correct size:

```
# (parted) print
```

5. Exit the **parted** shell:

```
# (parted) quit
```

6. Verify that the kernel registers the new partition:

```
# cat /proc/partitions
```

7. Optional: If you extended the partition, extend the file system on it as well.

**Additional resources**

- **parted(8)** man page.

- [Creating a partition table on a disk with parted](#)
- [Resizing an ext3 file system](#)
- [Increasing the size of an XFS file system](#)

## 27.6. REMOVING A PARTITION WITH PARTED

Using the **parted** utility, you can remove a disk partition to free up disk space.



### WARNING

Removing a partition deletes all data stored on the partition.

### 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.

The changes start applying as soon as you enter this command.

4. Verify that you have removed the partition from the partition table:

```
(parted) print
```

5. Exit the **parted** shell:

```
(parted) quit
```

6. Verify that the kernel registers that 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:

- a. List active kernel options and see if any option references the removed partition:

```
# grubby --info=ALL
```

- b. Remove the kernel options that reference the removed partition:

```
# grubby --update-kernel=ALL --remove-args="option"
```

10. To register the changes in the early boot system, rebuild the **initramfs** file system:

```
# dracut --force --verbose
```

### Additional resources

- **parted(8)** man page

## CHAPTER 28. GETTING STARTED WITH XFS

This is an overview of how to create and maintain XFS file systems.

### 28.1. 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

- Reblink-based file copies
- 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.

## 28.2. 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 29. MOUNTING FILE SYSTEMS

As a system administrator, you can mount file systems on your system to access data on them.

### 29.1. 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

- **mount(8)** man page
- [How to list persistent naming attributes such as the UUID](#) .

### 29.2. 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 29.1. 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

- findmnt(8)** man page

## 29.3. 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

- To attach a certain file system, use the **mount** utility:

```
# mount device mount-point
```

#### Example 29.2. 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
```

- If **mount** cannot recognize the file system type automatically, specify it using the **--types** option:

```
# mount --types type device mount-point
```

#### Example 29.3. Mounting an NFS file system



For example, to mount a remote NFS file system:

```
# mount --types nfs4 host:/remote-export /mnt/nfs
```

### Additional resources

- **mount(8)** man page

## 29.4. 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 29.4. 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

- **mount(8)** man page

## 29.5. UNMOUNTING A FILE SYSTEM WITH Umount

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.
```

- 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.

## 29.6. COMMON MOUNT OPTIONS

The following table lists the most common options of the **mount** utility. You can apply these mount options using the following syntax:

```
# mount --options option1,option2,option3 device mount-point
```

Table 29.1. 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.
<b>ro</b>	Mounts the file system for reading only.

Option	Description
<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.

## CHAPTER 30. 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.

### 30.1. 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 following mount types are available:

#### **private**

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**

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**

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**

This type prevents the given mount point from being duplicated whatsoever.

#### **Additional resources**

- [The \*Shared subtrees\* article on Linux Weekly News.](#)

### 30.2. 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 30.1. 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

- **mount(8)** man page

## 30.3. 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 30.2. 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

- **mount(8)** man page

## 30.4. CREATING A SLAVE MOUNT POINT DUPLICATE

This procedure duplicates a mount point as a **slave** mount type. 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 the **slave** type:

```
# mount --bind original-dir duplicate-dir
# mount --make-slave duplicate-dir
```

### Example 30.3. 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

- **mount(8)** man page

## 30.5. 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-runbindable** 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 30.4. Preventing /media from being duplicated

- To prevent the **/media** directory from being shared, use:

```
# mount --bind /media /media
# mount --make-unbindable /media
```

#### Additional resources

- **mount(8)** man page



## CHAPTER 31. PERSISTENTLY MOUNTING FILE SYSTEMS

As a system administrator, you can persistently mount file systems to configure non-removable storage.

### 31.1. THE /ETC/FSTAB FILE

Use the `/etc/fstab` configuration file to control persistent mount points of 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, which includes the `defaults` option to mount the partition at boot time with default options. The mount option field also recognizes the `systemd` mount unit options in the `x-systemd.option` format.
5. Backup option for the `dump` utility.
6. Check order for the `fsck` utility.



#### NOTE

The `systemd-fstab-generator` dynamically converts the entries from the `/etc/fstab` file to the `systemd-mount` units. The `systemd` auto mounts LVM volumes from `/etc/fstab` during manual activation unless the `systemd-mount` unit is masked.

#### Example 31.1. 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

- `fstab(5)` and `systemd.mount(5)` man pages

### 31.2. 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

1. Find out the UUID attribute of the file system:

```
$ lsblk --fs storage-device
```

For example:

### Example 31.2. Viewing the UUID of a partition

```
$ lsblk --fs /dev/sda1
```

NAME	FSTYPE	LABEL	UUID	MOUNTPOINT
sda1	xf	Boot	ea74bbec-536d-490c-b8d9-5b40bbd7545b	/boot

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.  
For example:

### Example 31.3. The /boot mount point in /etc/fstab

```
UUID=ea74bbec-536d-490c-b8d9-5b40bbd7545b /boot xfs defaults 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

- [Overview of persistent naming attributes.](#)

## CHAPTER 32. PERSISTENTLY MOUNTING A FILE SYSTEM USING RHEL SYSTEM ROLES

Use the **storage** role to persistently mount a file system.

### Prerequisites

- An Ansible playbook that uses the **storage** role exists.

### 32.1. 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.

#### Example 32.1. A playbook that mounts a file system on `/dev/sdb` to `/mnt/data`

```
---
- hosts: all
  vars:
    storage_volumes:
      - name: barefs
        type: disk
        disks:
          - sdb
        fs_type: xfs
        mount_point: /mnt/data
  roles:
    - rhel-system-roles.storage
```

- This playbook adds the file system to the `/etc/fstab` file, and mounts the file system immediately.
- If the file system on the `/dev/sdb` device or the mount point directory do not exist, the playbook creates them.

### Additional resources

- The `/usr/share/ansible/roles/rhel-system-roles.storage/README.md` file.

## CHAPTER 33. MOUNTING FILE SYSTEMS ON DEMAND

As a system administrator, you can configure file systems, such as NFS, to mount automatically on demand.

### 33.1. 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.

### 33.2. 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 33.1. 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 options are appended to the master map entry options, if any, or used instead of the master map options if the configuration entry **append\_options** is set to **no**.

#### 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 33.2. A map file

The following is a sample from a map file; for example, `/etc/auto.misc`:

```
payroll -fstype=nfs4 personnel:/exports/payroll
sales -fstype=ufs :/dev/hda4
```

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 not needed if the file system is NFS, including mounts for NFSv4 if the system default is NFSv4 for NFS mounts.

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

- **autofs(5)** man page
- **autofs.conf(5)** man page
- **auto.master(5)** man page
- **/usr/share/doc/autofs/README.amd-maps** file

## 33.3. 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

1. 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.
2. In the map file, fill in the mount point, options, and location fields as described in [The autofs configuration files](#) section.
3. Register the map file in the master map file, as described in [The autofs configuration files](#) section.
4. Allow the service to re-read the configuration, so it can manage the newly configured **autofs** mount:

```
# systemctl reload autofs.service
```

- 5. Try accessing content in the on-demand directory:

```
# ls automounted-directory
```

## 33.4. 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

1. 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
```

2. 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.

3. Reload the **autofs** service:

```
# systemctl reload autofs
```

## 33.5. 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 33.3. 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 **autofs** configuration option **BROWSE\_MODE** is set to **yes**:

```
BROWSE_MODE="yes"
```

- The file map **/etc/auto.home** does not exist.

## Procedure

This section describes the examples of mounting home directories from a different server and augmenting **auto.home** with only selected entries.

### Example 33.4. 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 33.5. Augmenting auto.home with only selected entries

Alternatively, to augment the site-wide **auto.home** map with just a few entries:

1. 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
```



2. 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.

## 33.6. 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

1. 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.
2. 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 33.6. 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"
```

3. Ensure that all other schema entries are commented in the configuration. The **automountKey** attribute of the **rfc2307bis** schema replaces the **cn** attribute of the **rfc2307** schema. Following is an example of an LDAP Data Interchange Format (LDIF) configuration:

#### Example 33.7. LDIF Configuration

```
# auto.master, example.com
dn: automountMapName=auto.master,dc=example,dc=com
objectClass: top
objectClass: automountMap
automountMapName: auto.master

# /home, auto.master, example.com
dn: automountMapName=auto.master,dc=example,dc=com
```

```

objectClass: automount
automountKey: /home
automountInformation: auto.home

# auto.home, example.com
dn: automountMapName=auto.home,dc=example,dc=com
objectClass: automountMap
automountMapName: auto.home

# 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/&

```

#### Additional resources

- [The rfc2307bis draft](#)

## 33.7. USING SYSTEMD.AUTOMOUNT TO MOUNT A FILE SYSTEM ON DEMAND WITH /ETC/FSTAB

This procedure shows how to mount a file system on demand using the automount systemd units when mount point is defined in **/etc/fstab**. You have to add an automount unit for each mount and enable it.

#### Procedure

1. Add desired fstab entry as documented in [Chapter 30. Persistently mounting file systems](#). For example:

```
/dev/disk/by-id/da875760-edb9-4b82-99dc-5f4b1ff2e5f4 /mount/point xfs defaults 0 0
```

2. Add **x-systemd.automount** to the options field of entry created in the previous step.
3. Load newly created units so that your system registers the new configuration:

```
# systemctl daemon-reload
```

4. Start the automount unit:

```
# systemctl start mount-point.automount
```

#### Verification

1. Check that **mount-point.automount** is running:

```
# systemctl status mount-point.automount
```

2. Check that automounted directory has desired content:

```
# ls /mount/point
```

#### Additional resources

- **systemd.automount(5)** man page.
- **systemd.mount(5)** man page.
- [Introduction to systemd](#).

## 33.8. USING SYSTEMD.AUTOMOUNT TO MOUNT A FILE SYSTEM ON DEMAND WITH A MOUNT UNIT

This procedure shows how to mount a file system on demand using the automount systemd units when mount point is defined by a mount unit. You have to add an automount unit for each mount and enable it.

#### Procedure

1. Create a mount unit. For example:

```
mount-point.mount
[Mount]
What=/dev/disk/by-uuid/f5755511-a714-44c1-a123-cfde0e4ac688
Where=/mount/point
Type=xfs
```

2. Create a unit file with the same name as the mount unit, but with extension **.automount**.
3. Open the file and create an **[Automount]** section. Set the **Where=** option to the mount path:

```
[Automount]
Where=/mount/point
[Install]
WantedBy=multi-user.target
```

4. Load newly created units so that your system registers the new configuration:

```
# systemctl daemon-reload
```

5. Enable and start the automount unit instead:

```
# systemctl enable --now mount-point.automount
```

#### Verification

1. Check that ***mount-point.automount*** is running:

```
# systemctl status mount-point.automount
```

2. Check that automounted directory has desired content:

```
# ls /mount/point
```

#### Additional resources

- **systemd.automount(5)** man page.
- **systemd.mount(5)** man page.
- [Introduction to systemd](#).

## CHAPTER 34. USING SSSD COMPONENT FROM IDM TO CACHE THE AUTOFS MAPS

The System Security Services Daemon (SSSD) is a system service to access remote service directories and authentication mechanisms. The data caching is useful in case of the slow network connection. To configure the SSSD service to cache the autofs map, follow the procedures below in this section.

### 34.1. CONFIGURING AUTOFS MANUALLY TO USE IDM SERVER AS AN LDAP SERVER

This procedure shows how to configure **autofs** to use IdM server as an LDAP server.

#### Procedure

1. Edit the `/etc/autofs.conf` file to specify the schema attributes that **autofs** searches for:

```
#
# Other common LDAP naming
#
map_object_class = "automountMap"
entry_object_class = "automount"
map_attribute = "automountMapName"
entry_attribute = "automountKey"
value_attribute = "automountInformation"
```



#### NOTE

User can write the attributes in both lower and upper cases in the `/etc/autofs.conf` file.

2. Optionally, specify the LDAP configuration. There are two ways to do this. The simplest is to let the automount service discover the LDAP server and locations on its own:

```
ldap_uri = "ldap:///dc=example,dc=com"
```

*This option requires DNS to contain SRV records for the discoverable servers.*

Alternatively, explicitly set which LDAP server to use and the base DN for LDAP searches:

```
ldap_uri = "ldap://ipa.example.com"
search_base = "cn=location,cn=automount,dc=example,dc=com"
```

3. Edit the `/etc/autofs_ldap_auth.conf` file so that autofs allows client authentication with the IdM LDAP server.

- Change **authrequired** to yes.
- Set the principal to the Kerberos host principal for the IdM LDAP server, `host/fqdn@REALM`. The principal name is used to connect to the IdM directory as part of GSS client authentication.

```
<autofs_ldap_sasl_conf
```

```

usetls="no"
tlsrequired="no"
authrequired="yes"
authtype="GSSAPI"
clientprinc="host/server.example.com@EXAMPLE.COM"
/>

```

For more information about host principal, see [Using canonicalized DNS host names in IdM](#) .

If necessary, run **klist -k** to get the exact host principal information.

## 34.2. CONFIGURING SSSD TO CACHE AUTOFS MAPS

The SSSD service can be used to cache **autofs** maps stored on an IdM server without having to configure **autofs** to use the IdM server at all.

### Prerequisites

- The **sssd** package is installed.

### Procedure

1. Open the SSSD configuration file:

```
# vim /etc/sss/sss.conf
```

2. Add the **autofs** service to the list of services handled by SSSD.

```

[sss]
domains = ldap
services = nss,pam,autofs

```

3. Create a new **[autofs]** section. You can leave this blank, because the default settings for an **autofs** service work with most infrastructures.

```

[nss]

[pam]

[sudo]

[autofs]

[ssh]

[pac]

```

For more information, see the **sss.conf** man page.

4. Optionally, set a search base for the **autofs** entries. By default, this is the LDAP search base, but a subtree can be specified in the **ldap\_autofs\_search\_base** parameter.

```
[domain/EXAMPLE]
```

```
ldap_search_base = "dc=example,dc=com"
ldap_autofs_search_base = "ou=automount,dc=example,dc=com"
```

- Restart SSSD service:

```
# systemctl restart sssd.service
```

- Check the **/etc/nsswitch.conf** file, so that SSSD is listed as a source for automount configuration:

```
automount: sss files
```

- Restart **autofs** service:

```
# systemctl restart autofs.service
```

- Test the configuration by listing a user's **/home** directory, assuming there is a master map entry for **/home**:

```
# ls /home/userName
```

If this does not mount the remote file system, check the **/var/log/messages** file for errors. If necessary, increase the debug level in the **/etc/sysconfig/autofs** file by setting the **logging** parameter to **debug**.

## CHAPTER 35. 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.

### 35.1. 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. Note that the **readonly-root** package is required to have this file present in your system.

```
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/**.

## 35.2. 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. Enable the **ro** kernel option:

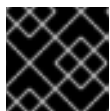
```
# grubby --update-kernel=ALL --args="ro"
```

4. Ensure that the **rw** kernel option is disabled:

```
# grubby --update-kernel=ALL --remove-args="rw"
```

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 36. MANAGING STORAGE DEVICES

### 36.1. SETTING UP STRATIS FILE SYSTEMS

Stratis runs as a service to manage pools of physical storage devices, simplifying local storage management with ease of use while helping you set up and manage complex storage configurations.



#### IMPORTANT

Stratis is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process. For more information about the support scope of Red Hat Technology Preview features, see <https://access.redhat.com/support/offerings/techpreview>.

#### 36.1.1. What is 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

#### Additional resources

- [Stratis website](#)

#### 36.1.2. Components of a Stratis volume

Learn about the components that comprise 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 **/dev/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.

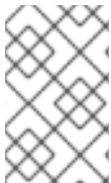
The file systems are formatted with XFS.



#### 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 **/dev/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.

### 36.1.3. Block devices usable with Stratis

Storage devices that can be used with 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

### 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.

## 36.1.4. Installing Stratis

Install the required packages for Stratis.

### Procedure

1. Install packages that provide the Stratis service and command-line utilities:

```
# yum install stratisd stratis-cli
```

2. Verify that the **stratisd** service is enabled:

```
# systemctl enable --now stratisd
```

## 36.1.5. Creating an unencrypted Stratis pool

You can create an unencrypted Stratis pool from one or more block devices.

### Prerequisites

- Stratis is installed. For more information, see [Installing Stratis](#).
- The **stratisd** service is running.
- The block devices on which you are creating a Stratis pool are not in use and are not mounted.
- Each block device on which you are creating a Stratis pool is at least 1 GB.
- 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](#).



### NOTE

You cannot encrypt an unencrypted Stratis pool.

### Procedure

1. Erase any file system, partition table, or RAID signatures that exist on each block device that you want to use in the Stratis pool:

```
# wipefs --all block-device
```

where ***block-device*** is the path to the block device; for example, **/dev/sdb**.

2. Create the new unencrypted Stratis pool on the selected block device:

```
# stratis pool create my-pool block-device
```

where **block-device** is the path to an empty or wiped block device.



#### NOTE

Specify multiple block devices on a single line:

```
# stratis pool create my-pool block-device-1 block-device-2
```

3. Verify that the new Stratis pool was created:

```
# stratis pool list
```

### 36.1.6. Creating an encrypted Stratis pool

To secure your data, you can create an encrypted Stratis pool from one or more block devices.

When you create an encrypted Stratis pool, the kernel keyring is used as the primary encryption mechanism. After subsequent system reboots this kernel keyring is used to unlock the encrypted Stratis pool.

When creating an encrypted Stratis pool from one or more block devices, note the following:

- Each block device is encrypted using the **cryptsetup** library and implements the **LUKS2** format.
- Each Stratis pool can either have a unique key or share the same key with other pools. These keys are stored in the kernel keyring.
- The block devices that comprise a Stratis pool must be either all encrypted or all unencrypted. It is not possible to have both encrypted and unencrypted block devices in the same Stratis pool.
- Block devices added to the data tier of an encrypted Stratis pool are automatically encrypted.

#### Prerequisites

- Stratis v2.1.0 or later is installed. For more information, see [Installing Stratis](#).
- The **stratisd** service is running.
- The block devices on which you are creating a Stratis pool are not in use and are not mounted.
- The block devices on which you are creating a Stratis pool are at least 1GB 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. Erase any file system, partition table, or RAID signatures that exist on each block device that you want to use in the Stratis pool:

```
# wipefs --all block-device
```

where ***block-device*** is the path to the block device; for example, `/dev/sdb`.

- If you have not created a key set already, run the following command and follow the prompts to create a key set to use for the encryption.

```
# stratis key set --capture-key key-description
```

where ***key-description*** is a reference to the key that gets created in the kernel keyring.

- Create the encrypted Stratis pool and specify the key description to use for the encryption. You can also specify the key path using the **`--keyfile-path`** option instead of using the ***key-description*** option.

```
# stratis pool create --key-desc key-description my-pool block-device
```

where

#### ***key-description***

References the key that exists in the kernel keyring, which you created in the previous step.

#### ***my-pool***

Specifies the name of the new Stratis pool.

#### ***block-device***

Specifies the path to an empty or wiped block device.



#### **NOTE**

Specify multiple block devices on a single line:

```
# stratis pool create --key-desc key-description my-pool block-device-1  
block-device-2
```

- Verify that the new Stratis pool was created:

```
# stratis pool list
```

### 36.1.7. Setting up a thin provisioning layer in Stratis filesystem

A storage stack can reach a state of overprovision. If the file system size becomes bigger than the pool backing it, the pool becomes full. To prevent this, disable overprovisioning, which ensures that the size of all filesystems on the pool does not exceed the available physical storage provided by the pool. If you use Stratis for critical applications or the root filesystem, this mode prevents certain failure cases.

If you enable overprovisioning, an API signal notifies you when your storage has been fully allocated. The notification serves as a warning to the user to inform them that when all the remaining pool space fills up, Stratis has no space left to extend to.

#### Prerequisites

- Stratis is installed. For more information, see [Installing Stratis](#).

## Procedure

To set up the pool correctly, you have two possibilities:

1. Create a pool from one or more block devices:

```
# stratis pool create --no-overprovision pool-name /dev/sdb
```

- By using the **--no-overprovision** option, the pool cannot allocate more logical space than actual available physical space.

2. Set overprovisioning mode in the existing pool:

```
# stratis pool overprovision pool-name <yes|no>
```

- If set to "yes", you enable overprovisioning to the pool. This means that the sum of the logical sizes of the Stratis filesystems, supported by the pool, can exceed the amount of available data space.

## Verification

1. Run the following to view the full list of Stratis pools:

```
# stratis pool list
```

Name	Total Physical	Properties	UUID	Alerts
<i>pool-name</i>	1.42 TiB / 23.96 MiB / 1.42 TiB	~Ca,~Cr,~Op	cb7cb4d8-9322-4ac4-a6fd-eb7ae9e1e540	

2. Check if there is an indication of the pool overprovisioning mode flag in the **stratis pool list** output. The " ~ " is a math symbol for "NOT", so **~Op** means no-overprovisioning.
3. Optional: Run the following to check overprovisioning on a specific pool:

```
# stratis pool overprovision pool-name yes
```

```
# stratis pool list
```

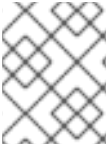
Name	Total Physical	Properties	UUID	Alerts
<i>pool-name</i>	1.42 TiB / 23.96 MiB / 1.42 TiB	~Ca,~Cr,~Op	cb7cb4d8-9322-4ac4-a6fd-eb7ae9e1e540	

## Additional resources

- [The Stratis Storage webpage](#).

### 36.1.8. Binding a Stratis pool to NBDE

Binding an encrypted Stratis pool to Network Bound Disk Encryption (NBDE) requires a Tang server. When a system containing the Stratis pool reboots, it connects with the Tang server to automatically unlock the encrypted pool without you having to provide the kernel keyring description.

**NOTE**

Binding a Stratis pool to a supplementary Clevis encryption mechanism does not remove the primary kernel keyring encryption.

**Prerequisites**

- Stratis v2.3.0 or later is installed. For more information, see [Installing Stratis](#).
- The **stratisd** service is running.
- You have created an encrypted Stratis pool, and you have the key description of the key that was used for the encryption. For more information, see [Creating an encrypted Stratis pool](#).
- You can connect to the Tang server. For more information, see [Deploying a Tang server with SELinux in enforcing mode](#)

**Procedure**

- Bind an encrypted Stratis pool to NBDE:

```
# stratis pool bind nbde --trust-url my-pool tang-server
```

where

***my-pool***

Specifies the name of the encrypted Stratis pool.

***tang-server***

Specifies the IP address or URL of the Tang server.

**Additional resources**

- [Configuring automated unlocking of encrypted volumes using policy-based decryption](#)

### 36.1.9. Binding a Stratis pool to TPM

When you bind an encrypted Stratis pool to the Trusted Platform Module (TPM) 2.0, when the system containing the pool reboots, the pool is automatically unlocked without you having to provide the kernel keyring description.

**Prerequisites**

- Stratis v2.3.0 or later is installed. For more information, see [Installing Stratis](#).
- The **stratisd** service is running.
- You have created an encrypted Stratis pool. For more information, see [Creating an encrypted Stratis pool](#).

**Procedure**

- Bind an encrypted Stratis pool to TPM:

```
# stratis pool bind tpm my-pool key-description
```



-  
where

***my-pool***

Specifies the name of the encrypted Stratis pool.

***key-description***

References the key that exists in the kernel keyring, which was generated when you created the encrypted Stratis pool.

### 36.1.10. Unlocking an encrypted Stratis pool with kernel keyring

After a system reboot, your encrypted Stratis pool or the block devices that comprise it might not be visible. You can unlock the pool using the kernel keyring that was used to encrypt the pool.

#### Prerequisites

- Stratis v2.1.0 is installed. For more information, see [Installing Stratis](#).
- The **stratisd** service is running.
- You have created an encrypted Stratis pool. For more information, see [Creating an encrypted Stratis pool](#).

#### Procedure

1. Re-create the key set using the same key description that was used previously:

```
# stratis key set --capture-key key-description
```

where *key-description* references the key that exists in the kernel keyring, which was generated when you created the encrypted Stratis pool.

2. Unlock the Stratis pool and the block device that comprise it:

```
# stratis pool unlock keyring
```

3. Verify that the Stratis pool is visible:

```
# stratis pool list
```

### 36.1.11. Unlocking an encrypted Stratis pool with Clevis

After a system reboot, your encrypted Stratis pool or the block devices that comprise it might not be visible. You can unlock an encrypted Stratis pool with the supplementary encryption mechanism that the pool is bound to.

#### Prerequisites

- Stratis v2.3.0 or later is installed. For more information, see [Installing Stratis](#).
- The **stratisd** service is running.

- You have created an encrypted Stratis pool. For more information, see [Creating an encrypted Stratis pool](#).
- The encrypted Stratis pool is bound to a supported, supplementary encryption mechanism. For more information, see [Binding an encrypted Stratis pool to NBDE](#)

or [Binding an encrypted Stratis pool to TPM](#).

### Procedure

1. Unlock the Stratis pool and the block devices that comprise it:

```
# stratis pool unlock clevis
```

2. Verify that the Stratis pool is visible:

```
# stratis pool list
```

## 36.1.12. Unbinding a Stratis pool from supplementary encryption

When you unbind an encrypted Stratis pool from a supported supplementary encryption mechanism, the primary kernel keyring encryption remains in place.

### Prerequisites

- Stratis v2.3.0 or later is installed on your system. For more information, see [Installing Stratis](#).
- You have created an encrypted Stratis pool. For more information, see [Creating an encrypted Stratis pool](#).
- The encrypted Stratis pool is bound to a supported supplementary encryption mechanism.

### Procedure

- Unbind an encrypted Stratis pool from a supplementary encryption mechanism:

```
# stratis pool unbind clevis my-pool
```

where

***my-pool*** specifies the name of the Stratis pool you want to unbind.

### Additional resources

- [Binding an encrypted Stratis pool to NBDE](#)
- [Binding an encrypted Stratis pool to TPM](#)

## 36.1.13. Starting and stopping Stratis pool

You can start and stop Stratis pools. This gives you the option to disassemble or bring down all the objects that were used to construct the pool, such as filesystems, cache devices, thin pool, and encrypted devices. Note that if the pool actively uses any device or filesystem, it might issue a warning and not be able to stop.

Stopped pools record their stopped state in their metadata. These pools do not start on the following boot, until the pool receives a start command.

If not encrypted, previously started pools automatically start on boot. Encrypted pools always need a **pool start** command on boot, as **pool unlock** is replaced by **pool start** in this version of Stratis.

### Prerequisites

- Stratis is installed. For more information, see [Installing Stratis](#).
- The **stratisd** service is running.
- You have created either an unencrypted or an encrypted Stratis pool. See [Creating an unencrypted Stratis pool](#)

or [Creating an encrypted Stratis pool](#).

### Procedure

- Use the following command to start the Stratis pool. The **--unlock-method** option specifies the method of unlocking the pool if it is encrypted:

```
# stratis pool start pool-uuid --unlock-method <keyring|clevis>
```

- Alternatively, use the following command to stop the Stratis pool. This tears down the storage stack but leaves all metadata intact:

```
# stratis pool stop pool-name
```

### Verification steps

- Use the following command to list all pools on the system:

```
# stratis pool list
```

- Use the following command to list all not previously started pools. If the UUID is specified, the command prints detailed information about the pool corresponding to the UUID:

```
# stratis pool list --stopped --uuid UUID
```

## 36.1.14. Creating a Stratis file system

Create a Stratis file system on an existing Stratis pool.

### Prerequisites

- Stratis is installed. For more information, see [Installing Stratis](#).
- The **stratisd** service is running.
- You have created a Stratis pool. See [Creating an unencrypted Stratis pool](#)

or [Creating an encrypted Stratis pool](#).

## Procedure

1. To create a Stratis file system on a pool, use:

```
# stratis filesystem create --size number-and-unit my-pool my-fs
```

where

### ***number-and-unit***

Specifies the size of a file system. The specification format must follow the standard size specification format for input, that is B, KiB, MiB, GiB, TiB or PiB.

### ***my-pool***

Specifies the name of the Stratis pool.

### ***my-fs***

Specifies an arbitrary name for the file system.

For example:

#### Example 36.1. Creating a Stratis file system

```
# stratis filesystem create --size 10GiB pool1 filesystem1
```

## Verification steps

- List file systems withing the pool to check if the Stratis filesystem is created:

```
# stratis fs list my-pool
```

## Additional resources

- [Mounting a Stratis file system](#).

### 36.1.15. Mounting a Stratis file system

Mount an existing Stratis file system to access the content.

## Prerequisites

- Stratis is installed. For more information, see [Installing Stratis](#).
- The **stratisd** service is running.
- You have created a Stratis file system. For more information, see [Creating a Stratis filesystem](#).

## Procedure

- To mount the file system, use the entries that Stratis maintains in the **/dev/stratis/** directory:

```
# mount /dev/stratis/my-pool/my-fs mount-point
```

The file system is now mounted on the *mount-point* directory and ready to use.

### Additional resources

- [Creating a Stratis file system](#).

## 36.1.16. 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 [Installing Stratis](#).
- The **stratisd** service is running.
- You have created a Stratis file system. See [Creating a Stratis filesystem](#).

### Procedure

1. Determine the UUID attribute of the file system:

```
$ lsblk --output=UUID /dev/stratis/my-pool/my-fs
```

For example:

#### Example 36.2. Viewing the UUID of Stratis file system

```
$ lsblk --output=UUID /dev/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 **xf** as the file system type and add the **x-systemd.requires=stratisd.service** option.

For example:

#### Example 36.3. 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

- [Persistently mounting file systems](#).

### 36.1.17. Setting up non-root Stratis filesystems in /etc/fstab using a systemd service

You can manage setting up non-root filesystems in /etc/fstab using a systemd service.

#### Prerequisites

- Stratis is installed. See [Installing Stratis](#).
- The **stratisd** service is running.
- You have created a Stratis file system. See [Creating a Stratis filesystem](#).

#### Procedure

- For all non-root Stratis filesystems, use:

```
# /dev/stratis/[STRATIS_SYMLINK] [MOUNT_POINT] xfs defaults, x-  
systemd.requires=stratis-fstab-setup@[POOL_UUID].service,x-systemd.after=stratis-stab-  
setup@[POOL_UUID].service <dump_value> <fsck_value>
```

#### Additional resources

- [Persistently mounting file systems](#).

## 36.2. 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.



### IMPORTANT

Stratis is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process. For more information about the support scope of Red Hat Technology Preview features, see <https://access.redhat.com/support/offerings/techpreview>.

### 36.2.1. Components of a Stratis volume

Learn about the components that comprise 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 **/dev/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.

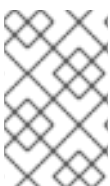
The file systems are formatted with XFS.



#### 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 **/dev/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.

## 36.2.2. 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 [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

- **stratis(8)** man page

### 36.2.3. Additional resources

- [The Stratis Storage website](#)

## 36.3. 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.



### IMPORTANT

Stratis is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process. For more information about the support scope of Red Hat Technology Preview features, see <https://access.redhat.com/support/offerings/techpreview>.

### 36.3.1. 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

- **stratis(8)** man page.

### 36.3.2. Displaying information about Stratis volumes

This procedure lists statistics about your Stratis volumes, such as the total, used, and free size of file systems and block devices belonging to a pool.



## Prerequisites

- Stratis is installed. See [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
my-pool /dev/sdb 9.10 TiB In-use Data
```

- 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 /dev/stratis/my-pool/my-fs
```

## Additional resources

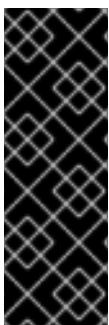
- **stratis(8)** man page.

### 36.3.3. Additional resources

- [The Stratis Storage website](#)

## 36.4. 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.



### IMPORTANT

Stratis is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process. For more information about the support scope of Red Hat Technology Preview features, see <https://access.redhat.com/support/offerings/techpreview>.

### 36.4.1. Characteristics of Stratis snapshots

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 snapshotted 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.

### 36.4.2. 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 [Installing Stratis](#).
- The **stratisd** service is running.
- You have created a Stratis file system. See [Creating a Stratis filesystem](#).

#### Procedure

- To create a Stratis snapshot, use:

```
# stratis fs snapshot my-pool my-fs my-fs-snapshot
```

#### Additional resources

- **stratis(8)** man page.

### 36.4.3. 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 [Installing Stratis](#).
- The **stratisd** service is running.
- You have created a Stratis snapshot. See [Creating a Stratis filesystem](#).

#### Procedure

- To access the snapshot, mount it as a regular file system from the `/dev/stratis/my-pool/` directory:

```
# mount /dev/stratis/my-pool/my-fs-snapshot mount-point
```

#### Additional resources

- [Mounting a Stratis file system](#).
- **mount(8)** man page.

### 36.4.4. 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 [Installing Stratis](#).
- The **stratisd** service is running.
- You have created a Stratis snapshot. See [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 /dev/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 /dev/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

- **stratis(8)** man page.

### 36.4.5. Removing a Stratis snapshot

This procedure removes a Stratis snapshot from a pool. Data on the snapshot are lost.

## Prerequisites

- Stratis is installed. See [Installing Stratis](#).
- The **stratisd** service is running.
- You have created a Stratis snapshot. See [Creating a Stratis snapshot](#).

## Procedure

1. Unmount the snapshot:

```
# umount /dev/stratis/my-pool/my-fs-snapshot
```

2. Destroy the snapshot:

```
# stratis filesystem destroy my-pool my-fs-snapshot
```

## Additional resources

- **stratis(8)** man page.

### 36.4.6. Additional resources

- [The Stratis Storage website](#)

## 36.5. REMOVING STRATIS FILE SYSTEMS

You can remove an existing Stratis file system, or a Stratis pool, by destroying data on them.



### IMPORTANT

Stratis is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process. For more information about the support scope of Red Hat Technology Preview features, see <https://access.redhat.com/support/offerings/techpreview>.

### 36.5.1. Components of a Stratis volume

Learn about the components that comprise 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 **/dev/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 **/dev/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.

## 36.5.2. Removing a Stratis file system

This procedure removes an existing Stratis file system. Data stored on it are lost.

### Prerequisites

- Stratis is installed. See [Installing Stratis](#).
- The **stratisd** service is running.
- You have created a Stratis file system. See [Creating a Stratis filesystem](#).

### Procedure

1. Unmount the file system:

```
# umount /dev/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

- **stratis(8)** man page.

## 36.5.3. Removing a Stratis pool

This procedure removes an existing Stratis pool. Data stored on it are lost.

### Prerequisites

- Stratis is installed. See [Installing Stratis](#).
- The **stratisd** service is running.
- You have created a Stratis pool:
  - To create an unencrypted pool, see [Creating an unencrypted Stratis pool](#)
  - To create an encrypted pool, see [Creating an encrypted Stratis pool](#).

### Procedure

1. List file systems on the pool:

```
# stratis filesystem list my-pool
```

2. Unmount all file systems on the pool:

```
# umount /dev/stratis/my-pool/my-fs-1 \  
          /dev/stratis/my-pool/my-fs-2 \  
          /dev/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

- **stratis(8)** man page.

## 36.5.4. Additional resources

- [The Stratis Storage website](#)

## 36.6. GETTING STARTED WITH SWAP

Use the swap space to provide temporary storage for inactive processes and data, and prevent out-of-memory errors when physical memory is full. The swap space acts as an extension to the physical memory and allows the system to continue running smoothly even when physical memory is exhausted. Note that using swap space can slow down system performance, so optimizing the use of physical memory, before relying on swap space, can be more favorable.

### 36.6.1. Overview of 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.

#### Adding swap space

The following are the different ways to add a swap space:

- [Extending swap on an LVM2 logical volume](#)
- [Creating an LVM2 logical volume for swap](#)
- [Creating a swap file](#)

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-intensive operations or run applications that require a large amount of memory.

#### Removing swap space

The following are the different ways to remove a swap space:

- [Reducing swap on an LVM2 logical volume](#)
- [Removing an LVM2 logical volume for swap](#)
- [Removing a swap file](#)

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.

### 36.6.2. Recommended system swap space

This section describes 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.

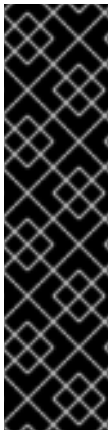
The following recommendations are especially important on systems with low memory such as 1 GB and less. Failure to allocate sufficient swap space on these systems can cause issues such as instability or even render the installed system unbootable.

**Table 36.1. Recommended 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 this table, 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.

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.

Resizing swap space requires temporarily removing the swap space from the system. This can be problematic if running applications rely on the additional swap space and might run into low-memory situations. Preferably, perform swap resizing from rescue mode, see [Debug boot options](#) in the *Performing an advanced RHEL 8 installation*. When prompted to mount the file system, select **Skip**.

### 36.6.3. 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

- You have sufficient disk space.

#### Procedure

1. Disable swapping for the associated logical volume:



```
# swapoff -v /dev/VolGroup00/LogVol01
```

2. Resize the LVM2 logical volume by 2 GB:

```
# lvresize /dev/VolGroup00/LogVol01 -L +2G
```

3. Format the new swap space:

```
# mkswap /dev/VolGroup00/LogVol01
```

4. Enable the extended logical volume:

```
# swapon -v /dev/VolGroup00/LogVol01
```

### Verification

- To test if the swap logical volume was successfully extended and activated, inspect active swap space by using the following command:

```
$ cat /proc/swaps
$ free -h
```

### 36.6.4. 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

- You have sufficient disk space.

#### Procedure

1. Create the LVM2 logical volume of size 2 GB:

```
# lvcreate VolGroup00 -n LogVol02 -L 2G
```

2. Format the new swap space:

```
# mkswap /dev/VolGroup00/LogVol02
```

3. Add the following entry to the `/etc/fstab` file:

```
/dev/VolGroup00/LogVol02 none swap defaults 0 0
```

4. Regenerate mount units so that your system registers the new configuration:

```
# systemctl daemon-reload
```

5. Activate swap on the logical volume:

```
# swapon -v /dev/VolGroup00/LogVol02
```

## Verification

- To test if the swap logical volume was successfully created and activated, inspect active swap space by using the following command:

```
$ cat /proc/swaps  
$ free -h
```

## 36.6.5. Creating a swap file

This procedure describes how to create a swap file.

### Prerequisites

- You have sufficient disk space.

### Procedure

1. 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.

2. Create an empty file:

```
# dd if=/dev/zero of=/swapfile bs=1024 count=65536
```

Replace *65536* 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. Edit the **/etc/fstab** file with the following entries to enable the swap file at boot time:

```
/swapfile none 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. Activate the swap file immediately:

```
# swapon /swapfile
```

## Verification

- To test if the new swap file was successfully created and activated, inspect active swap space by using the following command:

```
$ cat /proc/swaps  
$ free -h
```

### 36.6.6. 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
```

#### Verification

- To test if the swap logical volume was successfully reduced, inspect active swap space by using the following command:

```
$ cat /proc/swaps  
$ free -h
```

### 36.6.7. 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 none swap defaults 0 0
```

4. Regenerate mount units so that your system registers the new configuration:

```
# systemctl daemon-reload
```

## Verification

- To test if the logical volume was successfully removed, inspect active swap space by using the following command:

```
$ cat /proc/swaps  
$ free -h
```

## 36.6.8. 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 37. DEDUPLICATING AND COMPRESSING STORAGE

## 37.1. DEPLOYING VDO

As a system administrator, you can use VDO to create deduplicated and compressed storage pools.

### 37.1.1. 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. Use scripting to monitor the actual available space and generate an alert if use exceeds a threshold: for example, when the VDO volume is 80% full.

### 37.1.2. 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.

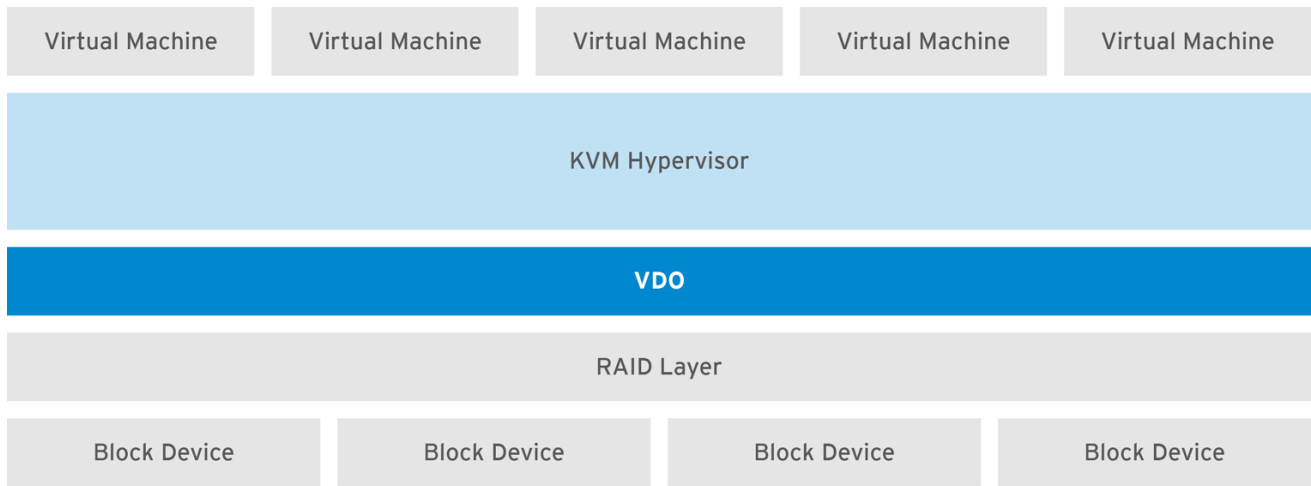


#### NOTE

Deployment of VDO volumes on top of Ceph RADOS Block Device (RBD) is currently supported. However, the deployment of Red Hat Ceph Storage cluster components on top of VDO volumes is currently not supported.

#### KVM

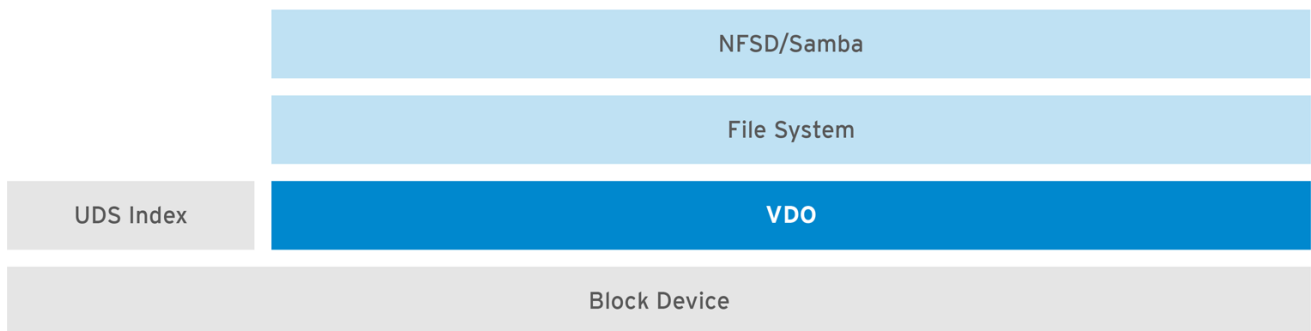
You can deploy VDO on a KVM server configured with Direct Attached Storage.



RHEL\_462492\_117

### File systems

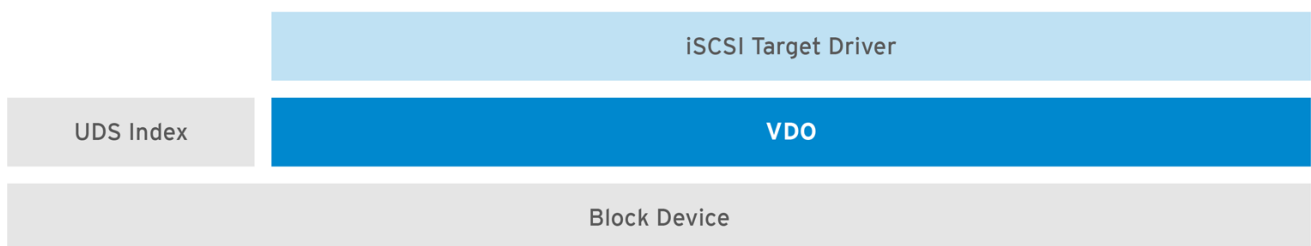
You can create file systems on top of VDO and expose them to NFS or CIFS users with the NFS server or Samba.



RHEL\_466924\_0218

### Placement of VDO on iSCSI

You can export the entirety of the VDO storage target as an iSCSI target to remote iSCSI initiators.



RHEL\_466924\_0218

When creating a VDO volume on iSCSI, you can place the VDO volume above or below the iSCSI layer. Although there are many considerations to be made, some guidelines are provided here to help you select the method that best suits your environment.

When placing the VDO volume on the iSCSI server (target) below the iSCSI layer:

- The VDO volume is transparent to the initiator, similar to other iSCSI LUNs. Hiding the thin provisioning and space savings from the client makes the appearance of the LUN easier to monitor and maintain.

- There is decreased network traffic because there are no VDO metadata reads or writes, and read verification for the dedupe advice does not occur across the network.
- The memory and CPU resources being used on the iSCSI target can result in better performance. For example, the ability to host an increased number of hypervisors because the volume reduction is happening on the iSCSI target.
- If the client implements encryption on the initiator and there is a VDO volume below the target, you will not realize any space savings.

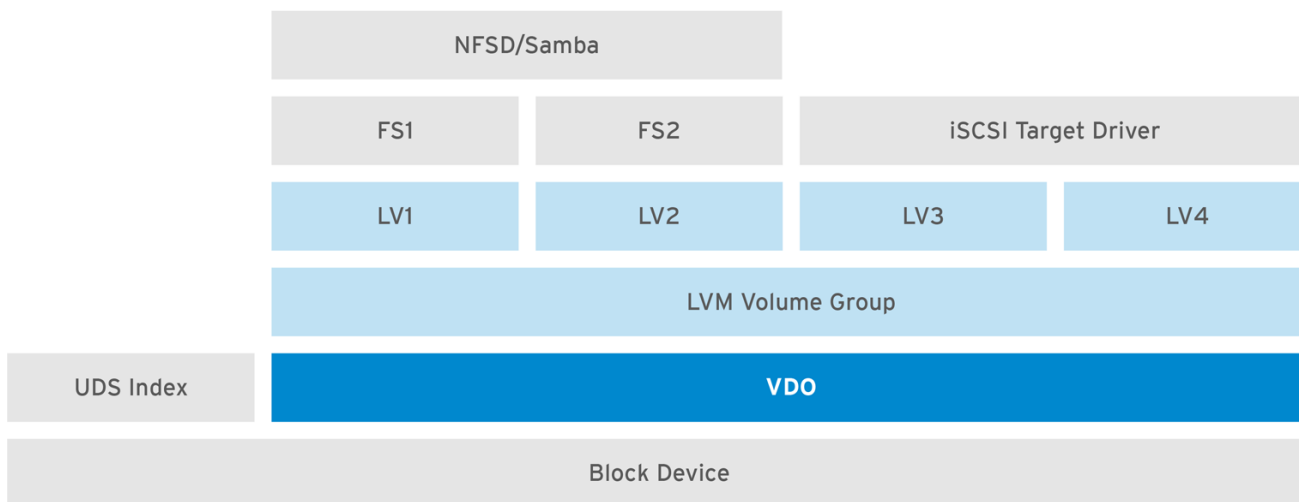
When placing the VDO volume on the iSCSI client (initiator) above the iSCSI layer:

- There is a potential for lower network traffic across the network in ASYNC mode if achieving high rates of space savings.
- You can directly view and control the space savings and monitor usage.
- If you want to encrypt the data, for example, using **dm-crypt**, you can implement VDO on top of the crypt and take advantage of space efficiency.

### 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.

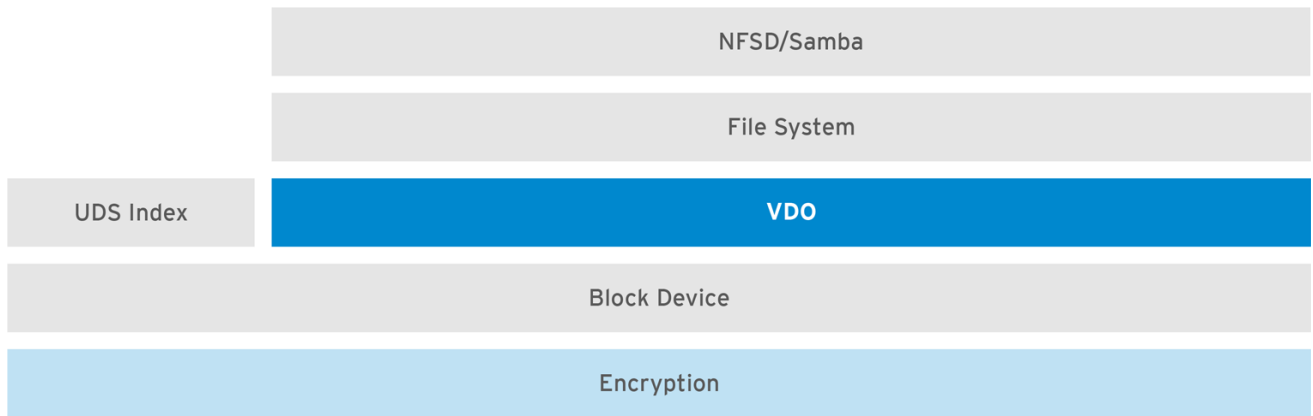


RHEL\_466924\_0218

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.

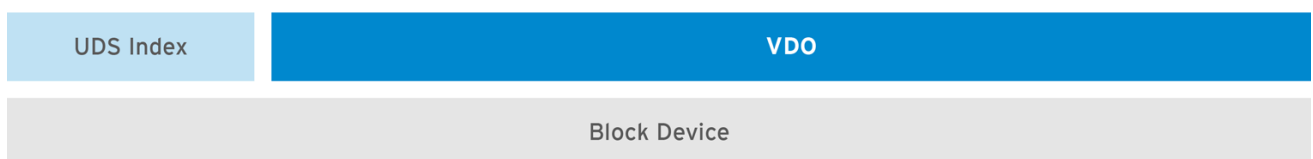
Always place the encryption layer below VDO.

When creating a VDO volume on iSCSI, you can place the VDO volume above or below the iSCSI layer. Although there are many considerations to be made, some guidelines are provided here to help you select the method that best suits your environment.

### 37.1.3. 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 37.1. 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.

### 37.1.4. The physical and logical size of a VDO volume

VDO utilizes physical, available physical, and logical size in the following ways:

#### 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

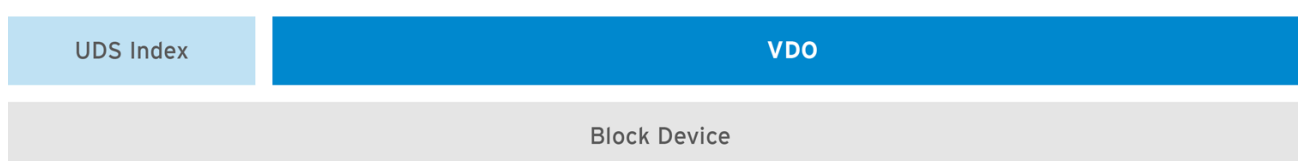
It is equivalent to the physical size minus the size of the metadata, minus the remainder after dividing the volume into slabs by the given slab size.

#### 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 37.2. 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 37.1.6.4, “Examples of VDO requirements by physical size”](#).

### 37.1.5. 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 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 37.1. 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



#### NOTE

The minimal disk usage for a VDO volume using default settings of 2 GB slab size and 0.25 dense index, requires approx 4.7 GB. This provides slightly less than 2 GB of physical data to write at 0% deduplication or compression.

Here, the minimal disk usage is the sum of the default slab size and dense index.

You can control the slab size by providing the `--config 'allocation/vdo_slab_size_mb=size-in-megabytes'` option to the `lvcreate` command.

### 37.1.6. VDO requirements

VDO has certain requirements on its placement and your system resources.

#### 37.1.6.1. VDO memory requirements

Each VDO volume has two distinct memory requirements:

##### The VDO module

VDO requires a fixed 38 MB of RAM and several variable amounts:

- 1.15 MB of RAM for each 1 MB of configured block map cache size. The block map cache requires a minimum of 150MB RAM.
- 1.6 MB of RAM for each 1 TB of logical space.
- 268 MB of RAM for each 1 TB of physical storage managed by the volume.

### The UDS index

The Universal Deduplication Service (UDS) requires a minimum of 250 MB of RAM, which is also the default amount that deduplication uses. You can configure the value when formatting a VDO volume, because the value also affects the amount of storage that the index needs.

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.



#### NOTE

The minimal disk usage for a VDO volume using default settings of 2 GB slab size and 0.25 dense index, requires approx 4.7 GB. This provides slightly less than 2 GB of physical data to write at 0% deduplication or compression.

Here, the minimal disk usage is the sum of the default slab size and dense index.

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

- [Examples of VDO requirements by physical size](#)

#### 37.1.6.2. 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

- [Examples of VDO requirements by physical size](#)
- [Slab size in VDO](#)

### 37.1.6.3. Placement of VDO in the storage stack

Place storage layers either above, or under the Virtual Data Optimizer (VDO), to fit the placement requirements.

A VDO volume is a thin-provisioned block device. You can prevent running out of physical space by placing the volume above a storage layer that you can expand at a later time. Examples of such expandable storage are Logical Volume Manager (LVM) volumes, or Multiple Device Redundant Array Inexpensive or Independent Disks (MD RAID) arrays.

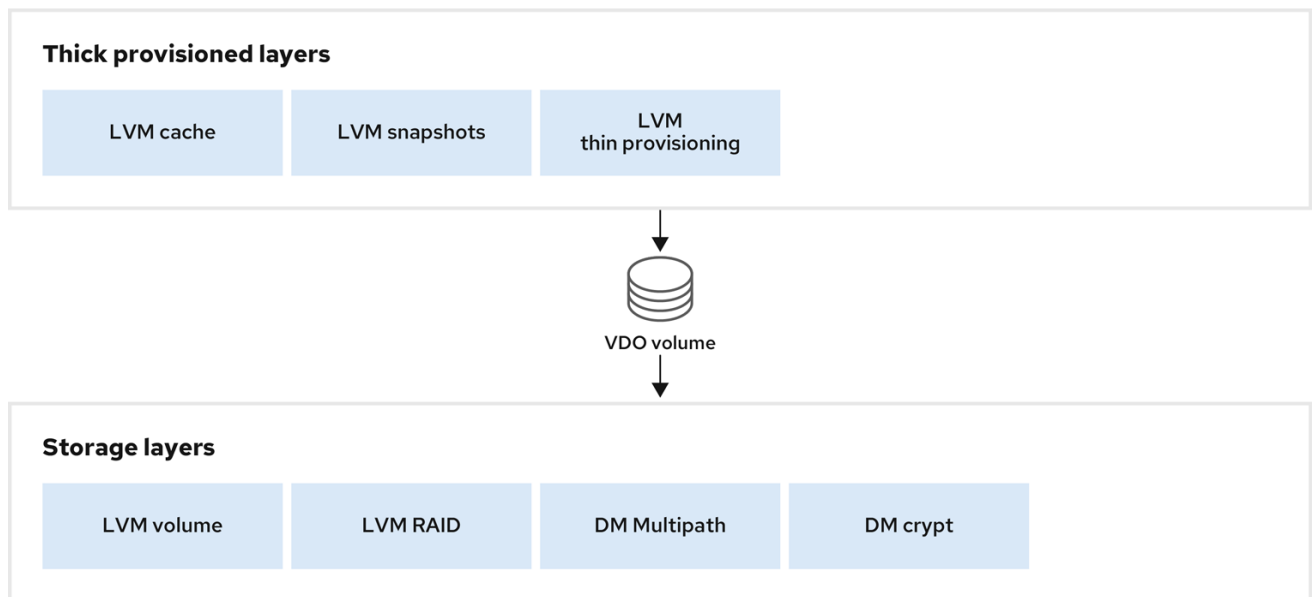
You can place thick provisioned layers above VDO. There are two aspects of thick provisioned layers that you must consider:

- Writing new data to unused logical space on a thick device. When using VDO, or other thin-provisioned storage, the device can report that it is out of space during this kind of write.
- Overwriting used logical space on a thick device with new data. When using VDO, overwriting data can also result in a report of the device being out of space.

These limitations affect all layers above the VDO layer. If you do not monitor the VDO device, you can unexpectedly run out of physical space on the thick-provisioned volumes above VDO.

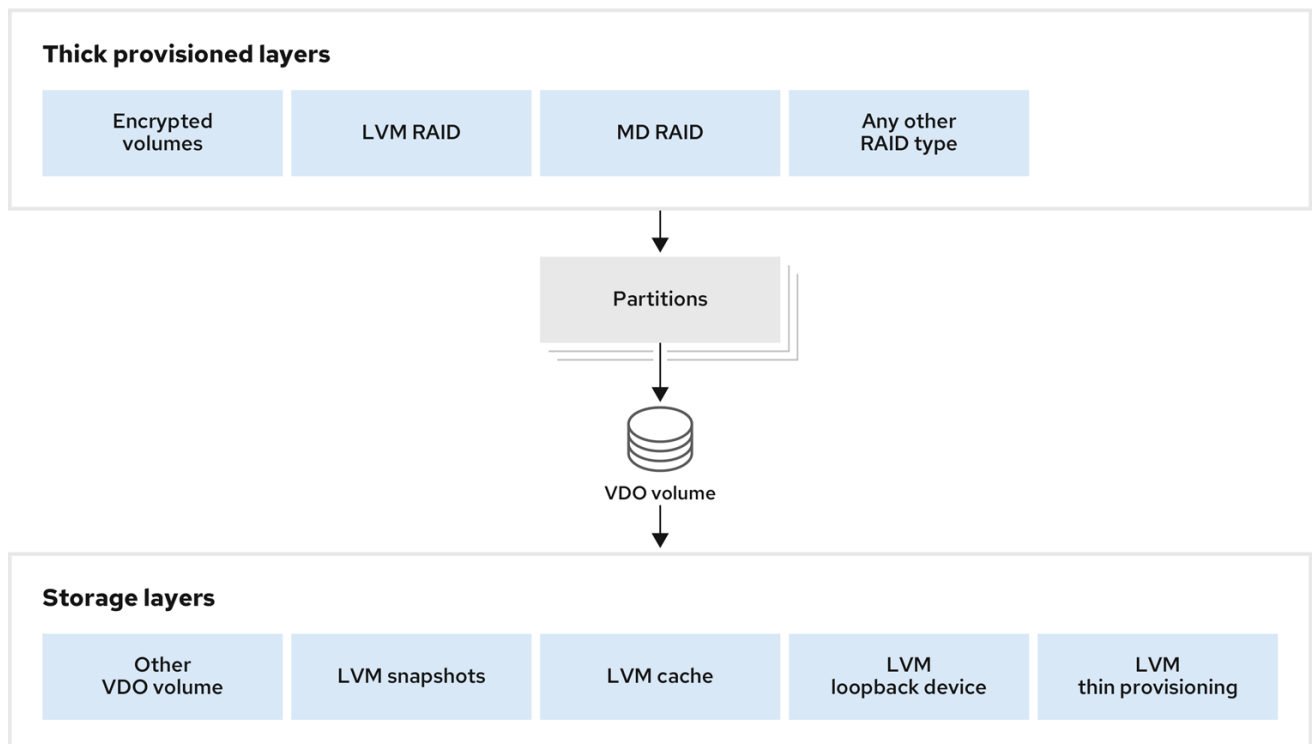
See the following examples of supported and unsupported VDO volume configurations.

Figure 37.3. Supported VDO volume configurations



309\_RHEL\_0223

Figure 37.4. Unsupported VDO volume configurations



309\_RHEL\_0223

### Additional resources

- For more information about stacking VDO with LVM layers, see the [Stacking LVM volumes](#) article.

### 37.1.6.4. Examples of VDO requirements by physical size

The following tables provide approximate system requirements of VDO based on the physical size of the underlying 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 size.

**Table 37.2. Storage and memory requirements for primary storage**

Physical size	RAM usage: UDS	RAM usage: VDO	Disk usage	Index type
10GB-1TB	250MB	472MB	2.5GB	Dense
2-10TB	1GB	3GB	10GB	Dense
	250MB		22GB	Sparse
11-50TB	2GB	14GB	170GB	Sparse
51-100TB	3GB	27GB	255GB	Sparse
101-256TB	12GB	69GB	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 size. If you expect the backup set or the physical size to grow in the future, factor this into the index size.

**Table 37.3. Storage and memory requirements for backup storage**

Physical size	RAM usage: UDS	RAM usage: VDO	Disk usage	Index type
10GB-1TB	250MB	472MB	2.5 GB	Dense
2-10TB	2GB	3GB	170GB	Sparse
11-50TB	10GB	14GB	850GB	Sparse
51-100TB	20GB	27GB	1700GB	Sparse
101-256TB	26GB	69GB	3400GB	Sparse

## 37.1.7. Installing VDO

This procedure installs software necessary to create, mount, and manage VDO volumes.

## Procedure

- Install the VDO software:

```
# yum install lvm2 kmod-kvdo vdo
```

### 37.1.8. Creating a VDO volume

This procedure creates a VDO volume on a block device.

#### Prerequisites

- Install the VDO software. See [Section 37.1.7, “Installing VDO”](#).
- Use expandable storage as the backing block device. For more information, see [Section 37.1.6.3, “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.

1. Find a persistent name for the block device where you want to create the VDO volume. For more information on persistent names, see [Chapter 26, 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.
2. 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 37.1. 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 [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
```



#### NOTE

The purpose of the **-K** and **-E nodiscard** options on a freshly created VDO volume is to not spend time sending requests, as it has no effect on an un-allocated block. A fresh VDO volume starts out 100% un-allocated.

4. Use the following command to wait for the system to register the new device node:

```
# udevadm settle
```

#### Next steps

1. Mount the file system. See [Section 37.1.9, “Mounting a VDO volume”](#) for details.
2. Enable the **discard** feature for the file system on your VDO device. See [Section 37.1.10, “Enabling periodic block discard”](#) for details.

#### Additional resources

- The **vdo(8)** man page

### 37.1.9. 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 37.1.8, “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 0 0
```

- For the ext4 file system:

```
/dev/mapper/vdo-name mount-point ext4 defaults 0 0
```

If the VDO volume is located on a block device that requires network, such as iSCSI, add the `_netdev` mount option.

## Additional resources

- The `vdo(8)` man page.
- For iSCSI and other block devices requiring network, see the `systemd.mount(5)` man page for information on the `_netdev` mount option.

### 37.1.10. 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
```

### 37.1.11. Monitoring VDO

This procedure describes how to obtain usage and efficiency information from a VDO volume.

## Prerequisites

- Install the VDO software. See [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.

## 37.2. 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 [Section 37.1, “Deploying VDO”](#).

### 37.2.1. 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.

#### 37.2.1.1. The physical and logical size of a VDO volume

VDO utilizes physical, available physical, and logical size in the following ways:

##### 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

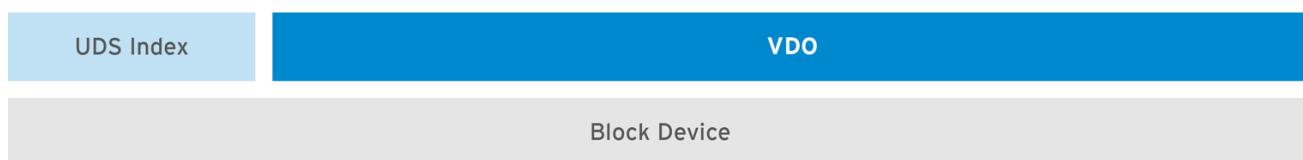
It is equivalent to the physical size minus the size of the metadata, minus the remainder after dividing the volume into slabs by the given slab size.

##### 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 37.5. 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 37.1.6.4, “Examples of VDO requirements by physical size”](#).

### 37.2.1.2. 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.

### 37.2.1.3. Monitoring VDO

This procedure describes how to obtain usage and efficiency information from a VDO volume.

#### Prerequisites

- Install the VDO software. See [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.

### 37.2.1.4. 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 to fill the free space and then delete that file.

#### 37.2.1.5. 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

#### 37.2.1.6. 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.

#### 37.2.2. Starting or stopping VDO volumes

You can start or stop a given VDO volume, or all VDO volumes, and their associated UDS indexes.

### 37.2.2.1. 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.
- The volume additionally writes the amount of data equal to the block map cache size plus up to 8MiB per slab.
- The volume must finish processing all outstanding IO requests.

### 37.2.2.2. 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

### 37.2.2.3. 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

#### 37.2.2.4. Additional resources

- If restarted after an unclean shutdown, VDO performs a rebuild to verify the consistency of its metadata and repairs it if necessary. See [Section 37.2.5, “Recovering a VDO volume after an unclean shutdown”](#) for more information on the rebuild process.

### 37.2.3. 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.

#### 37.2.3.1. 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.
- The volume additionally writes the amount of data equal to the block map cache size plus up to 8MiB per slab.
- The volume must finish processing all outstanding IO requests.

### 37.2.3.2. 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

### 37.2.3.3. 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

### 37.2.4. 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.



### 37.2.4.1. 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.

#### **async-unsafe**

This mode has the same properties as **async** but it is not compliant with Atomicity, Consistency, Isolation, Durability (ACID). Compared to **async**, **async-unsafe** has a better performance.



#### **WARNING**

When an application or a file system that assumes ACID compliance operates on top of the VDO volume, **async-unsafe** 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.

### 37.2.4.2. The internal processing of VDO write modes

The write modes for VDO are **sync** and **async**. The following information describes the operations of these modes.

If the **kvdo** module is operating in synchronous ( **synch** ) 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 ( **async** ) 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.

### 37.2.4.3. 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**

### 37.2.4.4. 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
```

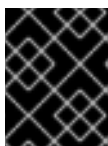
2. In the previous examples:

- 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**.

#### 37.2.4.5. 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 37.2.4.4, “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|async-unsafe|auto \  
--name=vdo-name
```

- To specify a write mode when creating a VDO volume, add the **--writePolicy=*sync|async|async-unsafe|auto*** option to the **vdo create** command.

#### 37.2.5. 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.

### 37.2.5.1. 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.

#### **async-unsafe**

This mode has the same properties as **async** but it is not compliant with Atomicity, Consistency, Isolation, Durability (ACID). Compared to **async**, **async-unsafe** has a better performance.



#### **WARNING**

When an application or a file system that assumes ACID compliance operates on top of the VDO volume, **async-unsafe** 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.

### 37.2.5.2. 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 the 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 37.2.5.3, “VDO operating modes”](#).

### 37.2.5.3. 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 37.2.5.4, “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 37.2.5.5, “Forcing an offline rebuild of a VDO volume metadata”](#).

### 37.2.5.4. 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.

### 37.2.5.5. 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 37.2.5.4, “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
```

**37.2.5.6. 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:

```
[...]
```

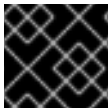
```

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

```

### 37.2.6. Optimizing the UDS index

You can configure certain settings of the UDS index to optimize it on your system.



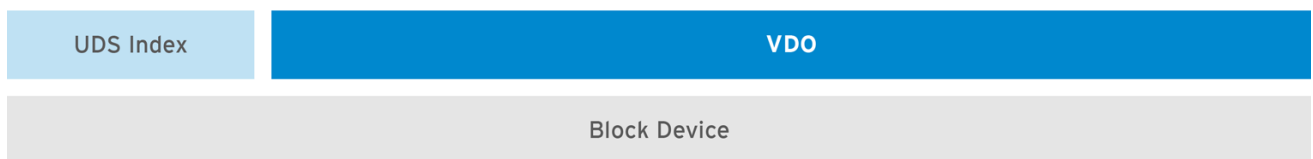
#### IMPORTANT

You cannot change the properties of the UDS index *after* creating the VDO volume.

#### 37.2.6.1. 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 37.6. 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.

#### 37.2.6.2. 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.

#### 37.2.6.3. 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 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.

### 37.2.7. 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.

#### 37.2.7.1. 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.

#### 37.2.7.2. 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
```

### 37.2.7.3. 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.

### 37.2.8. Enabling or disabling compression in VDO

VDO provides data compression. Disabling it can maximize performance and speed up processing of data that is unlikely to compress. Re-enabling it can increase space savings.

#### 37.2.8.1. 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.

### 37.2.8.2. 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
```

### 37.2.8.3. 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.

## 37.2.9. 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.

### 37.2.9.1. The physical and logical size of a VDO volume

VDO utilizes physical, available physical, and logical size in the following ways:

#### 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

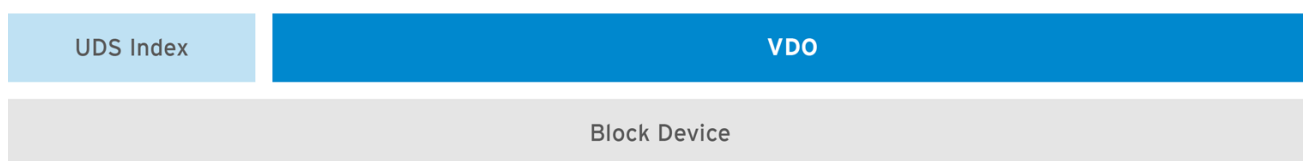
It is equivalent to the physical size minus the size of the metadata, minus the remainder after dividing the volume into slabs by the given slab size.

## 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 37.7. 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 37.1.6.4, “Examples of VDO requirements by physical size”](#).

### 37.2.9.2. 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.

### 37.2.9.3. 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.

#### 37.2.9.4. 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
```

#### 37.2.10. Removing VDO volumes

You can remove an existing VDO volume on your system.

##### 37.2.10.1. 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
```

##### 37.2.10.2. 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
```

### 37.2.11. Additional resources

- 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)

## 37.3. DISCARDING UNUSED BLOCKS

You can perform or schedule discard operations on block devices that support them.

### 37.3.1. 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.

### 37.3.2. 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.

**37.3.3. 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

- **fstrim(8)** man page.

### 37.3.4. 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

- **mount(8)** man page.
- **fstab(5)** man page.

### 37.3.5. 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
```

## 37.4. MANAGING VIRTUAL DATA OPTIMIZER VOLUMES USING THE WEB CONSOLE

Configure the Virtual Data Optimizer (VDO) using the RHEL 8 web console.

You will learn how 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](#).
- The **cockpit-storaged** package is installed on your system.

### 37.4.1. 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 compression in VDO](#).

#### Thin provisioning

For details, see [Creating and managing thin provisioned 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](#).

### 37.4.2. Creating VDO volumes in the web console

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 **+** button 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 compression in VDO](#).
10. [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.
11. Click **Create**.

## Verification steps

- Check that you can see the new VDO volume in the **Storage** section. Then you can format it with a file system.

### 37.4.3. 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 [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.

### 37.4.4. Extending VDO volumes in the web console

Extend VDO volumes in the RHEL 8 web console.

#### Prerequisites

- The **cockpit-storaged** package is installed on your system.
- 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.
  1. Click **Grow**.

#### Verification steps

- Check the VDO volume details for the new size to verify that your changes have been successful.

## PART V. DESIGN OF LOG FILE

## CHAPTER 38. 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.

### 38.1. 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 **nftables**, **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.

## 38.2. 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.

## 38.3. CONFIGURING AUDITD FOR A SECURE ENVIRONMENT

The default **auditd** configuration should be suitable for most environments. However, if your environment must 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. The **max\_log\_file** parameter specifies the maximum file size in megabytes. The value given must be numeric.

### 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. If the value of **space\_left** is specified as a whole number, it is interpreted as an absolute size in megabytes (MiB). If the value is specified as a number between 1 and 99 followed by a percentage sign (for example, 5%), the Audit daemon calculates the absolute size in megabytes based on the size of the file system containing **log\_file**.

### 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. The numeric value for this parameter should be lower than the number for **space\_left**. You can also append a percent sign (for example, 1%) to the number to have the audit daemon calculate the number based on the disk partition size.

### 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.

## 38.4. STARTING AND CONTROLLING AUDITD

After **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
```

You can temporarily disable **auditd** with the **# auditctl -e 0** command and re-enable it with **# auditctl -e 1**.

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**.

## 38.5. 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=7fffd19c5592 a1=0 a2=7fffd19c4b50 a3=a items=1 ppid=2686 pid=3538 auid=1000 uid=1000
gid=1000 euid=1000 suid=1000 fsuid=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=636174002F6574632F7373682F737368645F636F6E6666967
```

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=7fffd19c5592, a1=0, a2=7fffd19c5592, 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.

### **audit=1000**

The **audit** 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 **uid** 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.

#### **oid=0**

The **oid** 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\_fi=none**

The **cap\_fi** 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**.

## 38.6. 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

- **auditctl(8)** man page.

## 38.7. 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.

Furthermore, you can use the **auditctl** command to read rules from a specified file using the **-R** option, for example:

```
# auditctl -R /usr/share/audit/sample-rules/30-stig.rules
```

## 38.8. USING PRE-CONFIGURED RULES FILES

In the **/usr/share/audit/sample-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:

```
# cd /usr/share/audit/sample-rules/  
# cp 10-base-config.rules 30-stig.rules 31-privileged.rules 99-finalize.rules /etc/audit/rules.d/  
# augenrules --load
```

You can order Audit rules using a numbering scheme. See the `/usr/share/audit/sample-rules/README-rules` file for more information.

#### Additional resources

- `audit.rules(7)` man page.

## 38.9. 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  
...
```

#### Additional resources

- `audit.rules(8)` and `augenrules(8)` man pages.

## 38.10. DISABLING AUGENRULES

Use the following steps to disable the **augenrules** utility. This switches Audit to use rules defined in the **/etc/audit/audit.rules** file.

### Procedure

1. Copy the **/usr/lib/systemd/system/auditd.service** file to the **/etc/systemd/system/** directory:

```
# cp -f /usr/lib/systemd/system/auditd.service /etc/systemd/system/
```

2. Edit the **/etc/systemd/system/auditd.service** file in a text editor of your choice, for example:

```
# vi /etc/systemd/system/auditd.service
```

3. Comment out the line containing **augenrules**, and uncomment the line containing the **auditctl -R** command:

```
#ExecStartPost=-/sbin/augenrules --load  
ExecStartPost=-/sbin/auditctl -R /etc/audit/audit.rules
```

4. Reload the **systemd** daemon to fetch changes in the **auditd.service** file:

```
# systemctl daemon-reload
```

5. Restart the **auditd** service:

```
# service auditd restart
```

### Additional resources

- **augenrules(8)** and **audit.rules(8)** man pages.
- [Auditd service restart overrides changes made to /etc/audit/audit.rules](#) .

## 38.11. SETTING UP AUDIT TO MONITOR SOFTWARE UPDATES

In RHEL 8.6 and later versions, you can use the pre-configured rule **44-installers.rules** to configure Audit to monitor the following utilities that install software:

- **dnf** <sup>[3]</sup>
- **yum**
- **pip**
- **npm**
- **cpan**
- **gem**
- **luarocks**

By default, **rpm** already provides audit **SOFTWARE\_UPDATE** events when it installs or updates a package. You can list them by entering **ausearch -m SOFTWARE\_UPDATE** on the command line.

In RHEL 8.5 and earlier versions, you can manually add rules to monitor utilities that install software into a **.rules** file within the **/etc/audit/rules.d/** directory.



## NOTE

Pre-configured rule files cannot be used on systems with the **ppc64le** and **aarch64** architectures.

## Prerequisites

- **auditd** is configured in accordance with the settings provided in [Configuring auditd for a secure environment](#).

## Procedure

1. On RHEL 8.6 and later, copy the pre-configured rule file **44-installers.rules** from the **/usr/share/audit/sample-rules/** directory to the **/etc/audit/rules.d/** directory:

```
# cp /usr/share/audit/sample-rules/44-installers.rules /etc/audit/rules.d/
```

On RHEL 8.5 and earlier, create a new file in the **/etc/audit/rules.d/** directory named **44-installers.rules**, and insert the following rules:

```
-a always,exit -F perm=x -F path=/usr/bin/dnf-3 -F key=software-installer  
-a always,exit -F perm=x -F path=/usr/bin/yum -F
```

You can add additional rules for other utilities that install software, for example **pip** and **npm**, using the same syntax.

2. Load the audit rules:

```
# augenrules --load
```

## Verification

1. List the loaded rules:

```
# auditctl -l  
-p x-w /usr/bin/dnf-3 -k software-installer  
-p x-w /usr/bin/yum -k software-installer  
-p x-w /usr/bin/pip -k software-installer  
-p x-w /usr/bin/npm -k software-installer  
-p x-w /usr/bin/cpan -k software-installer  
-p x-w /usr/bin/gem -k software-installer  
-p x-w /usr/bin/luarocks -k software-installer
```

2. Perform an installation, for example:

```
# yum reinstall -y vim-enhanced
```

3. Search the Audit log for recent installation events, for example:

```
# ausearch -ts recent -k software-installer
_____
time->Thu Dec 16 10:33:46 2021
type=PROCTITLE msg=audit(1639668826.074:298):
proctitle=2F7573722F6C6962657865632F706C6174666F726D2D707974686F6E002F75737
22F62696E2F646E66007265696E7374616C6C002D790076696D2D656E68616E636564
type=PATH msg=audit(1639668826.074:298): item=2 name="/lib64/ld-linux-x86-64.so.2"
inode=10092 dev=fd:01 mode=0100755 ouid=0 ogid=0 rdev=00:00
obj=system_u:object_r:ld_so_t:s0 nametype=NORMAL cap_fp=0 cap_fi=0 cap_fe=0
cap_fver=0 cap_frootid=0
type=PATH msg=audit(1639668826.074:298): item=1 name="/usr/libexec/platform-python"
inode=4618433 dev=fd:01 mode=0100755 ouid=0 ogid=0 rdev=00:00
obj=system_u:object_r:bin_t:s0 nametype=NORMAL cap_fp=0 cap_fi=0 cap_fe=0
cap_fver=0 cap_frootid=0
type=PATH msg=audit(1639668826.074:298): item=0 name="/usr/bin/dnf" inode=6886099
dev=fd:01 mode=0100755 ouid=0 ogid=0 rdev=00:00 obj=system_u:object_r:rpm_exec_t:s0
nametype=NORMAL cap_fp=0 cap_fi=0 cap_fe=0 cap_fver=0 cap_frootid=0
type=CWD msg=audit(1639668826.074:298): cwd="/root"
type=EXECVE msg=audit(1639668826.074:298): argc=5 a0="/usr/libexec/platform-python"
a1="/usr/bin/dnf" a2="reinstall" a3="-y" a4="vim-enhanced"
type=SYSCALL msg=audit(1639668826.074:298): arch=c000003e syscall=59 success=yes
exit=0 a0=55c437f22b20 a1=55c437f2c9d0 a2=55c437f2aeb0 a3=8 items=3 ppid=5256
pid=5375 auid=0 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=3
comm="dnf" exe="/usr/libexec/platform-python3.6"
subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 key="software-installer"
```

## 38.12. MONITORING USER LOGIN TIMES WITH AUDIT

To monitor which users logged in at specific times, you do not need to configure Audit in any special way. You can use the **ausearch** or **aureport** tools, which provide different ways of presenting the same information.

### Prerequisites

- **auditd** is configured in accordance with the settings provided in [Configuring auditd for a secure environment](#).

### Procedure

To display user log in times, use any one of the following commands:

- Search the audit log for the **USER\_LOGIN** message type:

```
# ausearch -m USER_LOGIN -ts '12/02/2020' '18:00:00' -sv no
time->Mon Nov 22 07:33:22 2021
type=USER_LOGIN msg=audit(1637584402.416:92): pid=1939 uid=0 auid=4294967295
ses=4294967295 subj=system_u:system_r:sshd_t:s0-s0:c0.c1023 msg='op=login acct="
(unknown)" exe="/usr/sbin/sshd" hostname=? addr=10.37.128.108 terminal=ssh res=failed'
```

- You can specify the date and time with the **-ts** option. If you do not use this option, **ausearch** provides results from today, and if you omit time, **ausearch** provides results from midnight.

- You can use the **-sv yes** option to filter out successful login attempts and **-sv no** for unsuccessful login attempts.
- Pipe the raw output of the **ausearch** command into the **aulast** utility, which displays the output in a format similar to the output of the **last** command. For example:

```
# ausearch --raw | aulast --stdin
root  ssh      10.37.128.108  Mon Nov 22 07:33 - 07:33 (00:00)
root  ssh      10.37.128.108  Mon Nov 22 07:33 - 07:33 (00:00)
root  ssh      10.22.16.106   Mon Nov 22 07:40 - 07:40 (00:00)
reboot system boot 4.18.0-348.6.el8 Mon Nov 22 07:33
```

- Display the list of login events by using the **aureport** command with the **--login -i** options.

```
# aureport --login -i

Login Report
=====
# date time auid host term exe success event
=====
1. 11/16/2021 13:11:30 root 10.40.192.190 ssh /usr/sbin/sshd yes 6920
2. 11/16/2021 13:11:31 root 10.40.192.190 ssh /usr/sbin/sshd yes 6925
3. 11/16/2021 13:11:31 root 10.40.192.190 ssh /usr/sbin/sshd yes 6930
4. 11/16/2021 13:11:31 root 10.40.192.190 ssh /usr/sbin/sshd yes 6935
5. 11/16/2021 13:11:33 root 10.40.192.190 ssh /usr/sbin/sshd yes 6940
6. 11/16/2021 13:11:33 root 10.40.192.190 /dev/pts/0 /usr/sbin/sshd yes 6945
```

### Additional resources

- The **ausearch(8)** man page.
- The **aulast(8)** man page.
- The **aureport(8)** man page.

## 38.13. ADDITIONAL RESOURCES

- The [RHEL Audit System Reference](#) Knowledgebase article.
- The [Auditd execution options in a container](#) Knowledgebase article.
- The [Linux Audit Documentation Project](#) page .
- The **audit** package provides documentation in the **/usr/share/doc/audit/** directory.
- **auditd(8)**, **auditctl(8)**, **ausearch(8)**, **audit.rules(7)**, **audispd.conf(5)**, **audispd(8)**, **auditd.conf(5)**, **ausearch-expression(5)**, **aulast(8)**, **aulastlog(8)**, **aureport(8)**, **ausyscall(8)**, **autrace(8)**, and **auvirt(8)** man pages.

[3] Because **dnf** is a symlink in RHEL, the path in the **dnf** Audit rule must include the target of the symlink. To receive correct Audit events, modify the **44-installers.rules** file by changing the **path=/usr/bin/dnf** path to **/usr/bin/dnf-3**.

## PART VI. DESIGN OF KERNEL

## CHAPTER 39. THE LINUX KERNEL

Learn about the Linux kernel and the Linux kernel RPM package provided and maintained by Red Hat (Red Hat kernel). Keep the Red Hat kernel updated, which ensures the operating system has all the latest bug fixes, performance enhancements, and patches, and is compatible with new hardware.

### 39.1. WHAT THE KERNEL IS

The kernel is a core part of a Linux operating system that 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 easily upgraded and verified by the **yum** package manager.



#### WARNING

Kernels that have not been compiled by Red Hat are **not** supported by Red Hat.

### 39.2. RPM PACKAGES

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.



## 39.3. 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 required sub-packages are properly installed:

- **kernel-core** - contains the binary image of the kernel, all **initramfs**-related objects to bootstrap the system, and a minimal number of kernel modules to ensure 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 the remaining kernel modules that are not present in **kernel-core**.

The small set of **kernel** sub-packages above aims to provide a reduced maintenance surface to system administrators especially in virtualized and cloud environments.

Optional kernel packages are for example:

- **kernel-modules-extra** - contains kernel modules for rare hardware and modules which loading is disabled by default.
- **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-stablelists** - contains information pertaining to the RHEL 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.

### Additional resources

- [What are the kernel-core, kernel-modules, and kernel-modules-extras packages?](#)

## 39.4. DISPLAYING CONTENTS OF THE KERNEL PACKAGE

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

- The **rpm(8)** manual page
- [RPM packages](#)

## 39.5. UPDATING THE KERNEL

Update the kernel using the **yum** package manager.

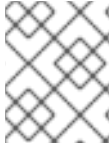
### Procedure

1. To update the kernel, enter the following command:

```
# 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 RHEL 7 to RHEL 8, follow relevant sections of the [Upgrading from RHEL 7 to RHEL 8](#) document.

#### Additional resources

- [Managing software packages](#)

## 39.6. INSTALLING SPECIFIC KERNEL VERSIONS

Install new kernels using the **yum** package manager.

#### Procedure

- To install a specific kernel version, enter the following command:

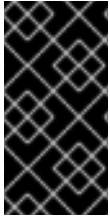
```
# yum install kernel-{version}
```

#### Additional resources

- [Red Hat Code Browser](#)
- [Red Hat Enterprise Linux Release Dates](#)

# CHAPTER 40. 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.

## 40.1. 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 GRUB bootloader are defined in the **kernelopts** variable of the **/boot/grub2/grubenv** file for each kernel boot entry.



## 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

- **kernel-command-line(7)**, **bootparam(7)** and **dracut.cmdline(7)** manual pages
- [How to install and boot custom kernels in Red Hat Enterprise Linux 8](#)

## 40.2. WHAT GRUBBY IS

**grubby** is a utility for manipulating boot loader configuration files.

You can also use **grubby** for changing the default boot entry, and for adding or removing arguments from a GRUB2 menu entry.

### Additional resources

- The **grubby(8)** manual page

### 40.3. 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 example 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

- [How to install and boot custom kernels in Red Hat Enterprise Linux 8](#)

### 40.4. CHANGING KERNEL COMMAND-LINE PARAMETERS FOR ALL BOOT ENTRIES

Change kernel command-line parameters for all boot entries on your system.

#### Prerequisites

- Verify that the **grubby** utility is installed on your system.
- Verify that the **zipl** utility is installed on your IBM Z system.

#### Procedure

- To add a parameter:

```
# grubby --update-kernel=ALL --args="<NEW_PARAMETER>"
```

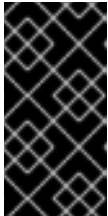
For systems that use the GRUB bootloader, the command updates the **/boot/grub2/grubenv** file by adding a new kernel parameter to the **kernelopts** variable in that file.

- On IBM Z, execute the **zipl** command with no options to update the boot menu.
- To remove a parameter:
  -

```
# grubby --update-kernel=ALL --remove-args="<PARAMETER_TO_REMOVE>"
```

- On IBM Z, execute the **zipl** command with no options to update the boot menu.
- After each update of your kernel package, propagate the configured kernel options to the new kernels:

```
# grub2-mkconfig -o /etc/grub2.cfg
```



### IMPORTANT

Newly installed kernels do not inherit the kernel command-line parameters from your previously configured kernels. You must run the **grub2-mkconfig** command on the newly installed kernel to propagate the needed parameters to your new kernel.

### Additional resources

- [Understanding kernel command-line parameters](#)
- **grubby(8)** and **zipl(8)** manual pages
- [grubby tool](#)

## 40.5. CHANGING KERNEL COMMAND-LINE PARAMETERS FOR A SINGLE BOOT ENTRY

Make changes in 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:

```
# grubby --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:

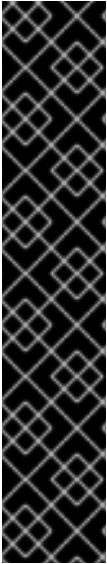
```
# grubby --update-kernel=/boot/vmlinuz-$(uname -r) --remove-args="<PARAMETER_TO_REMOVE>"
```

- 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 GRUB2 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

- [Understanding kernel command-line parameters](#)
- **grubby(8)** and **zipl(8)** manual pages
- [grubby tool](#)

## 40.6. CHANGING KERNEL COMMAND-LINE PARAMETERS TEMPORARILY AT BOOT TIME

Make temporary changes to a Kernel Menu Entry by changing the kernel parameters only during a single boot process.

### Procedure

1. Select the kernel you want to start when the GRUB 2 boot menu appears and press the **e** key to edit the kernel parameters.
2. Find the kernel command line by moving the cursor down. The kernel command line starts with **linux** on 64-Bit IBM Power Series and x86-64 BIOS-based systems, or **linuxefi** on UEFI systems.
3. Move the cursor to the end of the line.



### NOTE

Press **Ctrl+a** to jump to the start of the line and **Ctrl+e** to jump to the end of the line. On some systems, **Home** and **End** keys might also work.

4. Edit the kernel parameters as required. For example, to run the system in emergency mode, add the *emergency* parameter at the end of the **linux** line:

```
linux ($root)/vmlinuz-4.18.0-348.12.2.el8_5.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 emergency
```

To enable the system messages, remove the **rhgb** and **quiet** parameters.

5. Press **Ctrl+x** to boot with the selected kernel and the modified command line parameters.



### IMPORTANT

Press **Esc** key to leave command line editing and it will drop all the user made changes.



### NOTE

This procedure applies only for a single boot and does not persistently make the changes.

## 40.7. CONFIGURING GRUB SETTINGS TO ENABLE SERIAL CONSOLE CONNECTION

The serial console is beneficial when you need to connect to a headless server or an embedded system and the network is down. Or when you need to avoid security rules and obtain login access on a different system.

You need to configure some default GRUB settings to use the serial console connection.

### Prerequisites

- You have root permissions.

### Procedure

1. Add the following two lines to the **/etc/default/grub** file:

```
GRUB_TERMINAL="serial"  
GRUB_SERIAL_COMMAND="serial --speed=9600 --unit=0 --word=8 --parity=no --stop=1"
```

The first line disables the graphical terminal. The **GRUB\_TERMINAL** key overrides values of **GRUB\_TERMINAL\_INPUT** and **GRUB\_TERMINAL\_OUTPUT** keys.

The second line adjusts the baud rate (**--speed**), parity and other values to fit your environment and hardware. Note that a much higher baud rate, for example 115200, is preferable for tasks such as following log files.

2. Update the GRUB configuration file.

- On BIOS-based machines:

```
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

- On UEFI-based machines:

```
# grub2-mkconfig -o /boot/efi/EFI/redhat/grub.cfg
```

3. Reboot the system for the changes to take effect.



# CHAPTER 41. 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. 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.

## 41.1. 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 41.1. 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



## 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.

### Additional resources

- [sysctl\(8\)](#), and [sysctl.d\(5\)](#) manual pages

## 41.2. CONFIGURING KERNEL PARAMETERS TEMPORARILY WITH SYSCTL

Use the **sysctl** command to temporarily set kernel parameters at runtime. The command is also useful for listing and filtering tunables.

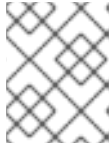
### Prerequisites

- Root permissions

### Procedure

1. List all parameters and their values.

```
# 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, enter:

```
# sysctl <TUNABLE_CLASS>.<PARAMETER>=<TARGET_VALUE>
```

The sample command above changes the parameter value while the system is running. The changes take effect immediately, without a need for restart.



### NOTE

The changes return back to default after your system reboots.

### Additional resources

- The [sysctl\(8\)](#) manual page
- [Configuring kernel parameters permanently with sysctl](#)
- [Using configuration files in /etc/sysctl.d/ to adjust kernel parameters](#)

## 41.3. CONFIGURING KERNEL PARAMETERS PERMANENTLY WITH SYSCTL

Use the **sysctl** command to permanently set kernel parameters.

### Prerequisites

- Root permissions

### Procedure

1. List all parameters.

```
# sysctl -a
```

The command displays all kernel parameters that can be configured at runtime.

2. 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

- The **sysctl(8)** and **sysctl.conf(5)** manual pages
- [Using configuration files in /etc/sysctl.d/ to adjust kernel parameters](#)

## 41.4. USING CONFIGURATION FILES IN /ETC/SYSCTL.D/ TO ADJUST KERNEL PARAMETERS

Modify configuration files in the **/etc/sysctl.d/** directory manually to permanently set kernel parameters.

### Prerequisites

- 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.

```
<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, enter:

```
# 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

- [sysctl\(8\), sysctl.d\(5\) manual pages](#)

## 41.5. CONFIGURING KERNEL PARAMETERS TEMPORARILY THROUGH /PROC/SYS/

Set kernel parameters temporarily through the files in the **/proc/sys/** virtual file system directory.

#### Prerequisites

- 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

- [Configuring kernel parameters permanently with sysctl](#)
- [Using configuration files in /etc/sysctl.d/ to adjust kernel parameters](#)

## CHAPTER 42. INSTALLING AND CONFIGURING KDUMP

### 42.1. INSTALLING KDUMP

The **kdump** service is installed and activated by default on the new Red Hat Enterprise Linux installations. Learn about **kdump** and how to install **kdump** when it is not enabled by default.

#### 42.1.1. What is kdump

**kdump** is a service which provides a crash dumping mechanism. The service enables you to save the contents of the system memory for 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 into a file. 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, operational **kdump** 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.

You can enable **kdump** for all installed kernels on a machine or only for specified kernels. This is useful when there are multiple kernels used on a machine, some of which are stable enough that there is no concern that they could crash.

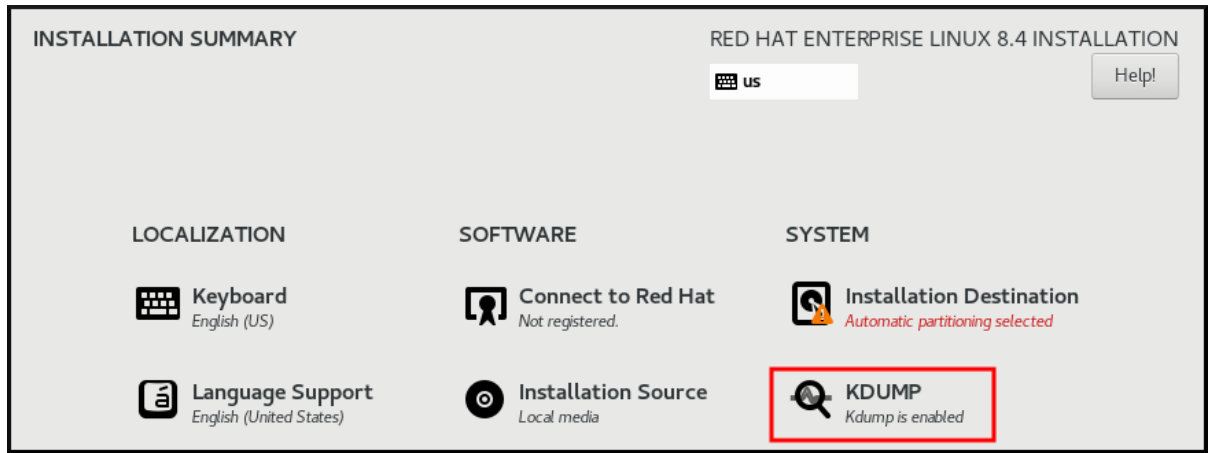
When **kdump** is installed, a default **/etc/kdump.conf** file is created. The file includes the default minimum **kdump** configuration. You can edit this file to customize the **kdump** configuration, but it is not required.

#### 42.1.2. Installing kdump using Anaconda

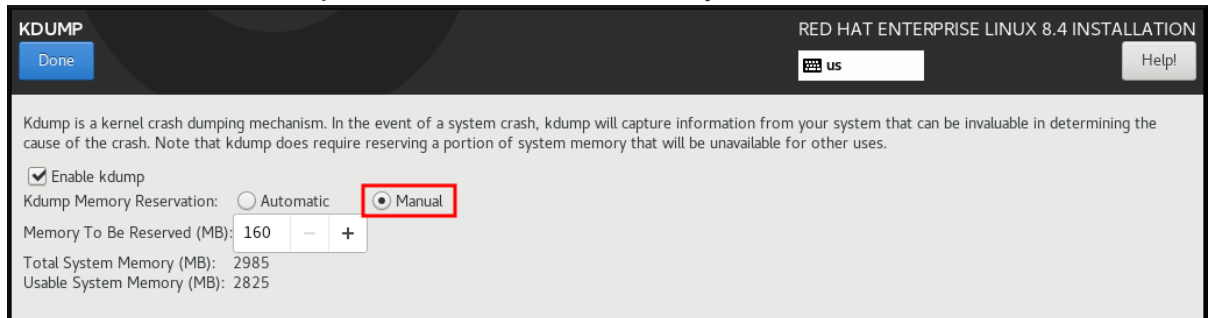
The **Anaconda** installer provides a graphical interface screen for **kdump** configuration during an interactive installation. The installer screen is titled as **KDUMP** and is available from the main **Installation Summary** screen. You can enable **kdump** and reserve the required amount of memory.

#### Procedure

1. Go to the **Kdump** field.
2. Enable **kdump** if not already enabled.



3. Define how much memory should be reserved for **kdump**.



### 42.1.3. Installing kdump on the command line

Some installation options, such as custom **Kickstart** installations, in some cases do **not** install or enable **kdump** by default. If this is your case, follow the procedure below.

#### Prerequisites

- An active RHEL subscription
- The **kexec-tools** package
- Fulfilled requirements for **kdump** configurations and targets. For details, see [Supported kdump configurations and targets](#).

#### Procedure

1. 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:

■

```
# dnf install kexec-tools
```



### IMPORTANT

Starting with kernel-3.10.0-693.el7 the **Intel IOMMU** driver is supported with **kdump**. For prior versions, kernel-3.10.0-514[.XYZ].el7 and earlier, it is advised that **Intel IOMMU** support is disabled, otherwise the capture kernel is likely to become unresponsive.

## 42.2. CONFIGURING KDUMP ON THE COMMAND LINE

Plan and build your **kdump** environment.

### 42.2.1. Estimating the kdump size

When planning and building your **kdump** environment, it is important to know how much space the crash dump file requires.

The **makedumpfile --mem-usage** command estimates how much space the crash dump file requires. It generates a memory usage report. The report helps you determine the dump level and which pages are safe to be excluded.

#### Procedure

- Execute the following command to generate a memory usage report:

```
# makedumpfile --mem-usage /proc/kcore
```

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 required memory in pages. This means that you must calculate the size of memory in use against the kernel page size.

### 42.2.2. Configuring kdump memory usage

The memory reservation for **kdump** occurs during the system boot. The memory size is set in the system's Grand Unified Bootloader (GRUB) configuration. The memory size depends on the value of the **crashkernel=** option specified in the configuration file and the size of the system physical memory.

You can define the **crashkernel=** option in many ways. You can specify the **crashkernel=** value or configure the **auto** option. The **crashkernel=auto** parameter reserves memory automatically, based on the total amount of physical memory in the system. When configured, the kernel automatically reserves an appropriate amount of required memory for the capture kernel. This helps to prevent Out-of-Memory (OOM) errors.



## NOTE

The automatic memory allocation for **kdump** varies based on system hardware architecture and available memory size.

If the system has less than the minimum memory threshold for automatic allocation, you can configure the amount of reserved memory manually.

## Prerequisites

- You have root permissions on the system.
- Fulfilled requirements for **kdump** configurations and targets. For details, see [Supported kdump configurations and targets](#).

## Procedure

1. Prepare the **crashkernel=** option.

- 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 command reserves 64 MB of memory if the total amount of system memory is in the range of 512 MB and 2 GB. If the total amount of memory is more than 2 GB, the memory reserve is 128 MB.

- Offset the reserved memory.  
Some systems require to reserve memory with a certain fixed offset because the **crashkernel** reservation happens early, and you may need to reserve more memory for special usage. When you define an offset, the reserved memory begins there. To offset the reserved memory, use the following syntax:

```
crashkernel=128M@16M
```

In this example, **kdump** reserves 128 MB of memory starting at 16 MB (physical address **0x01000000**). If you set the offset parameter to 0 or omit entirely, **kdump** offsets the reserved memory automatically. You can also use this syntax when setting a variable memory reservation. In that case, the offset is always specified last. For example:

```
crashkernel=512M-2G:64M,2G-:128M@16M
```

2. Apply the **crashkernel=** option to your boot loader configuration:

```
# grubby --update-kernel=ALL --args="crashkernel=<value>"
```

Replace **<value>** with the value of the **crashkernel=** option that you prepared in the previous step.



## Additional resources

- [Memory requirements for kdump](#)
- [Configuring kernel command-line parameters](#)
- [How to manually modify the boot parameter in grub before the system boots](#)
- [How to install and boot custom kernels in Red Hat Enterprise Linux 8](#)
- **grubby(8)** manual page

### 42.2.3. Configuring the kdump target

The crash dump is usually stored as a file in a local file system, written directly to a device. Alternatively, you can set up for the crash dump to be sent over a network using the **NFS** or **SSH** protocols. Only one of these options to preserve a crash dump file can be set at a time. The default behavior is to store it in the **/var/crash/** directory of the local file system.

#### Prerequisites

- **Root** permissions.
- Fulfilled requirements for **kdump** configurations and targets. For details, see [Supported kdump configurations and targets](#).

#### Procedure

- To store the crash dump 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 crash dump file.



#### NOTE

- When you specify a dump target in the **/etc/kdump.conf** file, then the path is **relative** to the specified dump target.
- When 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.

#### Example 42.1. The kdump target configuration

```
# grep -v ^# /etc/kdump.conf | grep -v ^$
ext4 /dev/mapper/vg00-varcrashvol
path /var/crash
core_collector makedumpfile -c --message-level 1 -d 31
```

Here, the dump target is specified (**ext4 /dev/mapper/vg00-varcrashvol**), and thus mounted at **/var/crash**. The **path** option is also set to **/var/crash**, so the **kdump** saves the **vmcore** file in the **/var/crash/var/crash** directory.

- To change the local directory in which the crash dump is to be saved, as **root**, edit the **/etc/kdump.conf** configuration file:
  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 RHEL 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 RHEL, where the directory was being created automatically if it did not exist when starting the service.

- To write the file to a different partition, edit the **/etc/kdump.conf** configuration file:
  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
```

### NOTE

The correct syntax for specifying UUID values is both **UUID="correct-uuid"** and **UUID=correct-uuid**.



### 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.

- To write the crash dump directly to a device, edit the **/etc/kdump.conf** configuration file:
  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 crash 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 crash 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
```

#### 42.2.4. Configuring the kdump core collector

The **kdump** service uses a **core\_collector** program to capture the crash dump image. In RHEL, the **makedumpfile** utility is the default core collector. It helps shrink the dump file by:

- Compressing the size of a crash dump file and copying only necessary pages using various dump levels
- Excluding unnecessary crash dump pages
- Filtering the page types to be included in the crash dump.

#### Syntax

```
core_collector makedumpfile -l --message-level 1 -d 31
```

#### Options

- **-c, -l** or **-p**: specify compress dump file format by each page using either, **zlib** for **-c** option, **lzo** for **-l** option or **snappy** for **-p** option.
- **-d (dump\_level)**: excludes pages so that they are not copied to the dump file.
- **--message-level**: specify the message types. You can restrict outputs printed by specifying **message\_level** with this option. For example, specifying 7 as **message\_level** prints common messages and error messages. The maximum value of **message\_level** is 31

## Prerequisites

- You have root permissions on the system.
- Fulfilled requirements for **kdump** configurations and targets. For details, see [Supported kdump configurations and targets](#).

## Procedure

1. 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`.
2. To enable crash dump file compression, execute:

```
core_collector makedumpfile -l --message-level 1 -d 31
```

The **-l** option specifies the **dump** compressed file format. The **-d** option specifies dump level as 31. The **--message-level** option specifies message level as 1.

Also, consider following examples with the **-c** and **-p** options:

- To compress a crash dump file using **-c**:

```
core_collector makedumpfile -c -d 31 --message-level 1
```

- To compress a crash dump file using **-p**:

```
core_collector makedumpfile -p -d 31 --message-level 1
```

## Additional resources

- [makedumpfile\(8\)](#) man page
- [The kdump configuration file](#)

### 42.2.5. Configuring the kdump default failure responses

By default, when **kdump** fails to create a crash dump file at the configured target location, the system reboots and the dump is lost in the process. To change this behavior, follow the procedure below.

## Prerequisites

- Root permissions.
- Fulfilled requirements for **kdump** configurations and targets. For details, see [Supported kdump configurations and targets](#).

## Procedure

1. As **root**, remove the hash sign ("**#**") from the beginning of the `#failure_action` line in the `/etc/kdump.conf` configuration file.
2. Replace the value with a desired action.

```
failure_action poweroff
```

### Additional resources

- [Configuring the kdump target](#)

## 42.2.6. Testing the kdump configuration

You can test that the crash 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 the **/etc/kdump.conf** file (by default to **/var/crash/**).



### NOTE

This action confirms the validity of the configuration. Also it is possible to use this action to record how long it takes for a crash dump to complete with a representative work-load.

### Additional resources

- [Configuring the kdump target](#)

## 42.3. ENABLING KDUMP

By using the procedure, you can enable or disable the **kdump** service for all installed kernels or for a specific kernel.

### 42.3.1. Enabling kdump for all installed kernels

You can enable and start the **kdump** service for all kernels installed on the machine.

#### Prerequisites

- Administrator privileges

#### Procedure

1. Add the **crashkernel=auto** command-line parameter to all installed kernels:

```
# grubby --update-kernel=ALL --args="crashkernel=auto"
```

2. Enable the **kdump** service.

```
# systemctl enable --now kdump.service
```

#### Verification

- Check that the **kdump** service is running:

```
# systemctl status kdump.service
○ kdump.service - Crash recovery kernel arming
  Loaded: loaded (/usr/lib/systemd/system/kdump.service; enabled; vendor preset:
disabled)
  Active: active (live)
```

### 42.3.2. Enabling kdump for a specific installed kernel

You can enable the **kdump** service for a specific kernel on the machine.

#### Prerequisites

- Administrator privileges

#### Procedure

1. List the kernels installed on the machine.

```
# ls -a /boot/vmlinuz-*
/boot/vmlinuz-0-rescue-2930657cd0dc43c2b75db480e5e5b4a9 /boot/vmlinuz-4.18.0-
330.el8.x86_64 /boot/vmlinuz-4.18.0-330.rt7.111.el8.x86_64
```

2. Add a specific **kdump** kernel to the system's Grand Unified Bootloader (GRUB) configuration file.

For example:

```
# grubby --update-kernel=vmlinuz-4.18.0-330.el8.x86_64 --args="crashkernel=auto"
```

3. Enable the **kdump** service.

```
# systemctl enable --now kdump.service
```

### Verification

- Check that the **kdump** service is running:

```
# systemctl status kdump.service
```

- kdump.service - Crash recovery kernel arming

```
Loaded: loaded (/usr/lib/systemd/system/kdump.service; enabled; vendor preset: disabled)
```

```
Active: active (live)
```

### 42.3.3. Disabling the kdump service

To disable the **kdump** service at boot time, follow the procedure below.

#### Prerequisites

- Fulfilled requirements for **kdump** configurations and targets. For details, see [Supported kdump configurations and targets](#).
- All configurations for installing **kdump** are set up according to your needs. For details, see [Installing kdump](#).

#### Procedure

1. To stop the **kdump** service in the current session:

```
# systemctl stop kdump.service
```

2. To disable the **kdump** service:

```
# systemctl disable kdump.service
```



#### WARNING

It is recommended to set **kptr\_restrict=1**. In that case, the **kdumpctl** service loads the crash kernel regardless of Kernel Address Space Layout (KASLR) being enabled or not.

## 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 `kdumppctl` service fails to access the `/proc/kcore` and load the crash kernel.

To work around this problem, the `/usr/share/doc/kexec-tools/kexec-kdump-howto.txt` file displays a warning message, which recommends the `kptr_restrict=1` setting.

To ensure that `kdumppctl` service loads the crash kernel, verify that `kernel.kptr_restrict = 1` is listed in the `sysctl.conf` file.

## Additional resources

- [Configuring basic system settings](#) in RHEL

## 42.4. CONFIGURING KDUMP IN THE WEB CONSOLE

Setup and test the `kdump` configuration in the RHEL 8 web console.

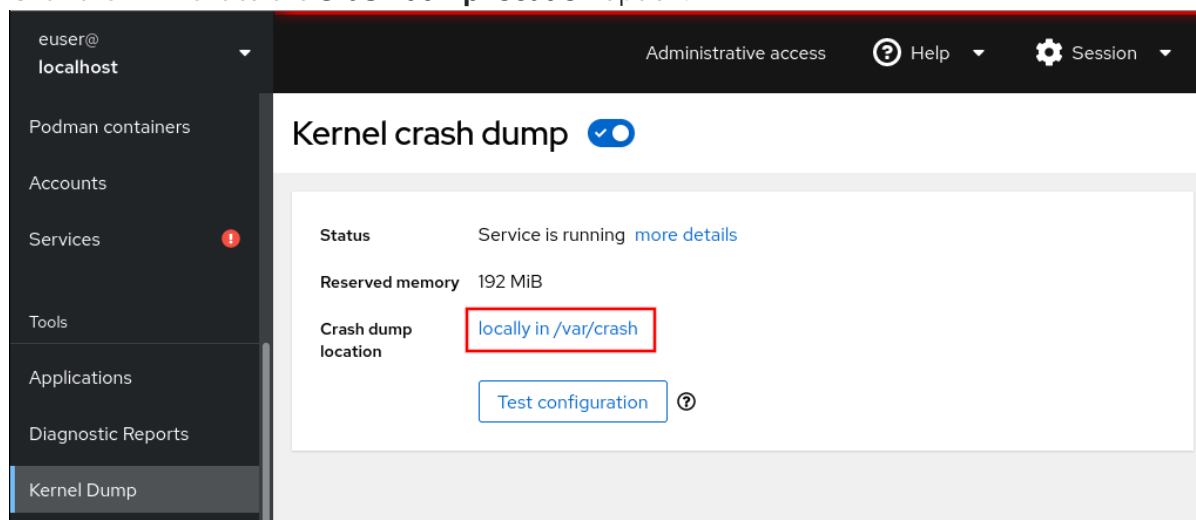
The web console is part of a default installation of RHEL 8 and enables or disables the `kdump` service at boot time. Further, the web console enables you to configure the reserved memory for `kdump`; or to select the `vmcore` saving location in an uncompressed or compressed format.

### 42.4.1. Configuring kdump memory usage and target location in web console

The procedure below shows you how to use the **Kernel Dump** tab in the RHEL 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.

#### Procedure

1. Open the **Kernel Dump** tab and start the `kdump` service.
2. Configure the `kdump` memory usage using the command line.
3. 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.



### Crash dump location

**Location**

**Directory**

**Compression**  Compress crash dumps to save space


[Apply](#) [Cancel](#)

- 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.

<b>Status</b>	Service is running <a href="#">more details</a>
<b>Reserved memory</b>	192 MiB
<b>Crash dump location</b>	<a href="#">locally in /var/crash</a>
	<a href="#">Test configuration</a> 

- a. Click **Test configuration**.
- b. In the **Test kdump settings** field, click **Crash system**.

**WARNING**

This step disrupts execution of the kernel and results in a system crash and loss of data.

**Additional resources**

- [Supported kdump targets](#)
- [Using secure communications between two systems with OpenSSH](#)

**42.4.2. Additional resources**

- [Getting started using the RHEL web console](#)

**42.5. SUPPORTED KDUMP CONFIGURATIONS AND TARGETS****42.5.1. 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 for **Minimum amount of reserved memory required forkdump**, includes the minimum memory requirements to automatically reserve a memory size for **kdump** on the latest available versions. The size changes according to the system's architecture and total available physical memory.

**Table 42.1. 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 4 GB	192 MB of RAM
	4 GB to 64 GB	256 MB of RAM
	64 GB and more	512 MB of RAM
64-bit ARM architecture ( <b>arm64</b> )	2 GB and more	480 MB of RAM
IBM Power Systems ( <b>ppc64le</b> )	2 GB to 4 GB	384 MB of RAM

Architecture	Available Memory	Minimum Reserved Memory
	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 (s390x)	1 GB to 4 GB	192 MB of RAM
	4 GB to 64 GB	256 MB of RAM
	64 GB 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

- [How has the crashkernel parameter changed between RHEL8 minor releases?](#)
- [Technology capabilities and limits tables](#)
- [Minimum threshold for automatic memory reservation](#)

#### 42.5.2. 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 threshold values for automatic memory allocation. If the system has memory less than the specified threshold value, you must configure the memory manually.

**Table 42.2. 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

### 42.5.3. 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**.

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.
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

- [Configuring the kdump target](#)

## 42.5.4. 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.

Option	Description
<b>1</b>	Zero pages
<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

- [Configuring the core collector](#)

## 42.5.5. 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.

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 <b>kdump</b> 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 <b>initramfs</b> , allowing the user to record the core dump manually.
<b>final_action</b>	Enable additional operations such as <b>reboot</b> , <b>halt</b> , and <b>poweroff</b> actions after a successful <b>kdump</b> or when <b>shell</b> or <b>dump_to_rootfs</b> failure action completes. The default <b>final_action</b> option is <b>reboot</b> .

#### Additional resources

- [Configuring the \*\*kdump\*\* default failure responses](#)

#### 42.5.6. Using **final\_action** parameter

The **final\_action** parameter enables you to use certain additional operations such as **reboot**, **halt**, and **poweroff** actions after a successful **kdump** or when an invoked **failure\_response** mechanism using **shell** or **dump\_to\_rootfs** completes. If the **final\_action** option is not specified, it defaults to **reboot**.

#### Procedure

1. Edit the `/etc/kdump.conf` file and add the **final\_action** parameter.

```
final_action <reboot | halt | poweroff>
```

2. Restart the **kdump** service:

```
kdumpctl restart
```

## 42.6. TESTING THE KDUMP CONFIGURATION

You can test that the crash 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 the **/etc/kdump.conf** file (by default to **/var/crash/**).

**NOTE**

This action confirms the validity of the configuration. Also it is possible to use this action to record how long it takes for a crash dump to complete with a representative work-load.

**Additional resources**

- [Configuring the kdump target](#)

**42.7. USING KEXEC TO BOOT INTO A DIFFERENT 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-commandline
```

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 your machine into a different kernel, the system does not go through the standard shutdown sequence before starting the next kernel. This can cause data loss or an unresponsive system.

## 42.8. PREVENTING KERNEL DRIVERS FROM LOADING FOR KDUMP

You can control the capture kernel from loading certain kernel drivers by adding the **KDUMP\_COMMANDLINE\_APPEND=** variable in the **/etc/sysconfig/kdump** configuration file. By using this method, you can prevent the **kdump** initial RAM disk image **initramfs** from loading the specified kernel module. This helps to prevent the out-of-memory (oom) killer errors or other crash kernel failures.

You can append the **KDUMP\_COMMANDLINE\_APPEND=** variable using one of the following configuration options:

- **rd.driver.blacklist=<modules>**
- **modprobe.blacklist=<modules>**

### Procedure

1. Select a kernel module that you intend to block from loading.

```
$ lsmod
```

Module	Size	Used by
fuse	126976	3
xt_CHECKSUM	16384	1



```

| ipt_MASQUERADE      16384 1
| uinput              20480 1
| xt_contrack        16384 1

```

The **lsmod** command displays a list of modules that are loaded to the currently running kernel.

2. Update the **KDUMP\_COMMANDLINE\_APPEND=** variable in the `/etc/sysconfig/kdump` file.

```

| #
| KDUMP_COMMANDLINE_APPEND="rd.driver.blacklist=hv_vmbus,hv_storvsc,hv_utils,
| hv_netvsc,hid-hyperv"

```

Also, consider the following example using the **modprobe.blacklist=<modules>** configuration option.

```

| # KDUMP_COMMANDLINE_APPEND="modprobe.blacklist=emcp
| modprobe.blacklist=bnx2fc modprobe.blacklist=libfcoe modprobe.blacklist=fcoe"

```

3. Restart the **kdump** service.

```

| # systemctl restart kdump

```

#### Additional resources

- **dracut.cmdline** man page

## 42.9. RUNNING KDUMP ON SYSTEMS WITH ENCRYPTED DISK

When you run a LUKS encrypted partition, systems require 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 an encrypted target location fails in the second kernel (capture kernel).

The **kdumpctl estimate** command helps you estimate the amount of memory you need for **kdump**. **kdumpctl estimate** prints the recommended **crashkernel** value, which is the most suitable memory size required for **kdump**.

The recommended **crashkernel** value is calculated based on the current kernel size, kernel module, initramfs, and the LUKS encrypted target memory requirement.

In case you are using the custom **crashkernel=** option, **kdumpctl estimate** prints the **LUKS required size** value. The value is the memory size required for LUKS encrypted target.

#### Procedure

1. Print the estimate **crashkernel=** value:

```

| # kdumpctl estimate
|
| Encrypted kdump target requires extra memory, assuming using the keyslot with minimum
| memory requirement
|   Reserved crashkernel: 256M
|   Recommended crashkernel: 652M

```

```

Kernel image size: 47M
Kernel modules size: 8M
Initramfs size: 20M
Runtime reservation: 64M
LUKS required size: 512M
Large modules: <none>
WARNING: Current crashkernel size is lower than recommended size 652M.

```

2. Configure the amount of required memory by increasing **crashkernel=** to the desired value.
3. Reboot the system.



#### NOTE

If the **kdump** service still fails to save the dump file to the encrypted target, increase the **crashkernel=** value as required.

## 42.10. FIRMWARE ASSISTED DUMP MECHANISMS

Firmware assisted dump (fadump) is a dump capturing mechanism, provided as an alternative to the **kdump** mechanism on IBM POWER systems. The **kexec** and **kdump** mechanisms are useful for capturing core dumps on AMD64 and Intel 64 systems. However, some hardware such as mini systems and mainframe computers, leverage the onboard firmware to isolate regions of memory and prevent any accidental overwriting of data that is important to the crash analysis. The **fadump** utility, is optimized for the **fadump** mechanisms and their integration with RHEL on IBM POWER systems.

### 42.10.1. Firmware assisted dump on IBM PowerPC hardware

The **fadump** utility captures the **vmcore** file from a fully-reset system with PCI and I/O devices. This mechanism uses firmware to preserve memory regions during a crash and then reuses the **kdump** userspace scripts to save the **vmcore** file. The memory regions consist of all system memory contents, except the boot memory, system registers, and hardware Page Table Entries (PTEs).

The **fadump** mechanism offers improved reliability over the traditional dump type, by rebooting the partition and using a new kernel to dump the data from the previous kernel crash. The **fadump** requires an IBM POWER6 processor-based or later version hardware platform.

For further details about the **fadump** mechanism, including PowerPC specific methods of resetting hardware, see the `/usr/share/doc/kexec-tools/fadump-howto.txt` file.



#### NOTE

The area of memory that is not preserved, known as boot memory, is the amount of RAM required to successfully boot the kernel after a crash event. By default, the boot memory size is 256MB or 5% of total system RAM, whichever is larger.

Unlike **kexec-initiated** event, the **fadump** mechanism uses the production kernel to recover a crash dump. When booting after a crash, PowerPC hardware makes the device node `/proc/device-tree/rtas/ibm.kernel-dump` available to the **proc** filesystem (**procf**s). The **fadump-aware kdump** scripts, check for the stored **vmcore**, and then complete the system reboot cleanly.

### 42.10.2. Enabling firmware assisted dump mechanism

You can enhance the crash dumping capabilities of IBM POWER systems by enabling the firmware assisted dump (**fadump**) mechanism.

In the Secure Boot environment, the **GRUB2** boot loader allocates a boot memory region, known as the Real Mode Area (RMA). The RMA has a size of 512 MB, which is divided among the boot components and, if a component exceeds its size allocation, **GRUB2** fails with an out-of-memory (**OOM**) error.



### WARNING

Do not enable firmware assisted dump (**fadump**) mechanism in the Secure Boot environment on RHEL 8.7 and 8.6 versions. The **GRUB2** boot loader fails with the following error:

```
error: ../../grub-core/kern/mm.c:376:out of memory.
Press any key to continue...
```

The system is recoverable only if you increase the default **initramfs** size due to the **fadump** configuration.

For information about workaround methods to recover the system, see the [System boot ends in GRUB Out of Memory \(OOM\)](#) article.

## Procedure

1. Install and configure **kdump**.
2. Enable the **fadump=on** kernel option:

```
# grubby --update-kernel=ALL --args="fadump=on"
```

3. (Optional) If you want to specify reserved boot memory instead of using the defaults, enable the **crashkernel=xxM** option, where **xx** is the amount of the memory required in megabytes:

```
# grubby --update-kernel=ALL --args="crashkernel=xxM fadump=on"
```



### IMPORTANT

When specifying boot configuration options, test all boot configuration options before you execute them. If the **kdump** kernel fails to boot, increase the value specified in **crashkernel=** argument gradually to set an appropriate value.

### 42.10.3. Firmware assisted dump mechanisms on IBM Z hardware

IBM Z systems support the following firmware assisted dump mechanisms:

- **Stand-alone dump (sadump)**
- **VMDUMP**

The **kdump** infrastructure is supported and utilized on IBM Z systems. However, using one of the firmware assisted dump (fadump) methods for IBM Z can provide various benefits:

- The **sadump** mechanism is initiated and controlled from the system console, and is stored on an **IPL** bootable device.
- The **VMDUMP** mechanism is similar to **sadump**. This tool is also initiated from the system console, but retrieves the resulting dump from hardware and copies it to the system for analysis.
- These methods (similarly to other hardware based dump mechanisms) have the ability to capture the state of a machine in the early boot phase, before the **kdump** service starts.
- Although **VMDUMP** contains a mechanism to receive the dump file into a Red Hat Enterprise Linux system, the configuration and control of **VMDUMP** is managed from the IBM Z Hardware console.

IBM discusses **sadump** in detail in the [Stand-alone dump program](#) article and **VMDUMP** in [Creating dumps on z/VM with VMDUMP](#) article.

IBM also has a documentation set for using the dump tools on Red Hat Enterprise Linux 7 in the [Using the Dump Tools on Red Hat Enterprise Linux 7.4](#) article.

#### Additional resources

- [Stand-alone dump program](#)
- [Creating dumps on z/VM with VMDUMP](#)
- [Using the Dump Tools on Red Hat Enterprise Linux 7.4](#)

#### 42.10.4. Using sadump on Fujitsu PRIMEQUEST systems

The Fujitsu **sadump** mechanism is designed to provide a **fallback** dump capture in an event when **kdump** is unable to complete successfully. The **sadump** mechanism is invoked manually from the system Management Board (MMB) interface. Using MMB, configure **kdump** like for an Intel 64 or AMD 64 server and then perform the following additional steps to enable **sadump**.

#### Procedure

1. Add or edit the following lines in the `/etc/sysctl.conf` file to ensure that **kdump** starts as expected for **sadump**:

```
kernel.panic=0
kernel.unknown_nmi_panic=1
```



#### WARNING

In particular, ensure that after **kdump**, the system does not reboot. If the system reboots after **kdump** has fails to save the **vmcore** file, then it is not possible to invoke the **sadump**.

2. Set the **failure\_action** parameter in **/etc/kdump.conf** appropriately as **halt** or **shell**.

```
failure_action shell
```

### Additional resources

- The FUJITSU Server PRIMEQUEST 2000 Series Installation Manual

## 42.11. 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 Kernel Oops Analyzer or the Kdump Helper tool.

### 42.11.1. Installing the crash utility

Install the **crash** tool to obtain the core analysis suite.

#### Procedure

1. Enable the relevant repositories:

```
# subscription-manager repos --enable baseos repository
```

```
# subscription-manager repos --enable appstream repository
```

```
# subscription-manager repos --enable rhel-8-for-x86_64-baseos-debug-rpms
```

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

- [Configuring basic system settings](#)

### 42.11.2. Running and exiting the crash utility

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/vmlinuz /var/crash/127.0.0.1-2018-10-06-14:05:33/vmcore
```

Use the same *<kernel>* version that was captured by **kdump**.

### Example 42.2. 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/vmlinuz
    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 42.3. 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.

**Additional resources**

- [A Guide to Unexpected System Restarts](#)

**42.11.3. Displaying various indicators in the crash utility**

Use the **crash** utility to 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> ffffffff 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 [ef4dbdccc] 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 [ef4dbee4] 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: ffffffff EBX: 00000001 ECX: b7776000 EDX: 00000002
    DS: 007b  ESI: 00000002 ES: 007b  EDI: b7776000
    SS: 007b  ESP: bfc2088 EBP: bfc20b4 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_VM
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.

### 42.11.4. Using Kernel Oops Analyzer

The Kernel Oops Analyzer tool 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.

## Procedure

- Access the Kernel Oops Analyzer tool.
- To diagnose a kernel crash issue, upload a kernel oops log generated in **vmcore**.
  - Alternatively you can also diagnose a kernel crash issue by providing a text message or a **vmcore-dmesg.txt** as an input.

The screenshot displays two side-by-side panels for the Kernel Oops Analyzer tool.

**Option 1: File Input**

- Contains a file selection button labeled "Choose File" and the text "No file chosen".
- Below the button, it says: "Choose and upload the [kernel oops log](#) generated from a vmcore. Maximum file size for uploaded kernel oops log is 10 MB."
- At the bottom is a blue button with a magnifying glass icon and the text "Detect".

**Option 2: Text Input**

- Contains a large text input area with a blue border and a cursor.
- At the bottom are two buttons: a blue "Detect" button with a magnifying glass icon, and a white "Clear" button with a blue border and a circular arrow icon.

- Click **DETECT** to compare the oops message based on information from the **makedumpfile** against known solutions.

## Additional resources

- [The Kernel Oops Analyzer](#) article
- [A Guide to Unexpected System Restarts](#)

### 42.11.5. The Kdump Helper tool

The Kdump Helper tool helps to set up the **kdump** using the provided information. Kdump Helper generates a configuration script based on your preferences. Initiating and running the script on your server sets up the **kdump** service.

## Additional resources

- [Kdump Helper](#)

## 42.12. USING EARLY KDUMP TO CAPTURE BOOT TIME CRASHES

You use the **early kdump** mechanism of the **kdump** service to capture the **vmcore** file during the early stages of the boot process. With the following information and the procedure, you can understand the **early kdump** mechanism, configuring, and checking the status of **early kdump**.

### 42.12.1. 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.

## 42.12.2. Enabling early kdump

The **early kdump** feature sets up the crash kernel and the initial RAM disk image ( **initramfs**) to load early enough to capture the **vmcore** information for an early crash. This helps to eliminate the risk of losing information about the early boot kernel crashes.

### Prerequisites

- An active RHEL subscription.
- A repository containing the **kexec-tools** package for your system CPU architecture
- Fulfilled **kdump** configuration and targets 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, set all required configurations and verify that **kdump** service is enabled.

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 the system to reflect the changes

```
# reboot
```

### Verification step

- 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

- The `/usr/share/doc/kexec-tools/early-kdump-howto.txt` file
- [What is early kdump support and how do I configure it?](#)

## 42.13. 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.
- [How to troubleshoot kernel crashes, hangs, or reboots with kdump on Red Hat Enterprise Linux](#)
- [What targets are supported for use with kdump?](#)

## CHAPTER 43. APPLYING PATCHES WITH KERNEL LIVE PATCHING

You can use the Red Hat Enterprise Linux kernel live patching solution to patch a running kernel without rebooting or restarting any processes.

With this solution, system administrators:

- Can immediately apply critical security patches to the kernel.
- Do not have to wait for long-running tasks to complete, for users to log off, or for scheduled downtime.
- Control the system's uptime more and do not sacrifice security or stability.

Note that not every critical or important CVE will be resolved using the kernel live patching solution. Our goal is to reduce the required reboots for security-related patches, not to eliminate them entirely. For more details about the scope of live patching, see the [Customer Portal Solutions article](#).



### WARNING

Some incompatibilities exist between kernel live patching and other kernel subcomponents. Read the

[Limitations of kpatch](#) carefully before using kernel live patching.

### 43.1. LIMITATIONS OF KPATCH

- The **kpatch** feature is not a general-purpose kernel upgrade mechanism. It is used for applying simple security and bug fix updates when rebooting the system is not immediately possible.
- Do not use the **SystemTap** or **kprobe** tools during or after loading a patch. The patch could fail to take effect until after such probes have been removed.

### 43.2. SUPPORT FOR THIRD-PARTY LIVE PATCHING

The **kpatch** utility is the only kernel live patching utility supported by Red Hat with the RPM modules provided by Red Hat repositories. Red Hat will not support any live patches which were not provided by Red Hat itself.

If you require support for an issue that arises with a third-party live patch, Red Hat recommends that you open a case with the live patching vendor at the outset of any investigation in which a root cause determination is necessary. This allows the source code to be supplied if the vendor allows, and for their support organization to provide assistance in root cause determination prior to escalating the investigation to Red Hat Support.

For any system running with third-party live patches, Red Hat reserves the right to ask for reproduction with Red Hat shipped and supported software. In the event that this is not possible, we require a similar system and workload be deployed on your test environment without live patches applied, to confirm if the same behavior is observed.

For more information about third-party software support policies, see [How does Red Hat Global Support Services handle third-party software, drivers, and/or uncertified hardware/hypervisors or guest operating systems?](#)

### 43.3. ACCESS TO KERNEL LIVE PATCHES

Kernel live patching capability is implemented as a kernel module (**kmod**) that is delivered as an RPM package.

All customers have access to kernel live patches, which are delivered through the usual channels. However, customers who do not subscribe to an extended support offering will lose access to new patches for the current minor release once the next minor release becomes available. For example, customers with standard subscriptions will only be able to live patch RHEL 8.2 kernel until the RHEL 8.3 kernel is released.

### 43.4. COMPONENTS OF KERNEL LIVE PATCHING

The components of kernel live patching are as follows:

#### Kernel patch module

- The delivery mechanism for kernel live patches.
- A kernel module which is built specifically for the kernel being patched.
- The patch module contains the code of the desired fixes for the kernel.
- The patch modules register with the **livepatch** kernel subsystem and provide information about original functions to be replaced, with corresponding pointers to the replacement functions. Kernel patch modules are delivered as RPMs.
- The naming convention is **kpatch\_<kernel version>\_<kpatch version>\_<kpatch release>**. The "kernel version" part of the name has *dots* replaced with *underscores*.

#### The **kpatch** utility

A command-line utility for managing patch modules.

#### The **kpatch** service

A **systemd** service required by **multiuser.target**. This target loads the kernel patch module at boot time.

#### The **kpatch-dnf** package

A DNF plugin delivered in the form of an RPM package. This plugin manages automatic subscription to kernel live patches.

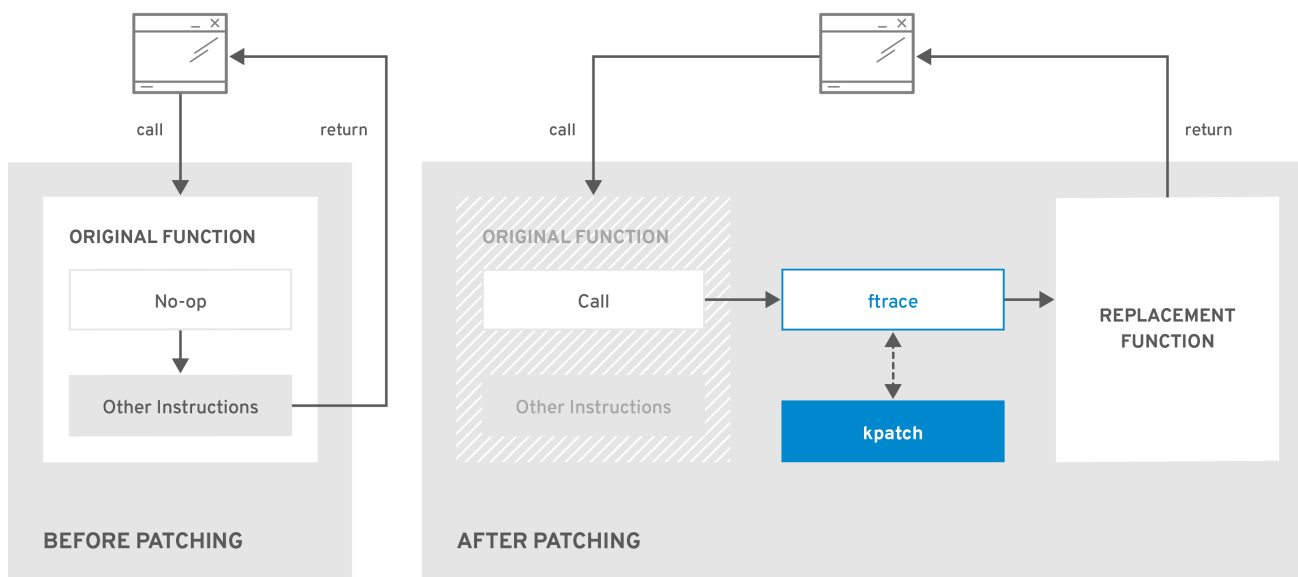
### 43.5. HOW KERNEL LIVE PATCHING WORKS

The **kpatch** kernel patching solution uses the **livepatch** kernel subsystem to redirect old functions to new ones. When a live kernel patch is applied to a system, the following things happen:

1. The kernel patch module is copied to the **/var/lib/kpatch/** directory and registered for re-application to the kernel by **systemd** on next boot.
2. The **kpatch** module is loaded into the running kernel and the new functions are registered to the **ftrace** mechanism with a pointer to the location in memory of the new code.

- When the kernel accesses the patched function, it is redirected by the **ftrace** mechanism which bypasses the original functions and redirects the kernel to patched version of the function.

Figure 43.1. How kernel live patching works



RHEL\_424549\_0119

## 43.6. SUBSCRIBING THE CURRENTLY INSTALLED KERNELS TO THE LIVE PATCHING STREAM

A kernel patch module is delivered in an RPM package, specific to the version of the kernel being patched. Each RPM package will be cumulatively updated over time.

The following procedure explains how to subscribe to all future cumulative live patching updates for a given kernel. Because live patches are cumulative, you cannot select which individual patches are deployed for a given kernel.



### WARNING

Red Hat does not support any third party live patches applied to a Red Hat supported system.

### Prerequisites

- Root permissions

### Procedure

1. Optionally, check your kernel version:

```
# uname -r
4.18.0-94.el8.x86_64
```

2. Search for a live patching package that corresponds to the version of your kernel:

```
# yum search $(uname -r)
```

3. Install the live patching package:

```
# yum install "kpatch-patch = $(uname -r)"
```

The command above installs and applies the latest cumulative live patches for that specific kernel only.

If the version of a live patching package is 1-1 or higher, the package will contain a patch module. In that case the kernel will be automatically patched during the installation of the live patching package.

The kernel patch module is also installed into the `/var/lib/kpatch/` directory to be loaded by the **systemd** system and service manager during the future reboots.



#### NOTE

An empty live patching package will be installed when there are no live patches available for a given kernel. An empty live patching package will have a `kpatch_version-kpatch_release` of 0-0, for example **kpatch-patch-4\_18\_0-94-0-0.el8.x86\_64.rpm**. The installation of the empty RPM subscribes the system to all future live patches for the given kernel.

4. Optionally, verify that the kernel is patched:

```
# kpatch list
Loaded patch modules:
kpatch_4_18_0_94_1_1 [enabled]

Installed patch modules:
kpatch_4_18_0_94_1_1 (4.18.0-94.el8.x86_64)
...
```

The output shows that the kernel patch module has been loaded into the kernel, which is now patched with the latest fixes from the **kpatch-patch-4\_18\_0-94-1-1.el8.x86\_64.rpm** package.

#### Additional resources

- **kpatch(1)** manual page
- [Configuring basic system settings](#) in RHEL

## 43.7. AUTOMATICALLY SUBSCRIBING ANY FUTURE KERNEL TO THE LIVE PATCHING STREAM

You can use the **kpatch-dnf** YUM plugin to subscribe your system to fixes delivered by the kernel patch module, also known as kernel live patches. The plugin enables **automatic** subscription for any kernel the system currently uses, and also for kernels **to-be-installed in the future**

#### Prerequisites



- You have root permissions.

## Procedure

1. Optionally, check all installed kernels and the kernel you are currently running:

```
# yum list installed | grep kernel
Updating Subscription Management repositories.
Installed Packages
...
kernel-core.x86_64      4.18.0-240.10.1.el8_3      @rhel-8-for-x86_64-baseos-rpms
kernel-core.x86_64      4.18.0-240.15.1.el8_3      @rhel-8-for-x86_64-baseos-rpms
...

# uname -r
4.18.0-240.10.1.el8_3.x86_64
```

2. Install the **kpatch-dnf** plugin:

```
# yum install kpatch-dnf
```

3. Enable automatic subscription to kernel live patches:

```
# yum kpatch auto
Updating Subscription Management repositories.
Last metadata expiration check: 19:10:26 ago on Wed 10 Mar 2021 04:08:06 PM CET.
Dependencies resolved.
=====
Package                Architecture
=====
Installing:
kpatch-patch-4_18_0-240_10_1  x86_64
kpatch-patch-4_18_0-240_15_1  x86_64

Transaction Summary
=====
Install 2 Packages
...

```

This command subscribes all currently installed kernels to receiving kernel live patches. The command also installs and applies the latest cumulative live patches, if any, for all installed kernels.

In the future, when you update the kernel, live patches will automatically be installed during the new kernel installation process.

The kernel patch module is also installed into the **/var/lib/kpatch/** directory to be loaded by the **systemd** system and service manager during future reboots.

**NOTE**

An empty live patching package will be installed when there are no live patches available for a given kernel. An empty live patching package will have a *kpatch\_version-kpatch\_release* of 0-0, for example **kpatch-patch-4\_18\_0-240-0-0.el8.x86\_64.rpm**. The installation of the empty RPM subscribes the system to all future live patches for the given kernel.

**Verification step**

- Verify that all installed kernels have been patched:

```
# kpatch list
Loaded patch modules:
kpatch_4_18_0_240_10_1_0_1 [enabled]

Installed patch modules:
kpatch_4_18_0_240_10_1_0_1 (4.18.0-240.10.1.el8_3.x86_64)
kpatch_4_18_0_240_15_1_0_2 (4.18.0-240.15.1.el8_3.x86_64)
```

The output shows that both the kernel you are running, and the other installed kernel have been patched with fixes from **kpatch-patch-4\_18\_0-240\_10\_1-0-1.rpm** and **kpatch-patch-4\_18\_0-240\_15\_1-0-1.rpm** packages respectively.

**Additional resources**

- **kpatch(1)** and **dnf-kpatch(8)** manual pages
- [Configuring basic system settings](#) in RHEL

## 43.8. DISABLING AUTOMATIC SUBSCRIPTION TO THE LIVE PATCHING STREAM

When you subscribe your system to fixes delivered by the kernel patch module, your subscription is **automatic**. You can disable this feature, and thus disable automatic installation of **kpatch-patch** packages.

**Prerequisites**

- You have root permissions.

**Procedure**

1. Optionally, check all installed kernels and the kernel you are currently running:

```
# yum list installed | grep kernel
Updating Subscription Management repositories.
Installed Packages
...
kernel-core.x86_64      4.18.0-240.10.1.el8_3      @rhel-8-for-x86_64-baseos-rpms
kernel-core.x86_64      4.18.0-240.15.1.el8_3      @rhel-8-for-x86_64-baseos-rpms
...
```

```
# uname -r
4.18.0-240.10.1.el8_3.x86_64
```

2. Disable automatic subscription to kernel live patches:

```
# yum kpatch manual
Updating Subscription Management repositories.
```

### Verification step

- You can check for the successful outcome:

```
# yum kpatch status
...
Updating Subscription Management repositories.
Last metadata expiration check: 0:30:41 ago on Tue Jun 14 15:59:26 2022.
Kpatch update setting: manual
```

### Additional resources

- **kpatch(1)** and **dnf-kpatch(8)** manual pages

## 43.9. UPDATING KERNEL PATCH MODULES

Since kernel patch modules are delivered and applied through RPM packages, updating a cumulative kernel patch module is like updating any other RPM package.

### Prerequisites

- The system is subscribed to the live patching stream, as described in [Subscribing the currently installed kernels to the live patching stream](#).

### Procedure

- Update to a new cumulative version for the current kernel:

```
# yum update "kpatch-patch = $(uname -r)"
```

The command above automatically installs and applies any updates that are available for the currently running kernel. Including any future released cumulative live patches.

- Alternatively, update all installed kernel patch modules:

```
# yum update "kpatch-patch"
```



### NOTE

When the system reboots into the same kernel, the kernel is automatically live patched again by the **kpatch.service** systemd service.

### Additional resources

- [Configuring basic system settings](#) in RHEL

## 43.10. REMOVING THE LIVE PATCHING PACKAGE

Disable the Red Hat Enterprise Linux kernel live patching solution by removing the live patching package.

### Prerequisites

- Root permissions
- The live patching package is installed.

### Procedure

1. Select the live patching package.

```
# yum list installed | grep kpatch-patch
kpatch-patch-4_18_0-94.x86_64    1-1.el8    @@commandline
...
```

The example output above lists live patching packages that you installed.

2. Remove the live patching package.

```
# yum remove kpatch-patch-4_18_0-94.x86_64
```

When a live patching package is removed, the kernel remains patched until the next reboot, but the kernel patch module is removed from disk. On future reboot, the corresponding kernel will no longer be patched.

3. Reboot your system.
4. Verify that the live patching package has been removed.

```
# yum list installed | grep kpatch-patch
```

The command displays no output if the package has been successfully removed.

5. Optionally, verify that the kernel live patching solution is disabled.

```
# kpatch list
Loaded patch modules:
```

The example output shows that the kernel is not patched and the live patching solution is not active because there are no patch modules that are currently loaded.



### IMPORTANT

Currently, Red Hat does not support reverting live patches without rebooting your system. In case of any issues, contact our support team.

### Additional resources

- The **kpatch(1)** manual page
- [Configuring basic system settings](#) in RHEL

## 43.11. UNINSTALLING THE KERNEL PATCH MODULE

Prevent the Red Hat Enterprise Linux kernel live patching solution from applying a kernel patch module on subsequent boots.

### Prerequisites

- Root permissions
- A live patching package is installed.
- A kernel patch module is installed and loaded.

### Procedure

1. Select a kernel patch module:

```
# kpatch list
Loaded patch modules:
kpatch_4_18_0_94_1_1 [enabled]

Installed patch modules:
kpatch_4_18_0_94_1_1 (4.18.0-94.el8.x86_64)
...
```

2. Uninstall the selected kernel patch module.

```
# kpatch uninstall kpatch_4_18_0_94_1_1
uninstalling kpatch_4_18_0_94_1_1 (4.18.0-94.el8.x86_64)
```

- Note that the uninstalled kernel patch module is still loaded:

```
# kpatch list
Loaded patch modules:
kpatch_4_18_0_94_1_1 [enabled]

Installed patch modules:
<NO_RESULT>
```

When the selected module is uninstalled, the kernel remains patched until the next reboot, but the kernel patch module is removed from disk.

3. Reboot your system.
4. Optionally, verify that the kernel patch module has been uninstalled.

```
# kpatch list
Loaded patch modules:
...
```

The example output above shows no loaded or installed kernel patch modules, therefore the kernel is not patched and the kernel live patching solution is not active.



## IMPORTANT

Currently, Red Hat does not support reverting live patches without rebooting your system. In case of any issues, contact our support team.

### Additional resources

- The **kpatch(1)** manual page

## 43.12. DISABLING KPATCH.SERVICE

Prevent the Red Hat Enterprise Linux kernel live patching solution from applying all kernel patch modules globally on subsequent boots.

### Prerequisites

- Root permissions
- A live patching package is installed.
- A kernel patch module is installed and loaded.

### Procedure

1. Verify **kpatch.service** is enabled.

```
# systemctl is-enabled kpatch.service
enabled
```

2. Disable **kpatch.service**.

```
# systemctl disable kpatch.service
Removed /etc/systemd/system/multi-user.target.wants/kpatch.service.
```

- Note that the applied kernel patch module is still loaded:

```
# kpatch list
Loaded patch modules:
kpatch_4_18_0_94_1_1 [enabled]

Installed patch modules:
kpatch_4_18_0_94_1_1 (4.18.0-94.el8.x86_64)
```

3. Reboot your system.
4. Optionally, verify the status of **kpatch.service**.

```
# systemctl status kpatch.service
● kpatch.service - "Apply kpatch kernel patches"
  Loaded: loaded (/usr/lib/systemd/system/kpatch.service; disabled; vendor preset: disabled)
  Active: inactive (dead)
```

- The example output testifies that **kpatch.service** has been disabled and is not running. Thereby, the kernel live patching solution is not active.
5. Verify that the kernel patch module has been unloaded.

```
# kpatch list
Loaded patch modules:
<NO_RESULT>

Installed patch modules:
kpatch_4_18_0_94_1_1 (4.18.0-94.el8.x86_64)
```

The example output above shows that a kernel patch module is still installed but the kernel is not patched.



### IMPORTANT

Currently, Red Hat does not support reverting live patches without rebooting your system. In case of any issues, contact our support team.

#### Additional resources

- The **kpatch(1)** manual page
- [Configuring basic system settings](#) in RHEL

## CHAPTER 44. SETTING LIMITS FOR APPLICATIONS

You can use the *control groups* (**cgroups**) kernel functionality to set limits, prioritize or isolate the hardware resources of processes. This allows you to granularly control resource usage of applications to utilize them more efficiently.

### 44.1. 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. The **systemd** system and service manager utilizes **cgroups** to organize all units and services that it governs. Alternatively, you can manage **cgroups** hierarchies manually by creating and removing sub-directories in the `/sys/fs/cgroup/` directory.

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.

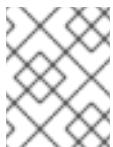
The **cgroups-v1** controllers were developed across a large time span and as a result, the behavior and naming of their control files is not uniform.

#### 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.



#### NOTE

**cgroups-v2** is fully supported in RHEL 8.2 and later versions. For more information, see [Control Group v2 is now fully supported in RHEL 8](#).

This sub-section was based on a Devconf.cz 2019 presentation.<sup>[4]</sup>

#### Additional resources

- [What are kernel resource controllers](#)
- **cgroups(7)** manual page



- [Role of systemd in control groups](#)

## 44.2. WHAT ARE KERNEL RESOURCE CONTROLLERS

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

- **cgroups(7)** manual page
- Documentation in **/usr/share/doc/kernel-doc-<kernel\_version>/Documentation/cgroups-v1/** directory (after installing the **kernel-doc** package).

## 44.3. WHAT ARE NAMESPACES

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

Namespace	Isolates
User	User and group IDs
Control groups	Control group root directory

### Additional resources

- [namespaces\(7\)](#) and [cgroup\\_namespaces\(7\)](#) manual pages
- [Understanding control groups](#)

## 44.4. 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

- You have root permissions.
- You have an application whose CPU consumption you want to restrict.
- You verified 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  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`).

- 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
```

- 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.

- 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.

- 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 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%.



### IMPORTANT

The **cgroups-v2** counterpart for **cpu.cfs\_period\_us** and **cpu.cfs\_quota\_us** is the **cpu.max** file. The **cpu.max** file is available through the **cpu** controller.

#### Additional resources

- [Understanding control groups](#)
- [What kernel resource controllers are](#)
- **cgroups(7)**, **sysfs(5)** manual pages

---

[4] Linux Control Group v2 - An Introduction, Devconf.cz 2019 presentation by Waiman Long

## CHAPTER 45. ANALYZING SYSTEM PERFORMANCE WITH BPF COMPILER COLLECTION

As a system administrator, you can 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.

### 45.1. INSTALLING THE BCC-TOOLS PACKAGE

Install the **bcc-tools** package, which also installs the BPF Compiler Collection (BCC) library as a dependency.

#### Procedure

1. Install **bcc-tools**.

```
# yum install bcc-tools
```

The BCC tools are installed in the **/usr/share/bcc/tools/** directory.

2. 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.

### 45.2. USING SELECTED BCC-TOOLS FOR PERFORMANCE ANALYSES

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

- [Installed bcc-tools package](#)
- Root permissions

#### Using **execsnoop** to examine the system processes

1. Run the **execsnoop** program in one terminal:

```
# /usr/share/bcc/tools/execsnoop
```

2. In another terminal run, 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/
...
```

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 **execsnoop** output displays the following fields:

- **PCOMM** - The parent process name. (**ls**)
- **PID** - The process ID. (**8382**)
- **PPID** - The parent process ID. (**8287**)
- **RET** - The return value of the **exec()** system call (**0**), which loads program code into new processes.
- **ARGS** - The location of the started program with arguments.

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. Run 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, enter:

```
$ 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 **opensnoop** output displays the following fields:

- **PID** - The process ID. (**8596**)
- **COMM** - The process name. (**uname**)
- **FD** - The file descriptor - a value that **open()** returns to refer to the open file. (**3**)
- **ERR** - Any errors.
- **PATH** - The location of files that **open()** tried to open.  
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. Run 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 enter, 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 **biotop** output displays the following fields:

- **PID** - The process ID. (**9568**)
- **COMM** - The process name. (**dd**)
- **DISK** - The disk performing the read operations. (**vda**)
- **I/O** - The number of read operations performed. (16294)
- **Kbytes** - The amount of Kbytes reached by the read operations. (14,440,636)
- **AVGms** - The average I/O time of read operations. (3.69)

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. Run 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 enter, 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'getsriptPlugin.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 **xfsslower** output displays the following fields:

- **COMM** - The process name. (**b'bash'**)
- **T** - The operation type. (**R**)
  - **Read**
  - **Write**
  - **Sync**
- **OFF\_KB** - The file offset in KB. (0)
- **FILENAME** - The file being read, written, or synced.

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 VII. DESIGN OF HIGH AVAILABILITY SYSTEM

## CHAPTER 46. 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**.

### 46.1. HIGH AVAILABILITY ADD-ON COMPONENTS

The Red Hat High Availability Add-On consists of several components that provide the high availability service.

The major components of the High Availability Add-On are as follows:

- 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.
- HAProxy – Routing software that provides high availability load balancing and failover in layer 4 (TCP) and layer 7 (HTTP, HTTPS) services.

### 46.2. HIGH AVAILABILITY ADD-ON CONCEPTS

Some of the key concepts of a Red Hat High Availability Add-On cluster are as follows.

#### 46.2.1. Fencing

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](#) .

### 46.2.2. Quorum

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.

### 46.2.3. Cluster resources

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*.

## 46.3. 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.

### 46.3.1. 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.

#### 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.

### 46.3.2. Pacemaker 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.

### 46.3.3. 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.

## 46.4. 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.

The cluster configurations you can choose are as follows:

- 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.

### 46.4.1. 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 can 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.

#### 46.4.2. Configuring LVM volumes in a cluster

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.



#### NOTE

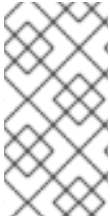
If an LVM volume group used by a Pacemaker cluster contains one or more physical volumes that reside on remote block storage, such as an iSCSI target, Red Hat recommends that you configure a **systemd resource-agents-deps** target and a **systemd** drop-in unit for the target to ensure that the service starts before Pacemaker starts. For information on configuring a **systemd resource-agents-deps** target, see [Configuring startup order for resource dependencies not managed by Pacemaker](#) .

- 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 procedures for configuring shared LVM volumes that use the **lvmlockd** daemon to manage storage devices in active/active configurations, see [GFS2 file systems in a cluster](#) and [Configuring an active/active Samba server in a Red Hat High Availability cluster](#) .

## CHAPTER 47. GETTING STARTED WITH PACEMAKER

To familiarize yourself with the tools and processes you use to create a Pacemaker cluster, you can run the following procedures. 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. For full information on Red Hat's support policies, requirements, and limitations for RHEL High Availability clusters, see [Support Policies for RHEL High Availability Clusters](#).

### 47.1. LEARNING TO USE PACEMAKER

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.

In this example:

- The node is **z1.example.com**.
- The floating IP address is 192.168.122.120.

#### Prerequisites

- A single node running RHEL 8
- A floating IP address that resides on the same network as one of the node's statically assigned IP addresses
- The name of the node on which you are running is in your **/etc/hosts** file

#### Procedure

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**.

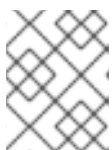


#### WARNING

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=1 min (WebSite-monitor-interval-1 min)
```

- Point your browser to the website you created using the floating IP address you configured. This should display the text message you defined.
- 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
```

- 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
```

## 47.2. LEARNING TO CONFIGURE FAILOVER

The following 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 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.

In this example:

- The nodes are **z1.example.com** and **z2.example.com**.
- The floating IP address is 192.168.122.120.

### Prerequisites

- Two nodes running RHEL 8 that can communicate with each other
- A floating IP address that resides on the same network as one of the node's statically assigned IP addresses
- The name of the node on which you are running is in your **/etc/hosts** file

### Procedure

- 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
```

- On both nodes in the cluster, set a password for user **hacluster** .

```
# passwd hacluster
```

- 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
```

- 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
```

- 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
```

- 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
```

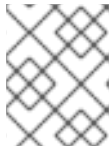
```
PCSD Status:
```

```
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 httpd** 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 node unstandby z1.example.com
```

11. For final cleanup, stop the cluster services on both nodes.

```
# pcs cluster stop --all
```

## CHAPTER 48. 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.

### 48.1. PCS HELP DISPLAY

You use the **-h** option of **pcs** to display the parameters of a **pcs** command and a description of those parameters.

The following command displays the parameters of the **pcs resource** command.

```
# pcs resource -h
```

### 48.2. 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
```

### 48.3. 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.

#### Procedure

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
```

- Copy the saved file to the working file you will be using for the configuration updates.

```
# cp original.xml updated.xml
```

- 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
```

- 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
```

## 48.4. DISPLAYING CLUSTER STATUS

There are a variety of commands you can use to display the status of a cluster and its components.

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
```

## 48.5. DISPLAYING THE FULL CLUSTER CONFIGURATION

Use the following command to display the full current cluster configuration.

```
# pcs config
```

## 48.6. MODIFYING THE COROSYNC.CONF FILE WITH THE PCS COMMAND

As of Red Hat Enterprise Linux 8.4, you can use the **pcs** command to modify the parameters in the **corosync.conf** file.

The following command modifies the parameters in the **corosync.conf** file.

```
pcs cluster config update [transport pass:quotes[transport options]] [compression  
pass:quotes[compression options]] [crypto pass:quotes[crypto options]] [totem pass:quotes[totem  
options]] [--corosync_conf pass:quotes[path]]
```

The following example command updates the **knet\_pmtud\_interval** transport value and the **token** and **join** totem values.

```
# pcs cluster config update transport knet_pmtud_interval=35 totem token=10000 join=100
```

### Additional resources

- For information on adding and removing nodes from an existing cluster, see [Managing cluster nodes](#).
- For information on adding and modifying links in an existing cluster, see [Adding and modifying links in an existing cluster](#).
- For information on modifying quorum options and managing the quorum device settings in a cluster, see [Configuring cluster quorum](#). and [Configuring quorum devices](#).

## 48.7. DISPLAYING THE COROSYNC.CONF FILE WITH THE PCS COMMAND

The following command displays the contents of the **corosync.conf** cluster configuration file.

```
# pcs cluster corosync
```

As of Red Hat Enterprise Linux 8.4, you can print the contents of the **corosync.conf** file in a human-readable format with the **pcs cluster config** command, as in the following example.

The output for this command includes the UUID for the cluster if the cluster was created in RHEL 8.7 or later or if the UUID was added manually as described in [Identifying clusters by UUID](#).

```
[root@r8-node-01 ~]# pcs cluster config  
Cluster Name: HACluster  
Cluster UUID: ad4ae07dcafe4066b01f1cc9391f54f5  
Transport: knet  
Nodes:  
r8-node-01:  
Link 0 address: r8-node-01
```

```

Link 1 address: 192.168.122.121
nodeid: 1
r8-node-02:
Link 0 address: r8-node-02
Link 1 address: 192.168.122.122
nodeid: 2
Links:
Link 1:
linknumber: 1
ping_interval: 1000
ping_timeout: 2000
pong_count: 5
Compression Options:
level: 9
model: zlib
threshold: 150
Crypto Options:
cipher: aes256
hash: sha256
Totem Options:
downcheck: 2000
join: 50
token: 10000
Quorum Device: net
Options:
sync_timeout: 2000
timeout: 3000
Model Options:
algorithm: lms
host: r8-node-03
Heuristics:
exec_ping: ping -c 1 127.0.0.1

```

As of RHEL 8.4, you can run the **pcs cluster config show** command with the **--output-format=cmd** option to display the **pcs** configuration commands that can be used to recreate the existing **corosync.conf** file, as in the following example.

```

[root@r8-node-01 ~]# pcs cluster config show --output-format=cmd
pcs cluster setup HACluster \
r8-node-01 addr=r8-node-01 addr=192.168.122.121 \
r8-node-02 addr=r8-node-02 addr=192.168.122.122 \
transport \
knet \
link \
linknumber=1 \
ping_interval=1000 \
ping_timeout=2000 \
pong_count=5 \
compression \
level=9 \
model=zlib \
threshold=150 \
crypto \
cipher=aes256 \
hash=sha256 \
totem \

```

downcheck=2000 \  
join=50 \  
token=10000



## CHAPTER 49. CREATING A RED HAT HIGH-AVAILABILITY CLUSTER WITH PACEMAKER

Create a Red Hat High Availability two-node cluster using the **pcs** command line interface with the following procedure.

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**.

### 49.1. INSTALLING CLUSTER SOFTWARE

Install the cluster software and configure your system for cluster creation with the following procedure.

#### Procedure

1. On each node in the cluster, enable the repository for high availability that corresponds to your system architecture. For example, to enable the high availability repository for an x86\_64 system, you can enter the following **subscription-manager** command:

```
# subscription-manager repos --enable=rhel-8-for-x86_64-highavailability-rpms
```

2. 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**

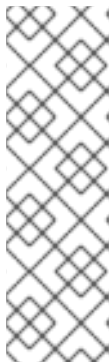
After you install the Red Hat High Availability Add-On packages, you should ensure that your software update preferences are set so that nothing is installed automatically. Installation on a running cluster can cause unexpected behaviors. For more information, see [Recommended Practices for Applying Software Updates to a RHEL High Availability or Resilient Storage Cluster](#).

3. 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.

4. 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.
```

5. 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
```

## 49.2. 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.

### Procedure

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.

## 49.3. CREATING A HIGH AVAILABILITY CLUSTER

Create a Red Hat High Availability Add-On cluster with the following procedure. This example procedure creates a cluster that consists of the nodes **z1.example.com** and **z2.example.com**.

### Procedure

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
...
```

## 49.4. 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 basic command to create a two-node cluster with two links is as follows.

```
pcs cluster setup pass:quotes[cluster_name] pass:quotes[node1_name]
addr=pass:quotes[node1_link0_address] addr=pass:quotes[node1_link1_address]
pass:quotes[node2_name] addr=pass:quotes[node2_link0_address]
addr=pass:quotes[node2_link1_address]
```

For the full syntax of this command, see the **pcs(8)** man page.

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.

- By default, if **link\_priority** is not specified for a link, the link's priority is equal to the link number. The link priorities are then 0, 1, 2, 3, and so forth, according to the order specified, with 0 being the highest link priority.
- The default link mode is **passive**, meaning the active link with the lowest-numbered link priority is used.
- With the default values of **link\_mode** and **link\_priority**, the first link specified will be used as the highest priority link, and if that link fails the next link specified will be used.
- 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
```

To set a link priority to a different value than the default value, which is the link number, you can set the link priority with the **link\_priority** option of the **pcs cluster setup** command. Each of the following two example commands creates a two-node cluster with two interfaces where the first link, link 0, has a link priority of 1 and the second link, link 1, has a link priority of 0. Link 1 will be used first and link 0 will serve as the failover link. Since link mode is not specified, it defaults to passive.

These two commands are equivalent. If you do not specify a link number following the **link** keyword, the **pcs** interface automatically adds a link number, starting with the lowest unused link number.

```
# 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 transport knet
link link_priority=1 link link_priority=0
```

```
# 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 transport knet
link linknumber=1 link_priority=0 link link_priority=1
```

You can set the link mode to a different value than the default value of **passive** with the **link\_mode** option of the **pcs cluster setup** command, as in the following example.

```
# 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 transport knet
link_mode=active
```

The following example sets both the link mode and the link priority.

```
# 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 transport knet
link_mode=active link link_priority=1 link link_priority=0
```

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](#).

## 49.5. 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.

### Procedure

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.

## 49.6. BACKING UP AND RESTORING A CLUSTER CONFIGURATION

The following commands back up a cluster configuration in a tar archive and restore the cluster configuration files on all nodes from the backup.

### Procedure

Use the following command to back up the cluster configuration in a tar archive. 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 cluster nodes from the backup. Specifying the **--local** option restores the cluster configuration files only on the node from which you run this command. If you do not specify a file name, the standard input will be used.

```
pcs config restore [--local] [filename]
```

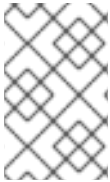
## 49.7. ENABLING PORTS FOR THE HIGH AVAILABILITY ADD-ON

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.

The following table 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 49.1. 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 pcsd 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>



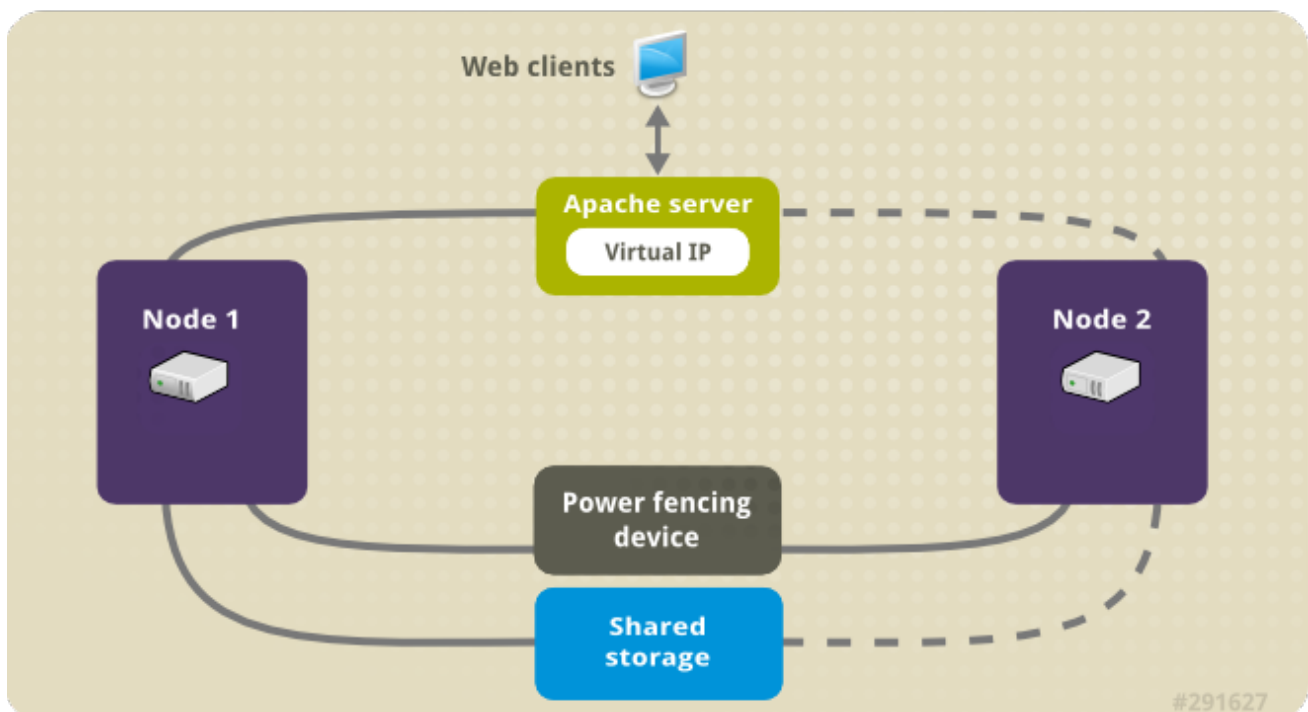
Port	When Required
TCP 5403	Required on the quorum device host when using a quorum device with <b>corosync-qnetd</b> . The default value can be changed with the <b>-p</b> option of the <b>corosync-qnetd</b> command.
UDP 5404-5412	Required on corosync nodes to facilitate communication between nodes. It 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.
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 50. CONFIGURING AN ACTIVE/PASSIVE APACHE HTTP SERVER IN A RED HAT HIGH AVAILABILITY CLUSTER

Configure an active/passive Apache HTTP server in a two-node Red Hat Enterprise Linux High Availability Add-On cluster with the following procedure. 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.

The following illustration 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 50.1. 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, Fibre Channel, or other shared network block device.

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 XFS 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.

## 50.1. CONFIGURING AN LVM VOLUME WITH AN XFS FILE SYSTEM IN A PACEMAKER CLUSTER

Create an LVM logical volume on storage that is shared between the nodes of the cluster with the following procedure.



### 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 XFS 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.

### Procedure

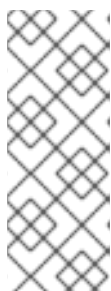
1. On both nodes of the cluster, perform the following steps to set the value for the LVM system ID to the value of the **uname** identifier for the system. The LVM system ID will be used to ensure that only the cluster is capable of activating the volume group.
  - a. Set the **system\_id\_source** 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 XFS 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.



### NOTE

If your LVM volume group contains one or more physical volumes that reside on remote block storage, such as an iSCSI target, Red Hat recommends that you ensure that the service starts before Pacemaker starts. For information about configuring startup order for a remote physical volume used by a Pacemaker cluster, see [Configuring startup order for resource dependencies not managed by Pacemaker](#).

- a. Create an LVM physical volume on partition **/dev/sdb1**.

```
[root@z1 ~]# pvcreate /dev/sdb1
Physical volume "/dev/sdb1" successfully created
```

**NOTE**

If your LVM volume group contains one or more physical volumes that reside on remote block storage, such as an iSCSI target, Red Hat recommends that you ensure that the service starts before Pacemaker starts. For information about configuring startup order for a remote physical volume used by a Pacemaker cluster, see [Configuring startup order for resource dependencies not managed by Pacemaker](#).

- b. Create the volume group **my\_vg** that consists of the physical volume **/dev/sdb1**. For RHEL 8.5 and later, specify the **--setautoactivation n** flag to ensure that volume groups managed by Pacemaker in a cluster will not be automatically activated on startup. If you are using an existing volume group for the LVM volume you are creating, you can reset this flag with the **vgchange --setautoactivation n** command for the volume group.

```
[root@z1 ~]# vgcreate --setautoactivation n my_vg /dev/sdb1
Volume group "my_vg" successfully created
```

For RHEL 8.4 and earlier, create the volume group with the following command.

```
[root@z1 ~]# vgcreate my_vg /dev/sdb1
Volume group "my_vg" successfully created
```

For information on ensuring that volume groups managed by Pacemaker in a cluster will not be automatically activated on startup for RHEL 8.4 and earlier, see [Ensuring a volume group is not activated on multiple cluster nodes](#).

- 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.

```
[root@z1 ~]# 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**.

```
[root@z1 ~]# 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.

```
[root@z1 ~]# lvs
LV   VG   Attr   LSize  Pool Origin Data%  Move Log Copy%  Convert
my_lv my_vg -wi-a---- 452.00m
...
```

- e. Create an XFS file system on the logical volume **my\_lv**.

```
[root@z1 ~]# mkfs.xfs /dev/my_vg/my_lv
meta-data=/dev/my_vg/my_lv  isize=512  agcount=4, agsize=28928 blks
         =                   sectsz=512  attr=2, projid32bit=1
         ...
```

3. If you are using an LVM devices file, supported in RHEL 8.5 and later, add the shared device to the devices file on the second node of the cluster.

```
[root@z2 ~]# lvmdevices --adddev /dev/sdb1
```

## 50.2. ENSURING A VOLUME GROUP IS NOT ACTIVATED ON MULTIPLE CLUSTER NODES (RHEL 8.4 AND EARLIER)

You can ensure that volume groups that are managed by Pacemaker in a cluster will not be automatically activated on startup with the following procedure. If a volume group is automatically activated on startup rather than by Pacemaker, there is a risk that the volume group will be active on multiple nodes at the same time, which could corrupt the volume group's metadata.



### NOTE

For RHEL 8.5 and later, you can disable autoactivation for a volume group when you create the volume group by specifying the **--setautoactivation n** flag for the **vgcreate** command, as described in [Configuring an LVM volume with an XFS file system in a Pacemaker cluster](#).

This procedure modifies the **auto\_activation\_volume\_list** entry in the **/etc/lvm/lvm.conf** configuration file. The **auto\_activation\_volume\_list** entry is used to limit autoactivation to specific logical volumes. Setting **auto\_activation\_volume\_list** to an empty list disables autoactivation entirely.

Any local volumes that are not shared and are not managed by Pacemaker should be included in the **auto\_activation\_volume\_list** entry, including volume groups related to the node's local root and home directories. All volume groups managed by the cluster manager must be excluded from the **auto\_activation\_volume\_list** entry.

### Procedure

Perform the following procedure on each node in the cluster.

1. Determine which volume groups are currently configured on your local storage with the following command. This will output a list of the currently-configured volume groups. If you have space allocated in separate volume groups for root and for your home directory on this node, you will see those volumes in the output, as in this example.

```
# vgs --noheadings -o vg_name
my_vg
rhel_home
rhel_root
```

2. Add the volume groups other than **my\_vg** (the volume group you have just defined for the cluster) as entries to **auto\_activation\_volume\_list** in the **/etc/lvm/lvm.conf** configuration file. For example, if you have space allocated in separate volume groups for root and for your home directory, you would uncomment the **auto\_activation\_volume\_list** line of the **lvm.conf** file and add these volume groups as entries to **auto\_activation\_volume\_list** as follows. Note that the

volume group you have just defined for the cluster (**my\_vg** in this example) is not in this list.

```
auto_activation_volume_list = [ "rhel_root", "rhel_home" ]
```



#### NOTE

If no local volume groups are present on a node to be activated outside of the cluster manager, you must still initialize the **auto\_activation\_volume\_list** entry as **auto\_activation\_volume\_list = []**.

3. Rebuild the **initramfs** boot image to guarantee that the boot image will not try to activate a volume group controlled by the cluster. Update the **initramfs** device with the following command. This command may take up to a minute to complete.

```
# dracut -H -f /boot/initramfs-$(uname -r).img $(uname -r)
```

4. Reboot the node.



#### NOTE

If you have installed a new Linux kernel since booting the node on which you created the boot image, the new **initrd** image will be for the kernel that was running when you created it and not for the new kernel that is running when you reboot the node. You can ensure that the correct **initrd** device is in use by running the **uname -r** command before and after the reboot to determine the kernel release that is running. If the releases are not the same, update the **initrd** file after rebooting with the new kernel and then reboot the node.

5. When the node has rebooted, check whether the cluster services have started up again on that node by executing the **pcs cluster status** command on that node. If this yields the message **Error: cluster is not currently running on this node** then enter the following command.

```
# pcs cluster start
```

Alternately, you can wait until you have rebooted each node in the cluster and start cluster services on all of the nodes in the cluster with the following command.

```
# pcs cluster start --all
```

## 50.3. CONFIGURING AN APACHE HTTP SERVER

Configure an Apache HTTP Server with the following procedure.

### Procedure

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 and enable the ports you will require for running **httpd**. This example enables the **httpd** ports for public access, but the specific ports to enable for **httpd** may vary for production use.

```
# firewall-cmd --permanent --add-service=http
# firewall-cmd --permanent --zone=public --add-service=http
# firewall-cmd --reload
```

2. 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
    Require local
</Location>
END
```

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 three lines.

```
/usr/bin/test -f /run/httpd.pid >/dev/null 2>/dev/null &&
/usr/bin/ps -q $(/usr/bin/cat /run/httpd.pid) >/dev/null 2>/dev/null &&
/usr/sbin/httpd -f /etc/httpd/conf/httpd.conf \
-c "PidFile /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, ensure that the logical volume you created in [Configuring an LVM volume with an XFS file system](#) is activated, mount the file system that you created on that logical volume, create the file **index.html** on that file system, and then unmount the file system.

```
# lvchange -ay my_vg/my_lv
# 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
```

## 50.4. CREATING THE RESOURCES AND RESOURCE GROUPS

Create the resources for your cluster with the following procedure. To ensure these resources all run on the same node, they are configured as part of the resource group **apachegroup**. The resources to create are as follows, listed in the order in which they will start.

1. An **LVM-activate** resource named **my\_lvm** that uses the LVM volume group you created in [Configuring an LVM volume with an XFS 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 XFS file system](#) .
3. An **IPaddr2** resource, which is a floating IP address for the **apachegroup** 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 **apachegroup** 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.

### Procedure

1. The following command creates the **LVM-activate** resource **my\_lvm**. Because the resource group **apachegroup** 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 apachegroup
```

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: apachegroup
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 **apachegroup**.

```
[root@z1 ~]# pcs resource create my_fs Filesystem device="/dev/my_vg/my_lv"
directory="/var/www" fstype="xfs" --group apachegroup
```



```
[root@z1 ~]# pcs resource create VirtualIP IPAddr2 ip=198.51.100.3 cidr_netmask=24 --
group apachegroup
```

```
[root@z1 ~]# pcs resource create Website apache
configfile="/etc/httpd/conf/httpd.conf" statusurl="http://127.0.0.1/server-status" --
group apachegroup
```

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-activate): 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.

4. 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.

## 50.5. TESTING THE RESOURCE CONFIGURATION

Test the resource configuration in a cluster with the following procedure.

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.

### Procedure

1. The following command puts node **z1.example.com** in **standby** mode.

```
[root@z1 ~]# pcs node standby z1.example.com
```

2. 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-activate): 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 node 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. This will depend on the **resource-stickiness** value for the resources. For information on the **resource-stickiness** meta attribute, see [Configuring a resource to prefer its current node](#).

## CHAPTER 51. CONFIGURING AN ACTIVE/PASSIVE NFS SERVER IN A RED HAT HIGH AVAILABILITY CLUSTER

The Red Hat High Availability Add-On provides support for running a highly available active/passive NFS server on a Red Hat Enterprise Linux High Availability Add-On cluster using shared storage. In the following example, you are configuring a two-node cluster in which clients access the NFS file system through a floating IP address. The NFS server runs on one of the 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.

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, Fibre Channel, or other shared network block device.

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 a file system on an LVM logical volume 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.

### 51.1. CONFIGURING AN LVM VOLUME WITH AN XFS FILE SYSTEM IN A PACEMAKER CLUSTER

Create an LVM logical volume on storage that is shared between the nodes of the cluster with the following procedure.



#### 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 XFS 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.

#### Procedure

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

- 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 XFS 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.



#### NOTE

If your LVM volume group contains one or more physical volumes that reside on remote block storage, such as an iSCSI target, Red Hat recommends that you ensure that the service starts before Pacemaker starts. For information about configuring startup order for a remote physical volume used by a Pacemaker cluster, see [Configuring startup order for resource dependencies not managed by Pacemaker](#).

- a. Create an LVM physical volume on partition `/dev/sdb1`.

```
[root@z1 ~]# pvcreate /dev/sdb1
Physical volume "/dev/sdb1" successfully created
```



#### NOTE

If your LVM volume group contains one or more physical volumes that reside on remote block storage, such as an iSCSI target, Red Hat recommends that you ensure that the service starts before Pacemaker starts. For information about configuring startup order for a remote physical volume used by a Pacemaker cluster, see [Configuring startup order for resource dependencies not managed by Pacemaker](#).

- b. Create the volume group **my\_vg** that consists of the physical volume `/dev/sdb1`. For RHEL 8.5 and later, specify the **--setautoactivation n** flag to ensure that volume groups managed by Pacemaker in a cluster will not be automatically activated on startup. If you are using an existing volume group for the LVM volume you are creating, you can reset this flag with the **vgchange --setautoactivation n** command for the volume group.

```
[root@z1 ~]# vgcreate --setautoactivation n my_vg /dev/sdb1
Volume group "my_vg" successfully created
```

For RHEL 8.4 and earlier, create the volume group with the following command.

```
[root@z1 ~]# vgcreate my_vg /dev/sdb1
Volume group "my_vg" successfully created
```

For information on ensuring that volume groups managed by Pacemaker in a cluster will not be automatically activated on startup for RHEL 8.4 and earlier, see [Ensuring a volume group is not activated on multiple cluster nodes](#).

- 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.

```
[root@z1 ~]# 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**.

```
[root@z1 ~]# 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.

```
[root@z1 ~]# lvs
LV   VG   Attr   LSize  Pool Origin Data%  Move Log Copy%  Convert
my_lv my_vg -wi-a---- 452.00m
...
```

- e. Create an XFS file system on the logical volume **my\_lv**.

```
[root@z1 ~]# mkfs.xfs /dev/my_vg/my_lv
meta-data=/dev/my_vg/my_lv  isize=512  agcount=4, agsize=28928 blks
        =                   sectsz=512  attr=2, projid32bit=1
...
```

3. If you are using an LVM devices file, supported in RHEL 8.5 and later, add the shared device to the devices file on the second node of the cluster.

```
[root@z2 ~]# lvmdevices --adddev /dev/sdb1
```

## 51.2. ENSURING A VOLUME GROUP IS NOT ACTIVATED ON MULTIPLE CLUSTER NODES (RHEL 8.4 AND EARLIER)

You can ensure that volume groups that are managed by Pacemaker in a cluster will not be automatically activated on startup with the following procedure. If a volume group is automatically activated on startup rather than by Pacemaker, there is a risk that the volume group will be active on multiple nodes at the same time, which could corrupt the volume group's metadata.



### NOTE

For RHEL 8.5 and later, you can disable autoactivation for a volume group when you create the volume group by specifying the **--setautoactivation n** flag for the **vgcreate** command, as described in [Configuring an LVM volume with an XFS file system in a Pacemaker cluster](#).

This procedure modifies the **auto\_activation\_volume\_list** entry in the `/etc/lvm/lvm.conf` configuration file. The **auto\_activation\_volume\_list** entry is used to limit autoactivation to specific logical volumes. Setting **auto\_activation\_volume\_list** to an empty list disables autoactivation entirely.

Any local volumes that are not shared and are not managed by Pacemaker should be included in the **auto\_activation\_volume\_list** entry, including volume groups related to the node's local root and home directories. All volume groups managed by the cluster manager must be excluded from the **auto\_activation\_volume\_list** entry.

## Procedure

Perform the following procedure on each node in the cluster.

1. Determine which volume groups are currently configured on your local storage with the following command. This will output a list of the currently-configured volume groups. If you have space allocated in separate volume groups for root and for your home directory on this node, you will see those volumes in the output, as in this example.

```
# vgs --noheadings -o vg_name
my_vg
rhel_home
rhel_root
```

2. Add the volume groups other than **my\_vg** (the volume group you have just defined for the cluster) as entries to **auto\_activation\_volume\_list** in the `/etc/lvm/lvm.conf` configuration file. For example, if you have space allocated in separate volume groups for root and for your home directory, you would uncomment the **auto\_activation\_volume\_list** line of the `lvm.conf` file and add these volume groups as entries to **auto\_activation\_volume\_list** as follows. Note that the volume group you have just defined for the cluster (**my\_vg** in this example) is not in this list.

```
auto_activation_volume_list = [ "rhel_root", "rhel_home" ]
```



### NOTE

If no local volume groups are present on a node to be activated outside of the cluster manager, you must still initialize the **auto\_activation\_volume\_list** entry as **auto\_activation\_volume\_list = []**.

3. Rebuild the **initramfs** boot image to guarantee that the boot image will not try to activate a volume group controlled by the cluster. Update the **initramfs** device with the following command. This command may take up to a minute to complete.

```
# dracut -H -f /boot/initramfs-$(uname -r).img $(uname -r)
```

4. Reboot the node.



## NOTE

If you have installed a new Linux kernel since booting the node on which you created the boot image, the new **initrd** image will be for the kernel that was running when you created it and not for the new kernel that is running when you reboot the node. You can ensure that the correct **initrd** device is in use by running the **uname -r** command before and after the reboot to determine the kernel release that is running. If the releases are not the same, update the **initrd** file after rebooting with the new kernel and then reboot the node.

- When the node has rebooted, check whether the cluster services have started up again on that node by executing the **pcs cluster status** command on that node. If this yields the message **Error: cluster is not currently running on this node** then enter the following command.

```
# pcs cluster start
```

Alternately, you can wait until you have rebooted each node in the cluster and start cluster services on all of the nodes in the cluster with the following command.

```
# pcs cluster start --all
```

## 51.3. CONFIGURING AN NFS SHARE

Configure an NFS share for an NFS service failover with the following procedure.

### Procedure

- On both nodes in the cluster, create the **/nfsshare** directory.

```
# mkdir /nfsshare
```

- On one node in the cluster, perform the following procedure.
  - Ensure that the logical volume you created in [Configuring an LVM volume with an XFS file system](#) is activated, then mount the file system you created on the logical volume on the **/nfsshare** directory.

```
[root@z1 ~]# lvchange -ay my_vg/my_lv
[root@z1 ~]# mount /dev/my_vg/my_lv /nfsshare
```

- 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
```

- 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
```

- Unmount the file system and deactivate the LVM volume group.

```
[root@z1 ~]# umount /dev/my_vg/my_lv
[root@z1 ~]# vgchange -an my_vg
```

## 51.4. CONFIGURING THE RESOURCES AND RESOURCE GROUP FOR AN NFS SERVER IN A CLUSTER

Configure the cluster resources for an NFS server in a cluster with the following procedure.



### 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.

### Procedure

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-activate): 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 XFS **Filesystem** resource named **nfsshare** as part of the **nfsgroup** resource group. This file system uses the LVM volume group and XFS file system you created in [Configuring an LVM volume with an XFS 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=xfs --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-activate): 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 **exportfs** 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 exportfs
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 exportfs
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 exportfs
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-activate): 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
...
```

## 51.5. TESTING THE NFS RESOURCE CONFIGURATION

You can validate your NFS resource configuration in a high availability cluster with the following procedures. You should be able to mount the exported file system with either NFSv3 or NFSv4.

### 51.5.1. Testing the NFS export

1. If you are running the **firewalld** daemon on your cluster nodes, ensure that the ports that your system requires for NFS access are enabled on all nodes.
2. 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
```

- 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
```

- 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
```

### 51.5.2. Testing for failover

- On a node outside of the cluster, mount the NFS share and verify access to the **clientdatafile1** file you 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-activate): 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-activate): 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 **nfsgroup** from standby mode.



#### NOTE

Removing a node from **standby** mode does not in itself cause the resources to fail back over to that node. This will depend on the **resource-stickiness** value for the resources. For information on the **resource-stickiness** meta attribute, see [Configuring a resource to prefer its current node](#) .

```
[root@z1 ~]# pcs node unstandby z1.example.com
```

## CHAPTER 52. GFS2 FILE SYSTEMS IN A CLUSTER

Use the following administrative procedures to configure GFS2 file systems in a Red Hat high availability cluster.

### 52.1. CONFIGURING A GFS2 FILE SYSTEM IN A CLUSTER

You can set up a Pacemaker cluster that includes GFS2 file systems with the following procedure. In this example, you create three GFS2 file systems on three logical volumes in a two-node cluster.

#### Prerequisites

- Install and start the cluster software on both cluster nodes and create a basic two-node cluster.
- Configure fencing for the cluster.

For information about creating a Pacemaker cluster and configuring fencing for the cluster, see [Creating a Red Hat High-Availability cluster with Pacemaker](#) .

#### Procedure

1. On both nodes in the cluster, enable the repository for Resilient Storage that corresponds to your system architecture. For example, to enable the Resilient Storage repository for an x86\_64 system, you can enter the following **subscription-manager** command:

```
# subscription-manager repos --enable=rhel-8-for-x86_64-resilientstorage-rpms
```

Note that the Resilient Storage repository is a superset of the High Availability repository. If you enable the Resilient Storage repository you do not also need to enable the High Availability repository.

2. On both nodes of the cluster, install the **lvm2-lockd**, **gfs2-utils**, and **dlm** packages. To support these packages, you must be subscribed to the AppStream channel and the Resilient Storage channel.

```
# yum install lvm2-lockd gfs2-utils dlm
```

3. On both nodes of the cluster, set the **use\_lvmlockd** configuration option in the **/etc/lvm/lvm.conf** file to **use\_lvmlockd=1**.

```
...
use_lvmlockd = 1
...
```

4. Set the global Pacemaker parameter **no-quorum-policy** to **freeze**.

**NOTE**

By default, the value of **no-quorum-policy** is set to **stop**, indicating that once quorum is lost, all the resources on the remaining partition will immediately be stopped. Typically this default is the safest and most optimal option, but unlike most resources, GFS2 requires quorum to function. When quorum is lost both the applications using the GFS2 mounts and the GFS2 mount itself cannot be correctly stopped. Any attempts to stop these resources without quorum will fail which will ultimately result in the entire cluster being fenced every time quorum is lost.

To address this situation, set **no-quorum-policy** to **freeze** when GFS2 is in use. This means that when quorum is lost, the remaining partition will do nothing until quorum is regained.

```
[root@z1 ~]# pcs property set no-quorum-policy=freeze
```

5. 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
```

6. 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
```

7. Set up an **lvmlockd** resource as part of the **locking** resource group.

```
[root@z1 ~]# pcs resource create lvmlockd --group locking ocf:heartbeat:lvmlockd op
monitor interval=30s on-fail=fence
```

8. 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 ]
```

9. 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.



#### NOTE

If your LVM volume group contains one or more physical volumes that reside on remote block storage, such as an iSCSI target, Red Hat recommends that you ensure that the service starts before Pacemaker starts. For information about configuring startup order for a remote physical volume used by a Pacemaker cluster, see [Configuring startup order for resource dependencies not managed by Pacemaker](#).

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...
```

10. On the second node in the cluster:

- a. If you are using an LVM devices file, supported in RHEL 8.5 and later, add the shared devices to the devices file.

```
[root@z2 ~]# lvmdevices --adddev /dev/vdb
[root@z2 ~]# lvmdevices --adddev /dev/vdc
```

- b. Start the lock manager for each of the shared volume groups.

```
[root@z2 ~]# vgchange --lockstart shared_vg1
VG shared_vg1 starting dlm lockspace
Starting locking. Waiting until locks are ready...
[root@z2 ~]# vgchange --lockstart shared_vg2
VG shared_vg2 starting dlm lockspace
Starting locking. Waiting until locks are ready...
```

11. 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. The format of the lock table name is *ClusterName:FSName* where *ClusterName* is the name of the cluster for which the GFS2 file system is being created and *FSName* is the file system name, which must be unique for all **lock\_dlm** file systems over the 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
```

12. 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
```

13. Clone the two new resource groups.

```
[root@z1 ~]# pcs resource clone shared_vg1 interleave=true
[root@z1 ~]# pcs resource clone shared_vg2 interleave=true
```

14. 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)
```

- 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
```

- 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
```

- 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 to display the 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
```

## Verification steps

- 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)
```

2. Check the status of the cluster.

```
[root@z1 ~]# pcs status --full
Cluster name: my_cluster
[...]

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
  sharedfs1 (ocf::heartbeat:Filesystem): Started z2.example.com
  sharedfs2 (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
  sharedfs1 (ocf::heartbeat:Filesystem): Started z1.example.com
  sharedfs2 (ocf::heartbeat:Filesystem): Started z1.example.com
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
  sharedfs3 (ocf::heartbeat:Filesystem): Started z2.example.com
Resource Group: shared_vg2:1
  sharedlv3 (ocf::heartbeat:LVM-activate): Started z1.example.com
  sharedfs3 (ocf::heartbeat:Filesystem): Started z1.example.com
Started: [ z1.example.com z2.example.com ]

...
```

#### Additional resources

- [Configuring GFS2 file systems](#)
- [Configuring a Red Hat High Availability cluster on Microsoft Azure](#)
- [Configuring a Red Hat High Availability cluster on AWS](#)
- [Configuring Red Hat High Availability Cluster on Google Cloud Platform](#)
- [Configuring shared block storage for a Red Hat High Availability cluster on Alibaba Cloud](#)

## 52.2. CONFIGURING AN ENCRYPTED GFS2 FILE SYSTEM IN A CLUSTER

(RHEL 8.4 and later) You can create a Pacemaker cluster that includes a LUKS encrypted GFS2 file system with the following procedure. In this example, you create one GFS2 file systems on a logical volume and encrypt the file system. Encrypted GFS2 file systems are supported using the **crypt** resource agent, which provides support for LUKS encryption.

There are three parts to this procedure:

- Configuring a shared logical volume in a Pacemaker cluster
- Encrypting the logical volume and creating a **crypt** resource
- Formatting the encrypted logical volume with a GFS2 file system and creating a file system resource for the cluster

### 52.2.1. Configure a shared logical volume in a Pacemaker cluster

#### Prerequisites

- Install and start the cluster software on two cluster nodes and create a basic two-node cluster.
- Configure fencing for the cluster.

For information about creating a Pacemaker cluster and configuring fencing for the cluster, see [Creating a Red Hat High-Availability cluster with Pacemaker](#) .

#### Procedure

1. On both nodes in the cluster, enable the repository for Resilient Storage that corresponds to your system architecture. For example, to enable the Resilient Storage repository for an x86\_64 system, you can enter the following **subscription-manager** command:

```
# subscription-manager repos --enable=rhel-8-for-x86_64-resilientstorage-rpms
```

Note that the Resilient Storage repository is a superset of the High Availability repository. If you enable the Resilient Storage repository you do not also need to enable the High Availability repository.

2. On both nodes of the cluster, install the **lvm2-lockd**, **gfs2-utils**, and **dlm** packages. To support these packages, you must be subscribed to the AppStream channel and the Resilient Storage channel.

```
# yum install lvm2-lockd gfs2-utils dlm
```

3. On both nodes of the cluster, set the **use\_lvmlockd** configuration option in the **/etc/lvm/lvm.conf** file to **use\_lvmlockd=1**.

```
...
use_lvmlockd = 1
...
```

4. Set the global Pacemaker parameter **no-quorum-policy** to **freeze**.

**NOTE**

By default, the value of **no-quorum-policy** is set to **stop**, indicating that when quorum is lost, all the resources on the remaining partition will immediately be stopped. Typically this default is the safest and most optimal option, but unlike most resources, GFS2 requires quorum to function. When quorum is lost both the applications using the GFS2 mounts and the GFS2 mount itself cannot be correctly stopped. Any attempts to stop these resources without quorum will fail which will ultimately result in the entire cluster being fenced every time quorum is lost.

To address this situation, set **no-quorum-policy** to **freeze** when GFS2 is in use. This means that when quorum is lost, the remaining partition will do nothing until quorum is regained.

```
[root@z1 ~]# pcs property set no-quorum-policy=freeze
```

- 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
```

- 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
```

- 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
```

- 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 ]
```

9. On one node of the cluster, create a shared volume group.



#### NOTE

If your LVM volume group contains one or more physical volumes that reside on remote block storage, such as an iSCSI target, Red Hat recommends that you ensure that the service starts before Pacemaker starts. For information on configuring startup order for a remote physical volume used by a Pacemaker cluster, see [Configuring startup order for resource dependencies not managed by Pacemaker](#).

The following command creates the shared volume group **shared\_vg1** on **/dev/sda1**.

```
[root@z1 ~]# vgcreate --shared shared_vg1 /dev/sda1
Physical volume "/dev/sda1" successfully created.
Volume group "shared_vg1" successfully created
VG shared_vg1 starting dlm lockspace
Starting locking. Waiting until locks are ready...
```

10. On the second node in the cluster:

- a. If you are using an LVM devices file, supported in RHEL 8.5 and later, add the shared device to the devices file.

```
[root@z2 ~]# lvmddevices --adddev /dev/sda1
```

- b. Start the lock manager for the shared volume group.

```
[root@z2 ~]# vgchange --lockstart shared_vg1
VG shared_vg1 starting dlm lockspace
Starting locking. Waiting until locks are ready...
```

11. On one node in the cluster, create the shared logical volume.

```
[root@z1 ~]# lvcreate --activate sy -L5G -n shared_lv1 shared_vg1
Logical volume "shared_lv1" created.
```

12. Create an **LVM-activate** resource for the logical volume to automatically activate the logical volume on all nodes.

The following command creates 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=lvmlckd
```

13. Clone the new resource group.

```
[root@z1 ~]# pcs resource clone shared_vg1 interleave=true
```

- Configure an 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)
```

- Configure a 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
```

## Verification steps

On both nodes in the cluster, verify that the logical volume is 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
```

```
[root@z2 ~]# lvs
LV      VG      Attr      LSize
shared_lv1 shared_vg1 -wi-a----- 5.00g
```

## 52.2.2. Encrypt the logical volume and create a crypt resource

### Prerequisites

- You have configured a shared logical volume in a Pacemaker cluster.

### Procedure

- On one node in the cluster, create a new file that will contain the crypt key and set the permissions on the file so that it is readable only by root.

```
[root@z1 ~]# touch /etc/crypt_keyfile
[root@z1 ~]# chmod 600 /etc/crypt_keyfile
```

- Create the crypt key.

```
[root@z1 ~]# dd if=/dev/urandom bs=4K count=1 of=/etc/crypt_keyfile
1+0 records in
1+0 records out
4096 bytes (4.1 kB, 4.0 KiB) copied, 0.000306202 s, 13.4 MB/s
[root@z1 ~]# scp /etc/crypt_keyfile root@z2.example.com:/etc/
```

- Distribute the crypt keyfile to the other nodes in the cluster, using the **-p** parameter to preserve the permissions you set.

```
[root@z1 ~]# scp -p /etc/crypt_keyfile root@z2.example.com:/etc/
```

4. Create the encrypted device on the LVM volume where you will configure the encrypted GFS2 file system.

```
[root@z1 ~]# cryptsetup luksFormat /dev/shared_vg1/shared_lv1 --type luks2 --key-
file=/etc/crypt_keyfile
WARNING!
=====
This will overwrite data on /dev/shared_vg1/shared_lv1 irrevocably.

Are you sure? (Type 'yes' in capital letters): YES
```

5. Create the crypt resource as part of the **shared\_vg1** volume group.

```
[root@z1 ~]# pcs resource create crypt --group shared_vg1 ocf:heartbeat:crypt
crypt_dev="luks_lv1" crypt_type=luks2 key_file=/etc/crypt_keyfile
encrypted_dev="/dev/shared_vg1/shared_lv1"
```

### Verification steps

Ensure that the crypt resource has created the crypt device, which in this example is **/dev/mapper/luks\_lv1**.

```
[root@z1 ~]# ls -l /dev/mapper/
...
lrwxrwxrwx 1 root root 7 Mar 4 09:52 luks_lv1 -> ../dm-3
...
```

### 52.2.3. Format the encrypted logical volume with a GFS2 file system and create a file system resource for the cluster

#### Prerequisites

- You have encrypted the logical volume and created a crypt resource.

#### Procedure

1. On one node in the cluster, format the volume 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. The format of the lock table name is *ClusterName:FSName* where *ClusterName* is the name of the cluster for which the GFS2 file system is being created and *FSName* is the file system name, which must be unique for all **lock\_dlm** file systems over the cluster.

```
[root@z1 ~]# mkfs.gfs2 -j3 -p lock_dlm -t my_cluster:gfs2-demo1 /dev/mapper/luks_lv1
/dev/mapper/luks_lv1 is a symbolic link to /dev/dm-3
This will destroy any data on /dev/dm-3
Are you sure you want to proceed? [y/n] y
Discarding device contents (may take a while on large devices): Done
Adding journals: Done
Building resource groups: Done
Creating quota file: Done
Writing superblock and syncing: Done
Device:                /dev/mapper/luks_lv1
Block size:            4096
```



```

Device size:          4.98 GB (1306624 blocks)
Filesystem size:     4.98 GB (1306622 blocks)
Journals:            3
Journal size:        16MB
Resource groups:     23
Locking protocol:    "lock_dlm"
Lock table:          "my_cluster:gfs2-demo1"
UUID:                de263f7b-0f12-4d02-bbb2-56642fade293

```

2. Create a file system resource to automatically mount the GFS2 file system on all nodes. Do 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 command creates the file system resource. This command adds the 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/mapper/luks_lv1" directory="/mnt/gfs1"
fstype="gfs2" options=noatime op monitor interval=10s on-fail=fence

```

### Verification steps

1. Verify that the GFS2 file system is mounted on both nodes of the cluster.

```

[root@z1 ~]# mount | grep gfs2
/dev/mapper/luks_lv1 on /mnt/gfs1 type gfs2 (rw,noatime,seclabel)

```

```

[root@z2 ~]# mount | grep gfs2
/dev/mapper/luks_lv1 on /mnt/gfs1 type gfs2 (rw,noatime,seclabel)

```

2. Check the status of the cluster.

```

[root@z1 ~]# pcs status --full
Cluster name: my_cluster
[...]

```

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
  crypt (ocf::heartbeat:crypt) Started z2.example.com
  sharedfs1 (ocf::heartbeat:Filesystem): Started z2.example.com

```

```
Resource Group: shared_vg1:1
  sharedlv1 (ocf::heartbeat:LVM-activate): Started z1.example.com
  crypt (ocf::heartbeat:crypt) Started z1.example.com
  sharedfs1 (ocf::heartbeat:Filesystem): Started z1.example.com
  Started: [z1.example.com z2.example.com ]
```

...

### Additional resources

- [Configuring GFS2 file systems](#)

## 52.3. MIGRATING A GFS2 FILE SYSTEM FROM RHEL7 TO RHEL8

You can use your existing Red Hat Enterprise 7 logical volumes when configuring a RHEL 8 cluster that includes GFS2 file systems.

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.

### Procedure

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 --lockstart upgrade_gfs_vg
VG upgrade_gfs_vg starting dlm lockspace
Starting locking. Waiting until locks are ready...
```

```
[root@rhel8-02 ~]# vgchange --lockstart 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 53. 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.

### 53.1. DISPLAYING AVAILABLE FENCE AGENTS AND THEIR OPTIONS

The following commands can be used to view available fencing agents and the available options for specific fencing agents.

This command lists all available fencing agents. When you specify a filter, this command displays only the fencing agents that match the filter.

```
pcs stonith list [filter]
```

This command displays the options for the specified fencing 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.

## 53.2. CREATING A FENCE DEVICE

The format for the command to create a fence device is as follows. For a listing of the available fence device 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](#).

## 53.3. GENERAL PROPERTIES OF FENCING DEVICES

There are many general properties you can set for fencing devices, as well as various cluster properties that determine fencing behavior.

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.

The following table describes the general properties you can set for fencing devices.

**Table 53.1. 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. As of RHEL 8.7, the <b>pcmk_host_map</b> property supports special characters inside <b>pcmk_host_map</b> values using a backslash in front of the value. For example, you can specify <b>pcmk_host_map="node3:plug\ 1"</b> to include a space in the host alias.
<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)

The following table summarizes additional properties you can set for fencing devices. Note that these properties are for advanced use only.

**Table 53.2. 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.

Field	Type	Default	Description
<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.
<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.



Field	Type	Default	Description
<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.
<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>	string	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>As of Red Hat Enterprise Linux 8.6, you can specify different values for different nodes with the <b>pcmk_delay_base</b> parameter. This allows a single fence device to be used in a two-node cluster, with a different delay for each node. This helps prevent a situation where each node attempts to fence the other node at the same time. To specify different values for different nodes, you map the host names to the delay value for that node using a similar syntax to <b>pcmk_host_map</b>. For example, <b>node1:0;node2:10s</b> would use no delay when fencing <b>node1</b> and a 10-second delay when fencing <b>node2</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>

Field	Type	Default	Description
<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>
<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.

Field	Type	Default	Description
<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 the following table.

**Table 53.3. Cluster Properties that Determine Fencing Behavior**

Option	Default	Description
<b>stonith-enabled</b>	true	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> .  If <b>true</b> , or unset, the cluster will refuse to start resources unless one or more STONITH resources have been configured also.  Red Hat only supports clusters with this value set to <b>true</b> .
<b>stonith-action</b>	reboot	Action to send to STONITH device. Allowed values: <b>reboot</b> , <b>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-only SBD configuration is used for fencing.

Option	Default	Description
<b>concurrent-fencing</b>	true (RHEL 8.1 and later)	Allow fencing operations to be performed in parallel.
<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](#).

## 53.4. TESTING A FENCE DEVICE

Fencing is a fundamental part of the Red Hat Cluster infrastructure and it is important to validate or test that fencing is working properly.

### Procedure

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**

These examples 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
```

## 53.5. 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.

Pacemaker processes fencing levels as follows:

- 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\_ilo** and an apc fence device called **my\_apc**. These commands set up fence levels so that if the device **my\_ilo** 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_ilo
# pcs stonith level add 2 rh7-2 my_apc
# pcs stonith level
Node: rh7-2
Level 1 - my_ilo
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
```

## 53.6. 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
```

## 53.7. 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]
```

## 53.8. EXPORTING FENCE DEVICES AS `pcs` COMMANDS

As of Red Hat Enterprise Linux 8.7, you can display the `pcs` commands that can be used to re-create configured fence devices on a different system using the `--output-format=cmd` option of the `pcs stonith config` command.

The following commands create a `fence_apc_snmp` fence device and display the `pcs` command you can use to re-create the device.

```
# pcs stonith create myapc fence_apc_snmp ip="zpc.example.com"
pcmk_host_map="z1.example.com:1;z2.example.com:2" username="apc" password="apc"
# pcs stonith config --output-format=cmd
Warning: Only 'text' output format is supported for stonith levels
pcs stonith create --no-default-ops --force -- myapc fence_apc_snmp \
  ip=zpc.example.com password=apc 'pcmk_host_map=z1.example.com:1;z2.example.com:2'
username=apc \
  op \
  monitor interval=60s id=myapc-monitor-interval-60s
```

## 53.9. MODIFYING AND DELETING FENCE DEVICES

Modify or add options to a currently configured fencing device with the following command.

```
pcs stonith update stonith_id [stonith_device_options]
```

Updating a SCSI fencing device with the `pcs stonith update` command causes a restart of all resources running on the same node where the stonith resource was running. As of RHEL 8.5, you can use either version of the following command to update SCSI devices without causing a restart of other cluster resources. As of RHEL 8.7, SCSI fencing devices can be configured as multipath devices.

```
pcs stonith update-scsi-devices stonith_id set device-path1 device-path2
pcs stonith update-scsi-devices stonith_id add device-path1 remove device-path2
```

Use the following command to remove a fencing device from the current configuration.

```
pcs stonith delete stonith_id
```

## 53.10. 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 fence 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
```

## 53.11. DISABLING A FENCE DEVICE

To disable a fencing device/resource, run the **pcs stonith disable** command.

The following command disables the fence device **myapc**.

```
# pcs stonith disable myapc
```

## 53.12. PREVENTING A NODE FROM USING A FENCING DEVICE

To prevent a specific node from using a fencing device, you can configure location constraints for the fencing resource.

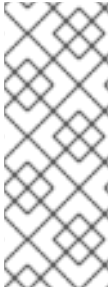
The following example prevents fence device **node1-ipmi** from running on **node1**.

```
# pcs constraint location node1-ipmi avoids node1
```

## 53.13. 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.

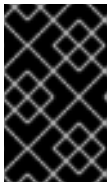
**NOTE**

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 "Disabling ACPI Soft-Off with the Bios" below.

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 "Disabling ACPI Soft-Off in the logind.conf file", below. This is the first alternate method of disabling ACPI Soft-Off.
- Appending **acpi=off** to the kernel boot command line, as described in "Disabling ACPI completely in the GRUB 2 file", below. 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.

### 53.13.1. 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.

**Procedure**

1. Reboot the node and start the **BIOS CMOS Setup Utility** program.
2. Navigate to the Power menu (or equivalent power management menu).
3. 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). The **BIOS CMOS Setup Utility** example below 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.

4. Exit the **BIOS CMOS Setup Utility** program, saving the BIOS configuration.
5. 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 THRM-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**.

**53.13.2. 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.

**Procedure**

1. 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](#).

### 53.13.3. 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.

#### Procedure

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 54. CONFIGURING CLUSTER RESOURCES

Create and delete cluster resources with the following commands.

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] [--wait[=n]]
```

Key cluster resource creation options include the following:

- 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.

There is no limit to the number of resources you can create in a cluster.

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

Delete a configured resource with the following command.

```
pcs resource delete resource_id
```

For example, the following command deletes an existing resource with a resource ID of **VirtualIP**.

```
# pcs resource delete VirtualIP
```

## 54.1. 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.

The following table describes these properties of a resource agent.

**Table 54.1. 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.

The following table summarizes the commands that display the available resource properties.

**Table 54.2. 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.

## 54.2. 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.

...
```

### 54.3. 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.

The following table describes the resource meta options.

**Table 54.3. 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>	<p>Indicates what state the cluster should attempt to keep this resource in. Allowed values:</p> <ul style="list-style-type: none"> <li>* <b>Stopped</b> - Force the resource to be stopped</li> <li>* <b>Started</b> - Allow the resource to be started (and in the case of promotable clones, promoted to master role if appropriate)</li> <li>* <b>Master</b> - Allow the resource to be started and, if appropriate, promoted</li> <li>* <b>Slave</b> - Allow the resource to be started, but only in slave mode if the resource is promotable</li> </ul> <p>As of RHEL 8.5, The <b>pcs</b> command-line interface accepts <b>Promoted</b> and <b>Unpromoted</b> anywhere roles are specified in Pacemaker configuration. These role names are the functional equivalent of the <b>Master</b> and <b>Slave</b> Pacemaker roles.</p>

Field	Default	Description
<b>is-managed</b>	<b>true</b>	Indicates whether the cluster is allowed to start and stop the resource. Allowed values: <b>true, false</b>
<b>resource-stickiness</b>	0	Value to indicate how much the resource prefers to stay where it is. For information on this attribute, see <a href="#">Configuring a resource to prefer its current node</a> .
<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.
<b>multiple-active</b>	<b>stop_start</b>	<p>Indicates what the cluster should do if it ever finds the resource active on more than one node. Allowed values:</p> <ul style="list-style-type: none"> <li>* <b>block</b> - mark the resource as unmanaged</li> <li>* <b>stop_only</b> - stop all active instances and leave them that way</li> <li>* <b>stop_start</b> - stop all active instances and start the resource in one location only</li> <li>* <b>stop_unexpected</b> - (RHEL 8.7 and later) stop only unexpected instances of the resource, without requiring a full restart. It is the user's responsibility to verify that the service and its resource agent can function with extra active instances without requiring a full restart.</li> </ul>
<b>critical</b>	<b>true</b>	(RHEL 8.4 and later) Sets the default value for the <b>influence</b> option for all colocation constraints involving the resource as a dependent resource ( <i>target_resource</i> ), including implicit colocation constraints created when the resource is part of a resource group. The <b>influence</b> colocation constraint option determines whether the cluster will move both the primary and dependent resources to another node when the dependent resource reaches its migration threshold for failure, or whether the cluster will leave the dependent resource offline without causing a service switch. The <b>critical</b> resource meta option can have a value of <b>true</b> or <b>false</b> , with a default value of <b>true</b> .

Field	Default	Description
<b>allow-unhealthy-nodes</b>	<b>false</b>	(RHEL 8.7 and later) When set to <b>true</b> , the resource is not forced off a node due to degraded node health. When health resources have this attribute set, the cluster can automatically detect if the node's health recovers and move resources back to it. A node's health is determined by a combination of the health attributes set by health resource agents based on local conditions, and the strategy-related options that determine how the cluster reacts to those conditions.

### 54.3.1. Changing the default value of a resource option

As of Red Hat Enterprise Linux 8.3, you can change the default value of a resource option for all resources with the **pcs resource defaults update** command. The following command resets the default value of **resource-stickiness** to 100.

```
# pcs resource defaults update resource-stickiness=100
```

The original **pcs resource defaults name=value** command, which set defaults for all resources in previous releases, remains supported unless there is more than one set of defaults configured. However, **pcs resource defaults update** is now the preferred version of the command.

### 54.3.2. Changing the default value of a resource option for sets of resources

As of Red Hat Enterprise Linux 8.3, you can create multiple sets of resource defaults with the **pcs resource defaults set create** command, which allows you to specify a rule that contains **resource** expressions. In RHEL 8.3, only **resource** expressions, including **and**, **or** and parentheses, are allowed in rules that you specify with this command. In RHEL 8.4 and later, only **resource** and **date** expressions, including **and**, **or** and parentheses, are allowed in rules that you specify with this command.

With the **pcs resource defaults set create** command, you can configure a default resource value for all resources of a particular type. If, for example, you are running databases which take a long time to stop, you can increase the **resource-stickiness** default value for all resources of the database type to prevent those resources from moving to other nodes more often than you desire.

The following command sets the default value of **resource-stickiness** to 100 for all resources of type **pgsql**.

- The **id** option, which names the set of resource defaults, is not mandatory. If you do not set this option **pcs** will generate an ID automatically. Setting this value allows you to provide a more descriptive name.
- In this example, **::pgsql** means a resource of any class, any provider, of type **pgsql**.
  - Specifying **ocf:heartbeat:pgsql** would indicate class **ocf**, provider **heartbeat**, type **pgsql**,
  - Specifying **ocf:pacemaker:** would indicate all resources of class **ocf**, provider **pacemaker**, of any type.

```
# pcs resource defaults set create id=pgsql-stickiness meta resource-stickiness=100 rule
resource ::pgsql
```

To change the default values in an existing set, use the **pcs resource defaults set update** command.

### 54.3.3. Displaying currently configured resource defaults

The **pcs resource defaults** command displays a list of currently configured default values for resource options, including any rules you specified.

The following example shows the output of this command after you have reset the default value of **resource-stickiness** to 100.

```
# pcs resource defaults
Meta Attrs: rsc_defaults-meta_attributes
resource-stickiness=100
```

The following example shows the output of this command after you have reset the default value of **resource-stickiness** to 100 for all resources of type **pgsql** and set the **id** option to **id=pgsql-stickiness**.

```
# pcs resource defaults
Meta Attrs: pgsql-stickiness
resource-stickiness=100
Rule: boolean-op=and score=INFINITY
Expression: resource ::pgsql
```

### 54.3.4. 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, or cloned 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
...
```

## 54.4. 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.

### 54.4.1. 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.

### 54.4.2. 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...
```

### 54.4.3. Displaying resource groups

The following command lists all currently configured resource groups.

```
pcs resource group list
```

### 54.4.4. 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](#).

### 54.4.5. 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.

## 54.5. 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.

## CHAPTER 55. 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](#) .

### 55.1. 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]] ...
```

The following table summarizes the meanings of the basic options for configuring location constraints.

**Table 55.1. 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 <b>score</b> in a <b>pcs constraint location rsc prefers</b> command 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.</p> <p>A value of <b>INFINITY</b> for <b>score</b> in a <b>pcs constraint location rsc avoids</b> command indicates that the resource will never run on that node, even if no other node is available. This is the equivalent of setting a <b>pcs constraint location add</b> command with a score of <b>INFINITY</b>.</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
```

## 55.2. 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]
```

The following table summarizes the meanings of the basic parameters for configuring constraints for resource discovery.

**Table 55.2. 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 <b>score</b> 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> for <b>score</b> 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	<p>* <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.</p> <p>* <b>never</b> - Never perform resource discovery for the specified resource on this node.</p> <p>* <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.</p>
-----------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------



### 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).

## 55.3. 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.

### 55.3.1. 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
```

### 55.3.2. 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 "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.

## 55.4. 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.

With a **resource-stickiness** value of 0, a cluster may move resources as needed to balance resources across nodes. This may result in resources moving when unrelated resources start or stop. With a positive stickiness, resources have a preference to stay where they are, and move only if other circumstances outweigh the stickiness. This may result in newly-added nodes not getting any resources assigned to them without administrator intervention.

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 update resource-stickiness=1**

With a positive **resource-stickiness** value, 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 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 56. 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]
```

The following table summarizes the properties and options for configuring ordering constraints.

**Table 56.1. 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 be ordered on the resource. Possible values of the <i>action</i> property are as follows:</p> <ul style="list-style-type: none"> <li>* <b>start</b> - Order start actions of the resource.</li> <li>* <b>stop</b> - Order stop actions of the resource.</li> <li>* <b>promote</b> - Promote the resource from a slave (unpromoted) resource to a master (promoted) resource.</li> <li>* <b>demote</b> - Demote the resource from a master (promoted) resource to a slave (unpromoted) 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 enforce the constraint (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 A stops). 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>Optional</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]...
```

## 56.1. 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.

## 56.2. 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
```

## 56.3. 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.

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 the "Properties of an Order Constraint" table in [Determining the order in which cluster resources are run](#) .
- **role**, which can be set to **Stopped**, **Started**, **Master**, or **Slave**. As of RHEL 8.5, the **pcs** command-line interface accepts **Promoted** and **Unpromoted** as a value for **role**. The **Promoted** and **Unpromoted** roles are the functional equivalent of the **Master** and **Slave** roles.

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 the "Properties of an Order Constraint" table in [Determining the order in which cluster resources are run](#) .
- **symmetrical**, to set whether the reverse of the constraint applies for the opposite action, as described in the "Properties of an Order Constraint" table in [Determining the order in which cluster resources are run](#) .

```
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
```

## 56.4. 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.

If an LVM volume group used by a Pacemaker cluster contains one or more physical volumes that reside on remote block storage, such as an iSCSI target, you can configure a **systemd resource-agents-deps** target and a **systemd** drop-in unit for the target to ensure that the service starts before Pacemaker starts.

The following procedure configures **blk-availability.service** as a dependency. The **blk-availability.service** service is a wrapper that includes **iscsi.service**, among other services. If your deployment requires it, you could configure **iscsi.service** (for iSCSI only) or **remote-fs.target** as the dependency instead of **blk-availability**.

1. Create the drop-in unit **/etc/systemd/system/resource-agents-deps.target.d/blk-availability.conf** that contains the following:

```
[Unit]
Requires=blk-availability.service
After=blk-availability.service
```

2. Run the **systemctl daemon-reload** command.

## CHAPTER 57. 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]
```

The following table summarizes the properties and options for configuring colocation constraints.

**Table 57.1. Parameters of a Colocation Constraint**

Parameter	Description
<code>source_resource</code>	The colocation source. If the constraint cannot be satisfied, the cluster may decide not to allow the resource to run at all.
<code>target_resource</code>	The colocation target. The cluster will decide where to put this resource first and then decide where to put the source resource.
<code>score</code>	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> .

Parameter	Description
<b>influence</b> option	<p>(RHEL 8.4 and later) Determines whether the cluster will move both the primary resource (<i>source_resource</i>) and dependent resources (<i>target_resource</i>) to another node when the dependent resource reaches its migration threshold for failure, or whether the cluster will leave the dependent resource offline without causing a service switch.</p> <p>The <b>influence</b> colocation constraint option can have a value of <b>true</b> or <b>false</b>. The default value for this option is determined by the value of the dependent resource's <b>critical</b> resource meta option, which has a default value of <b>true</b>.</p> <p>When this option has a value of <b>true</b>, Pacemaker will attempt to keep both the primary and dependent resource active. If the dependent resource reaches its migration threshold for failures, both resources will move to another node if possible.</p> <p>When this option has a value of <b>false</b>, Pacemaker will avoid moving the primary resource as a result of the status of the dependent resource. In this case, if the dependent resource reaches its migration threshold for failures, it will stop if the primary resource is active and can remain on its current node.</p>

## 57.1. 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.

## 57.2. SPECIFYING ADVISORY PLACEMENT OF RESOURCES

Advisory placement of resources indicates the placement of resources is a preference, but is not

mandatory. 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.

### 57.3. 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. 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 the "Location Constraint Options" table in [Configuring Location Constraints](#)

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]]
```

Use the following command to remove colocation constraints with *source\_resource*.

```
pcs constraint colocation remove source_resource target_resource
```

## CHAPTER 58. DISPLAYING RESOURCE CONSTRAINTS AND RESOURCE DEPENDENCIES

There are a several commands you can use to display constraints that have been configured. You can display all configured resource constraints, or you can limit the display of resource constraints to specific types of resource constraints. Additionally, you can display configured resource dependencies.

### 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 in the listing, 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.

### 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]
```

### Displaying ordering constraints

The following command lists all current ordering constraints.

```
pcs constraint order [show]
```

### Displaying colocation constraints

The following command lists all current colocation constraints.

```
pcs constraint colocation [show]
```

### Displaying resource-specific constraints

The following command lists the constraints that reference specific resources.

```
pcs constraint ref resource ...
```

### 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 59. DETERMINING RESOURCE LOCATION WITH RULES

For more complicated location constraints, you can use Pacemaker rules to determine a resource's location.

### 59.1. PACEMAKER RULES

Pacemaker 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 59.1. 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> .

#### 59.1.1. Node attribute expressions

Node attribute expressions are used to control a resource based on the attributes defined by a node or nodes.

Table 59.2. 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>number</b> (RHEL 8.4 and later), <b>version</b> . The default value is <b>string</b> .
<b>operation</b>	The comparison to perform. Allowed values:  <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 the following table.

**Table 59.3. 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.

### 59.1.2. 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 59.4. 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>	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: <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>

### 59.1.3. 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 59.5. 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).

## 59.2. 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, and moon as described in the "Properties of a Date Specification" table in [Date specifications](#).

- **defined|not\_defined *attribute***
- ***attribute* lt|gt|lte|gte|eq|ne [*string|integer|number*(RHEL 8.4 and later)|*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 60. MANAGING CLUSTER RESOURCES

There are a variety of commands you can use to display, modify, and administer cluster resources.

### 60.1. 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 status** command yields the following output.

```
# pcs resource status
VirtualIP (ocf::heartbeat:IPAddr2): Started
WebSite (ocf::heartbeat:apache): Started
```

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
```

As of RHEL 8.5, to display the status of an individual resource, use the following command.

```
pcs resource status resource_id
```

For example, if your system is configured with a resource named **VirtualIP** the **pcs resource status VirtualIP** command yields the following output.

```
# pcs resource status VirtualIP
VirtualIP (ocf::heartbeat:IPAddr2): Started
```

As of RHEL 8.5, to display the status of the resources running on a specific node, use the following command. You can use this command to display the status of resources on both cluster and remote nodes.

```
pcs resource status node=node_id
```

For example, if **node-01** is running resources named **VirtualIP** and **WebSite** the **pcs resource status node=node-01** command might yield the following output.

```
# pcs resource status node=node-01
VirtualIP (ocf::heartbeat:IPAddr2): Started
WebSite (ocf::heartbeat:apache): Started
```

## 60.2. EXPORTING CLUSTER RESOURCES AS `pcs` COMMANDS

As of Red Hat Enterprise Linux 8.7, you can display the `pcs` commands that can be used to re-create configured cluster resources on a different system using the `--output-format=cmd` option of the `pcs resource config` command.

The following commands create four resources created for an active/passive Apache HTTP server in a Red Hat high availability cluster: an `LVM-activate` resource, a `Filesystem` resource, an `IPaddr2` resource, and an `Apache` resource.

```
# pcs resource create my_lvm ocf:heartbeat:LVM-activate vgname=my_vg
vg_access_mode=system_id --group apachegroup
# pcs resource create my_fs Filesystem device="/dev/my_vg/my_lv" directory="/var/www"
fstype="xfs" --group apachegroup
# pcs resource create VirtualIP IPaddr2 ip=198.51.100.3 cidr_netmask=24 --group apachegroup
# pcs resource create Website apache configfile="/etc/httpd/conf/httpd.conf"
statusurl="http://127.0.0.1/server-status" --group apachegroup
```

After you create the resources, the following command displays the `pcs` commands you can use to re-create those resources on a different system.

```
# pcs resource config --output-format=cmd
pcs resource create --no-default-ops --force -- my_lvm ocf:heartbeat:LVM-activate \
  vg_access_mode=system_id vgname=my_vg \
  op \
  monitor interval=30s id=my_lvm-monitor-interval-30s timeout=90s \
  start interval=0s id=my_lvm-start-interval-0s timeout=90s \
  stop interval=0s id=my_lvm-stop-interval-0s timeout=90s;
pcs resource create --no-default-ops --force -- my_fs ocf:heartbeat:Filesystem \
  device=/dev/my_vg/my_lv directory=/var/www fstype=xfs \
  op \
  monitor interval=20s id=my_fs-monitor-interval-20s timeout=40s \
  start interval=0s id=my_fs-start-interval-0s timeout=60s \
  stop interval=0s id=my_fs-stop-interval-0s timeout=60s;
pcs resource create --no-default-ops --force -- VirtualIP ocf:heartbeat:IPaddr2 \
  cidr_netmask=24 ip=198.51.100.3 \
  op \
  monitor interval=10s id=VirtualIP-monitor-interval-10s timeout=20s \
  start interval=0s id=VirtualIP-start-interval-0s timeout=20s \
  stop interval=0s id=VirtualIP-stop-interval-0s timeout=20s;
pcs resource create --no-default-ops --force -- Website ocf:heartbeat:apache \
  configfile=/etc/httpd/conf/httpd.conf statusurl=http://127.0.0.1/server-status \
  op \
  monitor interval=10s id=Website-monitor-interval-10s timeout=20s \
  start interval=0s id=Website-start-interval-0s timeout=40s \
  stop interval=0s id=Website-stop-interval-0s timeout=60s;
pcs resource group add apachegroup \
  my_lvm my_fs VirtualIP Website
```

To display the `pcs` command or commands you can use to re-create only one configured resource, specify the resource ID for that resource.

```
# pcs resource config VirtualIP --output-format=cmd
pcs resource create --no-default-ops --force -- VirtualIP ocf:heartbeat:IPaddr2 \
  cidr_netmask=24 ip=198.51.100.3 \
```

```
op \
monitor interval=10s id=VirtualIP-monitor-interval-10s timeout=20s \
start interval=0s id=VirtualIP-start-interval-0s timeout=20s \
stop interval=0s id=VirtualIP-stop-interval-0s timeout=20s
```

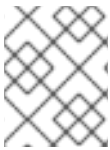
## 60.3. MODIFYING RESOURCE PARAMETERS

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.

## 60.4. 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
```

## 60.5. 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 [Disabling, enabling, 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.

### 60.5.1. 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 update 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.

## 60.5.2. 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.

The following table describes the properties you can set for a **ping** resource.

**Table 60.1. 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
```

## 60.6. 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 op defaults** 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
```

## 60.7. CONFIGURING AND MANAGING CLUSTER RESOURCE TAGS

As of Red Hat Enterprise Linux 8.3, you can use the **pcs** command to tag cluster resources. This allows you to enable, disable, manage, or unmanage a specified set of resources with a single command.

### 60.7.1. Tagging cluster resources for administration by category

The following procedure tags two resources with a resource tag and disables the tagged resources. In this example, the existing resources to be tagged are named **d-01** and **d-02**.

#### Procedure

1. Create a tag named **special-resources** for resources **d-01** and **d-02**.

```
[root@node-01]# pcs tag create special-resources d-01 d-02
```

2. Display the resource tag configuration.

```
[root@node-01]# pcs tag config
special-resources
  d-01
  d-02
```

3. Disable all resources that are tagged with the **special-resources** tag.

```
[root@node-01]# pcs resource disable special-resources
```

4. Display the status of the resources to confirm that resources **d-01** and **d-02** are disabled.

```
[root@node-01]# pcs resource
* d-01      (ocf::pacemaker:Dummy): Stopped (disabled)
* d-02      (ocf::pacemaker:Dummy): Stopped (disabled)
```

In addition to the **pcs resource disable** command, the **pcs resource enable**, **pcs resource manage**, and **pcs resource unmanage** commands support the administration of tagged resources.

After you have created a resource tag:

- You can delete a resource tag with the **pcs tag delete** command.
- You can modify resource tag configuration for an existing resource tag with the **pcs tag update** command.

## 60.7.2. Deleting a tagged cluster resource

You cannot delete a tagged cluster resource with the **pcs** command. To delete a tagged resource, use the following procedure.

### Procedure

1. Remove the resource tag.

- a. The following command removes the resource tag **special-resources** from all resources with that tag,

```
[root@node-01]# pcs tag remove special-resources
[root@node-01]# pcs tag
No tags defined
```

- b. The following command removes the resource tag **special-resources** from the resource **d-01** only.

```
[root@node-01]# pcs tag update special-resources remove d-01
```

2. Delete the resource.

```
[root@node-01]# pcs resource delete d-01
Attempting to stop: d-01... Stopped
```

## CHAPTER 61. 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.

### 61.1. CREATING AND REMOVING A CLONED RESOURCE

You can create a resource and a clone of that resource at the same time.

To create a resource and clone of the resource with the following single command.

RHEL 8.4 and later:

```
pcs resource create resource_id [standard:[provider:]]type [resource options] [meta resource meta options] clone [clone_id] [clone options]
```

RHEL 8.3 and earlier:

```
pcs resource create resource_id [standard:[provider:]]type [resource options] [meta resource meta options] clone [clone options]
```

By default, the name of the clone will be **resource\_id-clone**. As of RHEL 8.4, you can set a custom name for the clone by specifying a value for the *clone\_id* option.

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.

RHEL 8.4 and later:

```
pcs resource clone resource_id | group_id [clone_id][clone options]...
```

RHEL 8.3 and earlier:

```
pcs resource clone resource_id | group_id [clone options]...
```

By default, the name of the clone will be **resource\_id-clone** or **group\_name-clone**. As of RHEL 8.4, you can set a custom name for the clone by specifying a value for the *clone\_id* option.

**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, by default the clone takes on the name of the resource with **-clone** appended to the name. The following command 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 | clone_id | group_name
```

The following table describes the options you can specify for a cloned resource.

**Table 61.1. Resource Clone Options**

Field	Description
<b>priority, target-role, is-managed</b>	Options inherited from resource that is being cloned, as described in the "Resource Meta Options" table in <a href="#">Configuring 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, true</b> . The default value is <b>false</b> .

Field	Description
<b>globally-unique</b>	<p>Does each copy of the clone perform a different function? Allowed values: <b>false, true</b></p> <p>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.</p> <p>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>.</p>
<b>ordered</b>	<p>Should the copies be started in series (instead of in parallel). Allowed values: <b>false, true</b>. The default value is <b>false</b>.</p>
<b>interleave</b>	<p>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, true</b>. The default value is <b>false</b>.</p>
<b>clone-min</b>	<p>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>.</p>

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-stickness** 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-stickness** resource meta-option, see [Configuring resource meta options](#).

## 61.2. 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

## 61.3. PROMOTABLE CLONE RESOURCES

Promotable clone resources are clone resources with the **promotable** 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.

### 61.3.1. Creating a promotable clone resource

You can create a resource as a promotable clone with the following single command.

RHEL 8.4 and later:

```
pcs resource create resource_id [standard:[provider:]]type [resource options] promotable [clone_id]
[clone options]
```

RHEL 8.3 and earlier:

```
pcs resource create resource_id [standard:[provider:]]type [resource options] promotable [clone
options]
```

By default, the name of the promotable clone will be **resource\_id-clone**.

As of RHEL 8.4, you can set a custom name for the clone by specifying a value for the *clone\_id* option.

Alternately, you can create a promotable resource from a previously-created resource or resource group with the following command.

RHEL 8.4 and later:

```
pcs resource promotable resource_id [clone_id] [clone options]
```



RHEL 8.3 and earlier:

```
pcs resource promotable resource_id [clone options]
```

By default, the name of the promotable clone will be ***resource\_id-clone*** or ***group\_name-clone***.

As of RHEL 8.4, you can set a custom name for the clone by specifying a value for the *clone\_id* option.

The following table describes the extra clone options you can specify for a promotable resource.

**Table 61.2. Extra Clone Options Available for Promotable 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.

### 61.3.2. Configuring promotable resource constraints

In most cases, a promotable resource 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]
```

For information on resource order constraints, see [Determining the order in which cluster resources are run](#).

## 61.4. DEMOTING A PROMOTED RESOURCE ON FAILURE

As of RHEL 8.3, you can configure a promotable resource so that when a **promote** or **monitor** action fails for that resource, or the partition in which the resource is running loses quorum, the resource will be demoted but will not be fully stopped. This can prevent the need for manual intervention in situations where fully stopping the resource would require it.

- To configure a promotable resource to be demoted when a **promote** action fails, set the **on-fail** operation meta option to **demote**, as in the following example.

```
# pcs resource op add my-rsc promote on-fail="demote"
```

- To configure a promotable resource to be demoted when a **monitor** action fails, set **interval** to a nonzero value, set the **on-fail** operation meta option to **demote**, and set **role** to **Master**, as in the following example.

```
# pcs resource op add my-rsc monitor interval="10s" on-fail="demote" role="Master"
```

- To configure a cluster so that when a cluster partition loses quorum any promoted resources will be demoted but left running and all other resources will be stopped, set the **no-quorum-policy** cluster property to **demote**

Setting the **on-fail** meta-attribute to **demote** for an operation does not affect how promotion of a resource is determined. If the affected node still has the highest promotion score, it will be selected to be promoted again.

## CHAPTER 62. MANAGING CLUSTER NODES

There are a variety of **pcs** commands you can use to manage cluster nodes, including commands to start and stop cluster services and to add and remove cluster nodes.

### 62.1. 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
```

### 62.2. ENABLING AND DISABLING CLUSTER SERVICES

Enable the cluster services with the following command. This 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.

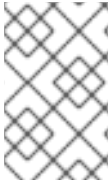
- If you specify the **--all** option, the command disables cluster services on all nodes.
- If you do not specify any nodes, cluster services are disabled on the local node only.

```
pcs cluster disable [--all | node] [...]
```

### 62.3. ADDING CLUSTER NODES

Add a new node to an existing cluster with the following procedure.

This procedure adds standard cluster 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](#).



## NOTE

It is 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.

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**.

## Procedure

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 (**sbdd**, **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.

## 62.4. 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
```

## 62.5. 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
```

## 62.6. ADDING AND MODIFYING LINKS IN AN EXISTING CLUSTER

As of RHEL 8.1, in most cases, you can add or modify the links in an existing cluster without restarting the cluster.

### 62.6.1. 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 or (for RHEL 8.4 and later) the **pcs cluster config show** 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
```

### 62.6.2. 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.

#### 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
```

### 62.6.3. 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.

## Procedure

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
```

### 62.6.4. 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.

## Procedure

Modify the link option in a cluster with a single link with 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 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
```

### 62.6.5. 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.

#### Procedure

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
```

## 62.7. CONFIGURING A NODE HEALTH STRATEGY

A node might be functioning well enough to maintain its cluster membership and yet be unhealthy in some respect that makes it an undesirable location for resources. For example, a disk drive might be reporting SMART errors, or the CPU might be highly loaded. As of RHEL 8.7, You can use a node health strategy in Pacemaker to automatically move resources off unhealthy nodes.

You can monitor a node's health with the the following health node resource agents, which set node attributes based on CPU and disk status:

- **ocf:pacemaker:HealthCPU**, which monitors CPU idling
- **ocf:pacemaker:HealthIOWait**, which monitors the CPU I/O wait
- **ocf:pacemaker:HealthSMART**, which monitors SMART status of a disk drive



- **ocf:pacemaker:SysInfo**, which sets a variety of node attributes with local system information and also functions as a health agent monitoring disk space usage

Additionally, any resource agent might provide node attributes that can be used to define a health node strategy.

## Procedure

The following procedure configures a health node strategy for a cluster that will move resources off of any node whose CPU I/O wait goes above 15%.

1. Set the **health-node-strategy** cluster property to define how Pacemaker responds to changes in node health.

```
# pcs property set node-health-strategy=migrate-on-red
```

2. Create a cloned cluster resource that uses a health node resource agent, setting the **allow-unhealthy-nodes** resource meta option to define whether the cluster will detect if the node's health recovers and move resources back to the node. Configure this resource with a recurring monitor action, to continually check the health of all nodes.

This example creates a **HealthIOWait** resource agent to monitor the CPU I/O wait, setting a red limit for moving resources off a node to 15%. This command sets the **allow-unhealthy-nodes** resource meta option to **true** and configures a recurring monitor interval of 10 seconds.

```
# pcs resource create io-monitor ocf:pacemaker:HealthIOWait red_limit=15 op monitor interval=10s meta allow-unhealthy-nodes=true clone
```

## 62.8. CONFIGURING A LARGE CLUSTER WITH MANY RESOURCES

If the cluster you are deploying consists of a large number of nodes and many resources, you may need to modify the default values of the following parameters for your cluster.

### The **cluster-ipc-limit** cluster property

The **cluster-ipc-limit** cluster property is the maximum IPC message backlog before one cluster daemon will disconnect another. When a large number of resources are cleaned up or otherwise modified simultaneously in a large cluster, a large number of CIB updates arrive at once. This could cause slower clients to be evicted if the Pacemaker service does not have time to process all of the configuration updates before the CIB event queue threshold is reached.

The recommended value of **cluster-ipc-limit** for use in large clusters is the number of resources in the cluster multiplied by the number of nodes. This value can be raised if you see "Evicting client" messages for cluster daemon PIDs in the logs.

You can increase the value of **cluster-ipc-limit** from its default value of 500 with the **pcs property set** command. For example, for a ten-node cluster with 200 resources you can set the value of **cluster-ipc-limit** to 2000 with the following command.

```
# pcs property set cluster-ipc-limit=2000
```

### The **PCMK\_ipc\_buffer** Pacemaker parameter

On very large deployments, internal Pacemaker messages may exceed the size of the message buffer. When this occurs, you will see a message in the system logs of the following format:

Compressed message exceeds  $X\%$  of configured IPC limit ( $X$  bytes); consider setting `PCMK_ipc_buffer` to  $X$  or higher

When you see this message, you can increase the value of `PCMK_ipc_buffer` in the `/etc/sysconfig/pacemaker` configuration file on each node. For example, to increase the value of `PCMK_ipc_buffer` from its default value to 13396332 bytes, change the uncommented `PCMK_ipc_buffer` field in the `/etc/sysconfig/pacemaker` file on each node in the cluster as follows.

```
PCMK_ipc_buffer=13396332
```

To apply this change, run the following command.

```
# systemctl restart pacemaker
```

## CHAPTER 63. PACEMAKER CLUSTER PROPERTIES

Cluster properties control how the cluster behaves when confronted with situations that may occur during cluster operation.

### 63.1. SUMMARY OF CLUSTER PROPERTIES AND OPTIONS

The following table summarizes 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 the table of cluster properties that determine fencing behavior in [General properties of fencing devices](#).



#### 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 63.1. 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.

Option	Default	Description
<b>no-quorum-policy</b>	stop	<p>What to do when the cluster does not have quorum. Allowed values:</p> <ul style="list-style-type: none"> <li>* ignore - continue all resource management</li> <li>* freeze - continue resource management, but do not recover resources from nodes not in the affected partition</li> <li>* stop - stop all resources in the affected cluster partition</li> <li>* suicide - fence all nodes in the affected cluster partition</li> <li>* demote - if a cluster partition loses quorum, demote any promoted resources and stop all other resources</li> </ul>
<b>symmetric-cluster</b>	true	Indicates whether resources can run on any node by default.
<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>dc-deadtime</b>	20s	How long to wait for a response from other nodes during startup. The "correct" value will depend on the speed and load of your network and the type of switches used.
<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.

Option	Default	Description
<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.
<b>dc-version</b>		Version of Pacemaker on the cluster's Designated Controller (DC). Used for diagnostic purposes; not user-configurable.

Option	Default	Description
<b>cluster-recheck-interval</b>	15 minutes	Pacemaker is primarily event-driven, and looks ahead to know when to recheck the cluster for failure timeouts and most time-based rules. Pacemaker will also recheck the cluster after the duration of inactivity specified by this property. This cluster recheck has two purposes: rules with <b>date-spec</b> are guaranteed to be checked this often, and it serves as a fail-safe for some kinds of scheduler bugs. A value of 0 disables this polling; positive values indicate a time interval.
<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>	default	Indicates whether and how the cluster will take utilization attributes into account when determining resource placement on cluster nodes.

Option	Default	Description
<b>priority-fencing-delay</b>	0 (disabled)	<p>(RHEL 8.3 and later) Allows you to configure a two-node cluster so that in a split-brain situation the node with the fewest resources running is the node that gets fenced.</p> <p>The <b>priority-fencing-delay</b> property can be set to a time duration. The default value for this property is 0 (disabled). If this property is set to a non-zero value, and the <b>priority</b> meta-attribute is configured for at least one resource, then in a split-brain situation the node with the highest combined priority of all resources running on it will be more likely to survive.</p> <p>For example, if you set <b>pcs resource defaults priority=1</b> and <b>pcs property set priority-fencing-delay=15s</b> and no other priorities are set, then the node running the most resources will be more likely to survive because the other node will wait 15 seconds before initiating fencing. If a particular resource is more important than the rest, you can give it a higher priority.</p> <p>The node running the master role of a promotable clone gets an extra 1 point if a priority has been configured for that clone.</p> <p>Any delay set with the <b>priority-fencing-delay</b> property will be added to any delay from the <b>pcmk_delay_base</b> and <b>pcmk_delay_max</b> fence device properties. This behavior allows some delay when both nodes have equal priority, or both nodes need to be fenced for some reason other than node loss (for example, if <b>on-fail=fencing</b> is set for a resource monitor operation). If used in combination, it is recommended that you set the <b>priority-fencing-delay</b> property to a value that is significantly greater than the maximum delay from <b>pcmk_delay_base</b> and <b>pcmk_delay_max</b>, to be sure the prioritized node is preferred (twice the value would be completely safe).</p> <p>Only fencing scheduled by Pacemaker itself will observe <b>priority-fencing-delay</b>. Fencing scheduled by external code such as <b>dlm_controld</b> will not provide the necessary information to the fence device.</p>

Option	Default	Description
<b>node-health-strategy</b>	none	<p>When used in conjunction with a health resource agent, controls how Pacemaker responds to changes in node health. Allowed values:</p> <ul style="list-style-type: none"> <li>* <b>none</b> - Do not track node health.</li> <li>* <b>migrate-on-red</b> - Resources are moved off any node where a health agent has determined that the node's status is <b>red</b>, based on the local conditions that the agent monitors.</li> <li>* <b>only-green</b> - Resources are moved off any node where a health agent has determined that the node's status is <b>yellow</b> or <b>red</b>, based on the local conditions that the agent monitors.</li> <li>* <b>progressive, custom</b> - Advanced node health strategies that offer finer-grained control over the cluster's response to health conditions according to the internal numeric values of health attributes.</li> </ul>

## 63.2. 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=
```



### 63.3. 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 64. 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.

### 64.1. VIRTUAL DOMAIN RESOURCE OPTIONS

The following table describes the resource options you can configure for a **VirtualDomain** resource.

**Table 64.1. 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 it true, 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.

## 64.2. CREATING THE VIRTUAL DOMAIN RESOURCE

The following procedure creates a **VirtualDomain** resource in a cluster for a virtual machine you have previously created.

### Procedure

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 resource 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 65. CONFIGURING 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.

### 65.1. 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. The following table summarizes these options.

Table 65.1. 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.//LINK TO</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.

## 65.2. MODIFYING QUORUM OPTIONS

You can modify general quorum options for your cluster with the **pcs quorum update** command. You can modify the **quorum.two\_node** and **quorum.expected\_votes** options on a running system. For all other quorum options, 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
```

## 65.3. DISPLAYING QUORUM CONFIGURATION AND STATUS

Once a cluster is running, you can enter the following cluster quorum commands to display the quorum configuration and status.

The following command shows the quorum configuration.

```
pcs quorum [config]
```

The following command shows the quorum runtime status.

```
pcs quorum status
```

## 65.4. 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 unblock** 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 unblock
```



## CHAPTER 66. 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.

## 66.1. 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.

## 66.2. 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.

### 66.2.1. 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 the "Resource Options for Virtual Domain Resources" table in [Virtual domain resource options](#).

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. The following table describes these metadata options.

Table 66.1. 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. <i>WARNING:</i> This value cannot overlap with any resource or node IDs.
<b>remote-port</b>	3121	Configures a custom port to use for the guest connection to <b>pacemaker_remote</b>

Field	Default	Description
<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

## 66.2.2. 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.

### Procedure

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 pcsd.service
# systemctl enable pcsd.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.
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
```

## 66.3. 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.

### 66.3.1. Remote node resource options

The following table describes the resource options you can configure for a **remote** resource.

Table 66.2. 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.

### 66.3.2. Remote node configuration overview

The following procedure 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.

#### Procedure

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.

## 66.4. 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).

## 66.5. 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**.

### Procedure

1. Stop the node's connection resource with the **pcs resource disable *resourcename*** command, which will move all services off the node. The connection resource would be the **ocf:pacemaker:remote** resource for a remote node or, commonly, the **ocf:heartbeat:VirtualDomain** resource for a guest node. For guest nodes, this command will also stop the VM, so the VM must be started outside the cluster (for example, using **virsh**) to perform any maintenance.

```
pcs resource disable resourcename
```

2. Perform the required maintenance.
3. When ready to return the node to the cluster, re-enable the resource with the **pcs resource enable** command.

```
pcs resource enable resourcename
```

## CHAPTER 67. 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.  
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.
- 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 [Disabling, enabling, 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 RHEL 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](#).
- If you need to migrate a VM in a RHEL cluster, you will first need to stop the cluster services on the VM to remove the node from the cluster and then start the cluster back up after performing the migration. as described in [Migrating VMs in a RHEL cluster](#).

### 67.1. 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.

## 67.2. 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 need 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.

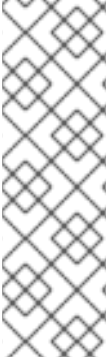
- You can use the **pcs resource move** command to move a resource off a node on which it is currently running.
- You can use the **pcs resource relocate run** command to move a resource to its preferred node, as determined by current cluster status, constraints, location of resources and other settings.

### 67.2.1. 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 run 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. As of RHEL 8.6, you can specify the **--autodelete** option for this command, which will cause the location constraint that this command creates to be removed automatically once the resource has been moved. For earlier releases, you can run the **pcs resource clear** or the **pcs constraint delete** command to remove the constraint manually. Removing the constraint 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 constraint applies only to promoted instances of the resource.

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 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
```

### 67.2.2. 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.

## 67.3. 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 RHEL 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

As of RHEL 8.5 you can specify the **--brief** option for the **pcs resource disable --safe** command to print errors only. Also as of RHEL 8.5, the error report that the **pcs resource disable --safe** command generates if the safe disable operation fails contains the affected resource IDs. If you need to know only the resource IDs of resources that would be affected by disabling a resource, use the **--brief** option, which does not provide the full simulation result.

### 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
```

## 67.4. 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.

## 67.5. 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=
```

## 67.6. 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](#).

## 67.7. 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**.

### Procedure

1. Stop the node's connection resource with the **pcs resource disable *resourcename*** command, which will move all services off the node. The connection resource would be the **ocf:pacemaker:remote** resource for a remote node or, commonly, the **ocf:heartbeat:VirtualDomain** resource for a guest node. For guest nodes, this command will also stop the VM, so the VM must be started outside the cluster (for example, using **virsh**) to perform any maintenance.

```
pcs resource disable resourcename
```

2. Perform the required maintenance.
3. When ready to return the node to the cluster, re-enable the resource with the **pcs resource enable** command.

```
pcs resource enable resourcename
```

## 67.8. MIGRATING VMS IN A RHEL CLUSTER

Red Hat does not support live migration of active cluster nodes across hypervisors or hosts, as noted in [Support Policies for RHEL High Availability Clusters - General Conditions with Virtualized Cluster Members](#). If you need to perform a live migration, you will first need to stop the cluster services on the VM to remove the node from the cluster, and then start the cluster back up after performing the migration. The following steps outline the procedure for removing a VM from a cluster, migrating the VM, and restoring the VM to the cluster.

The following steps outline the procedure for removing a VM from a cluster, migrating the VM, and restoring the VM to the cluster.

This procedure applies to VMs that are used as full cluster nodes, not to VMs managed as cluster resources (including VMs used as guest nodes) which can be live-migrated without special precautions. For general information on the fuller procedure required for updating packages that make up the RHEL High Availability and Resilient Storage Add-Ons, either individually or as a whole, see [Recommended Practices for Applying Software Updates to a RHEL High Availability or Resilient Storage Cluster](#).



### NOTE

Before performing this procedure, consider the effect on cluster quorum of removing a cluster node. For example, if you have a three-node cluster and you remove one node, your cluster can withstand only one more node failure. If one node of a three-node cluster is already down, removing a second node will lose quorum.

### Procedure

1. If any preparations need to be made before stopping or moving the resources or software running on the VM to migrate, perform those steps.
2. Run the following command on the VM to stop the cluster software on the VM.

```
# pcs cluster stop
```

3. Perform the live migration of the VM.
4. Start cluster services on the VM.

```
# pcs cluster start
```

## 67.9. IDENTIFYING CLUSTERS BY UUID

As of Red Hat Enterprise Linux 8.7, when you create a cluster it has an associated UUID. Since a cluster name is not a unique cluster identifier, a third-party tool such as a configuration management database that manages multiple clusters with the same name can uniquely identify a cluster by means of its UUID. You can display the current cluster UUID with the **pcs cluster config [show]** command, which includes the cluster UUID in its output.

To add a UUID to an existing cluster, run the following command.

```
# pcs cluster config uuid generate
```

To regenerate a UUID for a cluster with an existing UUID, run the following command.

```
# pcs cluster config uuid generate --force
```

# CHAPTER 68. CONFIGURING AND MANAGING LOGICAL VOLUMES

## 68.1. OVERVIEW OF LOGICAL VOLUME MANAGEMENT

Logical volume management (LVM) creates a layer of abstraction over physical storage, which helps 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.

### 68.1.1. LVM architecture

The following are the components of LVM:

#### Physical volume

A physical volume (PV) is a partition or whole disk designated for LVM use. For more information, see [Managing LVM physical volumes](#).

#### Volume group

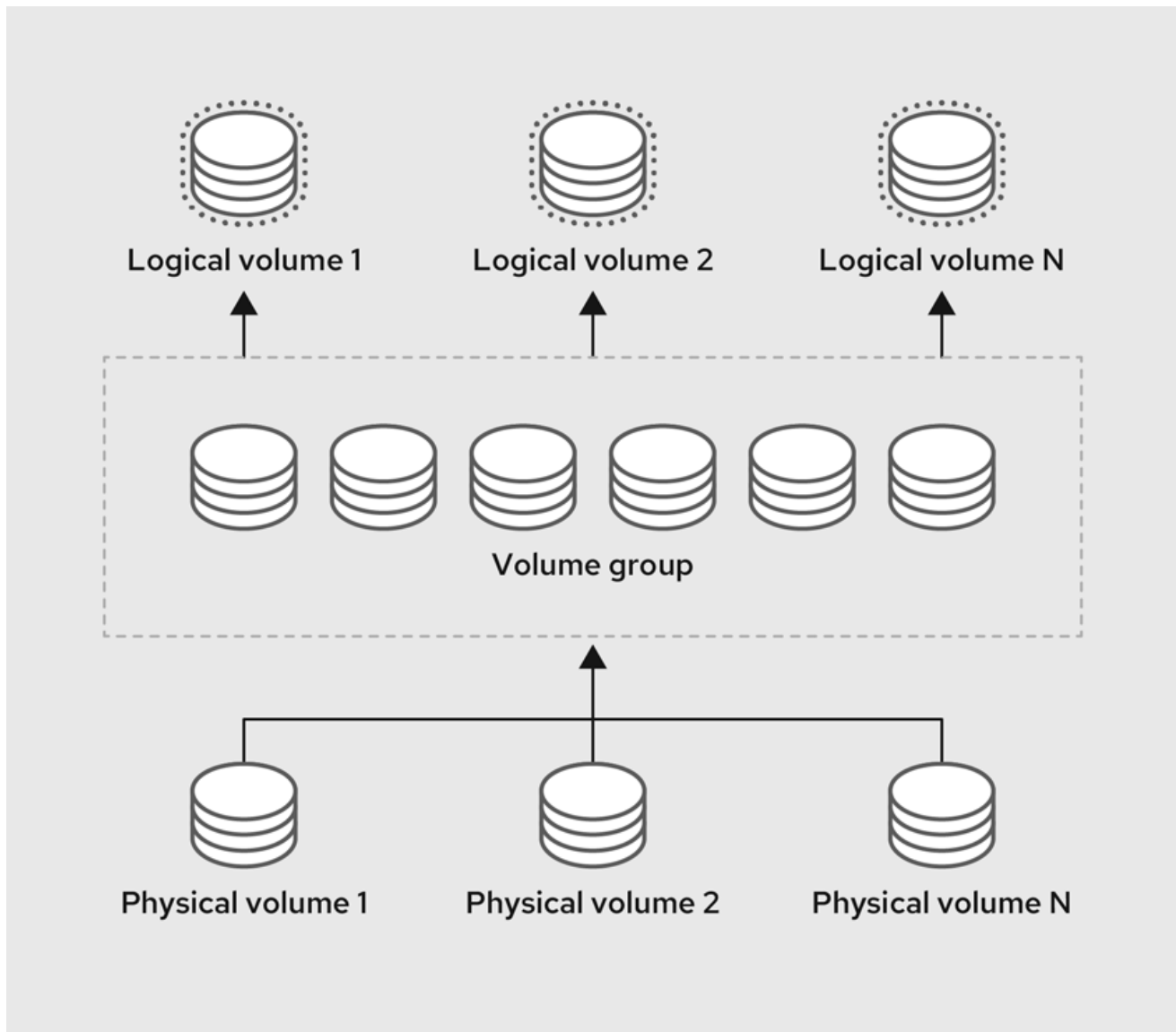
A volume group (VG) is a collection of physical volumes (PVs), which creates a pool of disk space out of which logical volumes can be allocated. For more information, see [Managing LVM volume groups](#).

#### Logical volume

A logical volume represents a mountable storage device. For more information, see [Managing LVM logical volumes](#).

The following diagram illustrates the components of LVM:

Figure 68.1. LVM logical volume components



### 68.1.2. Advantages of LVM

Logical volumes provide the following advantages over using physical storage directly:

#### Flexible capacity

When using logical volumes, you can aggregate devices and partitions into a single logical volume. With this functionality, file systems can extend across multiple devices as though they were a single, large one.

#### Resizable storage volumes

You can extend logical volumes or reduce logical volumes in size with simple software commands, without reformatting and repartitioning the underlying 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 with user-defined and custom names.

#### Striped Volumes



You can create a logical volume that stripes data across two or more devices. This can dramatically increase throughput.

### RAID volumes

Logical volumes provide a convenient way to configure RAID for your data. This provides protection against device failure and improves performance.

### Volume snapshots

You can take snapshots, which is a point-in-time copy of logical volumes 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 physical space.

### Cache volumes

A cache logical volume uses a fast block device, such as an SSD drive to improve the performance of a larger and slower block device.

## 68.2. MANAGING LVM PHYSICAL VOLUMES

The physical volume (PV) is a partition or whole disk designated for LVM use. To use the device for an LVM logical volume, the device must be initialized as 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 **cdisk** command or an equivalent. If you are using a whole disk device for your physical volume, the disk must have no partition table. Any existing partition table must be erased, which will effectively destroy all data on that disk. You can remove an existing partition table using the **wipefs -a <PhysicalVolume>** command as root.

### 68.2.1. Overview of physical volumes

Initializing a block device as a physical volume places a label near the start of the device. The following describes the LVM label:

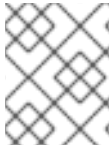
- An LVM label provides correct identification and device ordering for a physical device. An unlabeled, non-LVM device can change names across reboots depending on the order they are discovered by the system during boot. An LVM label remains persistent across reboots and throughout a cluster.
- The LVM 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.
- By default, the LVM label is placed in the second 512-byte sector. You can overwrite this default setting 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.

The following describes the LVM metadata:

- 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.

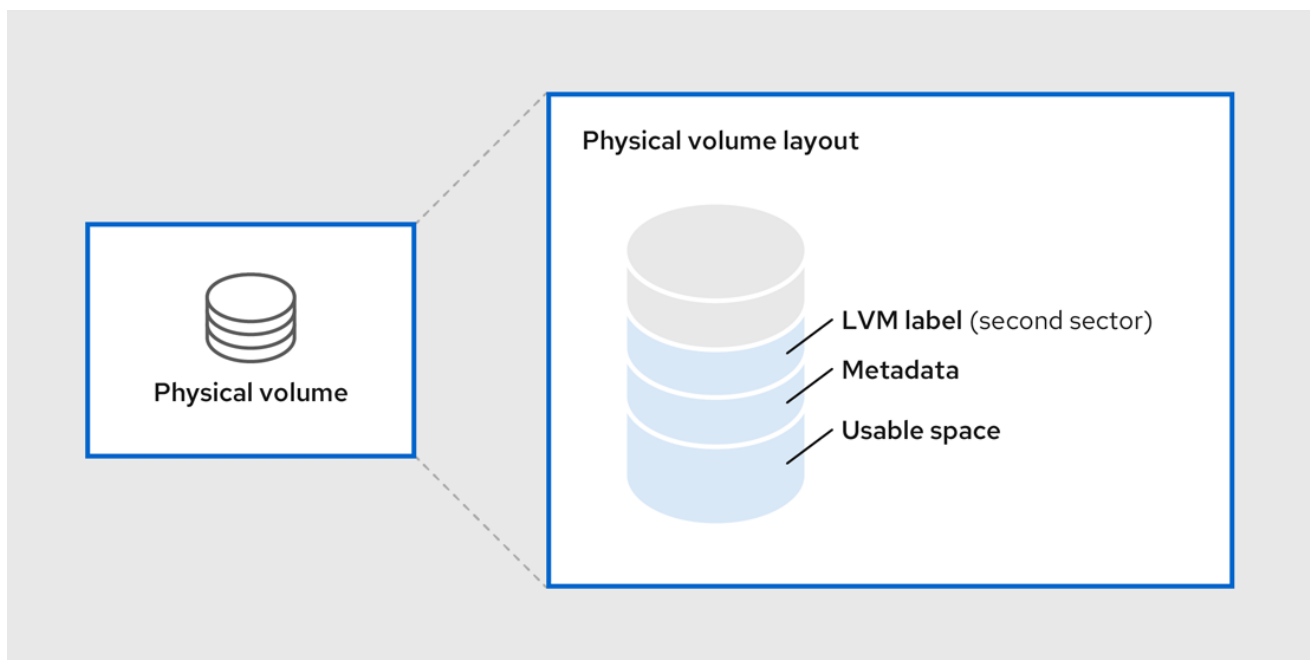
The following diagram illustrates the layout of an LVM physical volume. The LVM label is on the second sector, followed by the metadata area, followed by the usable space on the device.



#### NOTE

In the Linux kernel and throughout this document, sectors are considered to be 512 bytes in size.

Figure 68.2. Physical volume layout



#### Additional resources

- [Multiple partitions on a disk](#)

### 68.2.2. Multiple partitions on a disk

You can create physical volumes (PV) out of disk partitions by using LVM.

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.

#### 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.

#### RAID redundancy

LVM cannot determine that the two physical volumes are on the same device. If you create a RAID logical volume when two physical volumes are on the same device, performance and fault tolerance could be lost.

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 volumes.

Note that although LVM supports using a non-partitioned disk as physical volume, it is recommended to create a single, whole-disk partition because creating a PV without a partition can be problematic in a mixed operating system environment. Other operating systems may interpret the device as free, and overwrite the PV label at the beginning of the drive.

### 68.2.3. Creating LVM physical volume

This procedure describes how to create and label LVM physical volumes (PVs).

In this procedure, replace the `/dev/vdb1`, `/dev/vdb2`, and `/dev/vdb3` with the available storage devices in your system.

#### Prerequisites

- The **lvm2** package is installed.

#### Procedure

1. Create multiple physical volumes by using the space-delimited device names as arguments to the **pvcreate** command:

```
# pvcreate /dev/vdb1 /dev/vdb2 /dev/vdb3
Physical volume "/dev/vdb1" successfully created.
Physical volume "/dev/vdb2" successfully created.
Physical volume "/dev/vdb3" successfully created.
```

This places a label on `/dev/vdb1`, `/dev/vdb2`, and `/dev/vdb3`, marking them as physical volumes belonging to LVM.

2. View the created physical volumes by using any one of the following commands as per your requirement:
  - a. The **pvdisplay** command, which provides a verbose multi-line output for each physical volume. It displays physical properties, such as size, extents, volume group, and other options in a fixed format:

```
# pvdisplay
--- NEW Physical volume ---
PV Name          /dev/vdb1
VG Name
PV Size          1.00 GiB
[..]
--- NEW Physical volume ---
```

```

PV Name      /dev/vdb2
VG Name
PV Size      1.00 GiB
[.]
--- NEW Physical volume ---
PV Name      /dev/vdb3
VG Name
PV Size      1.00 GiB
[.]

```

- b. The **pvs** command provides physical volume information in a configurable form, displaying one line per physical volume:

```

# pvs
PV      VG Fmt  Attr  PSize  PFree
/dev/vdb1  lvm2  1020.00m  0
/dev/vdb2  lvm2  1020.00m  0
/dev/vdb3  lvm2  1020.00m  0

```

- c. The **pvscan** command scans all supported LVM block devices in the system for physical volumes. You can define a filter in the **lvm.conf** file so that this command avoids scanning specific physical volumes:

```

# pvscan
PV /dev/vdb1      lvm2 [1.00 GiB]
PV /dev/vdb2      lvm2 [1.00 GiB]
PV /dev/vdb3      lvm2 [1.00 GiB]

```

#### Additional resources

- **pvcreate(8)**, **pvdisplay(8)**, **pvs(8)**, **pvscan(8)**, and **lvm(8)** man pages

### 68.2.4. Removing LVM physical volumes

If a device is no longer required for use by LVM, you can remove the LVM label by using the **pvremove** command. Executing the **pvremove** command zeroes the LVM metadata on an empty physical volume.

#### Procedure

1. Remove a physical volume:

```

# pvremove /dev/vdb3
Labels on physical volume "/dev/vdb3" successfully wiped.

```

2. View the existing physical volumes and verify if the required volume is removed:

```

# pvs
PV      VG Fmt  Attr  PSize  PFree
/dev/vdb1  lvm2  1020.00m  0
/dev/vdb2  lvm2  1020.00m  0

```

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. For more information, see [Removing physical volumes from a volume group](#)

### Additional resources

- **pvremove(8)** man page

### 68.2.5. Additional resources

- [Creating a partition table on a disk with parted](#) .
- **parted(8)** man page.

## 68.3. MANAGING LVM VOLUME GROUPS

A volume group (VG) is a collection of physical volumes (PVs), which creates a pool of disk space out of which logical volumes (LVs) 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 therefore the same for all logical volumes in the volume group. The volume group maps the logical extents to physical extents.

### 68.3.1. Creating LVM volume group

This procedure describes how to create an LVM volume group (VG) *myvg*, by using the */dev/vdb1* and */dev/vdb2* physical volumes.

#### Prerequisites

- The **lvm2** package is installed.
- One or more physical volumes are created. For more information about creating physical volumes, see [Creating LVM physical volume](#) .

#### Procedure

1. Create a volume group:

```
# vgcreate myvg /dev/vdb1 /dev/vdb2
Volume group "myvg" successfully created.
```

This creates a VG with the name of *myvg*. The PVs */dev/vdb1* and */dev/vdb2* are the base storage level for the *myvg* VG .

2. View the created volume groups by using any one of the following commands according to your requirement:
  - a. The **vgs** command provides volume group information in a configurable form, displaying one line per volume groups:

```
# vgs
VG #PV #LV #SN Attr VSize VFree
myvg 2 0 0 wz-n 159.99g 159.99g
```

- b. The **vgdisplay** command displays volume group properties such as size, extents, number of physical volumes, and other options in a fixed form. The following example shows the output of the **vgdisplay** command for the volume group *myvg*. To display all existing volume groups, do not specify a volume group:

```
# vgdisplay myvg
--- Volume group ---
VG Name          myvg
System ID
Format           lvm2
Metadata Areas   4
Metadata Sequence No 6
VG Access        read/write
[.]
```

- c. The **vgscan** command scans all supported LVM block devices in the system for volume group:

```
# vgscan
Found volume group "myvg" using metadata type lvm2
```

3. Optional: Increase a volume group's capacity by adding one or more free physical volumes:

```
# vgextend myvg /dev/vdb3
Physical volume "/dev/vdb3" successfully created.
Volume group "myvg" successfully extended
```

4. Optional: Rename an existing volume group:

```
# vgrename myvg myvg1
Volume group "myvg" successfully renamed to "myvg1"
```

### Additional resources

- **vgcreate(8)**, **vgextend(8)**, **vgdisplay(8)**, **vgs(8)**, **vgscan(8)**, **vgrename(8)**, and **lvm(8)** man pages

### 68.3.2. Combining LVM volume groups

To combine two volume groups into a single volume group, use the **vgmerge** command. You can merge an inactive "source" volume with an active or an inactive "destination" volume if the physical extent sizes of the volume are equal and the physical and logical volume summaries of both volume groups fit into the destination volume groups limits.

#### Procedure

- Merge the inactive volume group *databases* into the active or inactive volume group *myvg* giving verbose runtime information:

```
# vgmerge -v myvg databases
```

### Additional resources

- **vgmerge(8)** man page

### 68.3.3. 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.

#### Procedure

1. If the physical volume is still being used, migrate the data to another physical volume from the same volume group :

```
# pvmove /dev/vdb3
/dev/vdb3: Moved: 2.0%
...
/dev/vdb3: Moved: 79.2%
...
/dev/vdb3: Moved: 100.0%
```

2. If there are no enough free extents on the other physical volumes in the existing volume group:
  - a. Create a new physical volume from `/dev/vdb4`:

```
# pvcreate /dev/vdb4
Physical volume "/dev/vdb4" successfully created
```

- b. Add the newly created physical volume to the `myvg` volume group:

```
# vgextend myvg /dev/vdb4
Volume group "myvg" successfully extended
```

- c. Move the data from `/dev/vdb3` to `/dev/vdb4`:

```
# pvmove /dev/vdb3 /dev/vdb4
/dev/vdb3: Moved: 33.33%
/dev/vdb3: Moved: 100.00%
```

3. Remove the physical volume `/dev/vdb3` from the volume group:

```
# vgreduce myvg /dev/vdb3
Removed "/dev/vdb3" from volume group "myvg"
```

#### Verification

- Verify if the `/dev/vdb3` physical volume is removed from the `myvg` volume group:

```
# pvs
```

```

PV          VG  Fmt Attr PSize   PFree   Used
/dev/vdb1  myvg lvm2 a-- 1020.00m 0      1020.00m
/dev/vdb2  myvg lvm2 a-- 1020.00m 0      1020.00m
/dev/vdb3          lvm2 a-- 1020.00m 1008.00m 12.00m

```

### Additional resources

- **vgreduce(8)**, **pvmove(8)**, and **pvs(8)** man pages

## 68.3.4. Splitting a LVM volume group

This procedure describes how to split the existing volume group. If there is enough unused space on the physical volumes, a new volume group can be created without adding new disks.

In the initial setup, the volume group *myvg* consists of */dev/vdb1*, */dev/vdb2*, and */dev/vdb3*. After completing this procedure, the volume group *myvg* will consist of */dev/vdb1* and */dev/vdb2*, and the second volume group, *yourvg*, will consist of */dev/vdb3*.

### Prerequisites

- You have sufficient space in the volume group. Use the **vgscan** command to determine how much free space is currently available in the volume group.
- Depending on the free capacity in the existing physical volume, move all the used physical extents to other physical volume using the **pvmove** command. For more information, see [Removing physical volumes from a volume group](#).

### Procedure

1. Split the existing volume group *myvg* to the new volume group *yourvg*:

```

# vgsplit myvg yourvg /dev/vdb3
Volume group "yourvg" successfully split from "myvg"

```



#### NOTE

If you have created a logical volume using the existing volume group, use the following command to deactivate the logical volume:

```
# lvchange -a n /dev/myvg/mylv
```

For more information on creating logical volumes, see [Managing LVM logical volumes](#).

2. View the attributes of the two volume group:

```

# vgs
VG   #PV #LV #SN Attr   VSize VFree
myvg 2  1  0 wz--n- 34.30G 10.80G
yourvg 1  0  0 wz--n- 17.15G 17.15G

```

### Verification



- Verify if the newly created volume group *yourvg* consists of */dev/vdb3* physical volume:

```
# pvs
PV      VG      Fmt Attr PSize   PFree   Used
/dev/vdb1 myvg  lvm2 a--  1020.00m  0      1020.00m
/dev/vdb2 myvg  lvm2 a--  1020.00m  0      1020.00m
/dev/vdb3 yourvg lvm2 a--  1020.00m 1008.00m 12.00m
```

### Additional resources

- **vgsplit(8)**, **vg(8)**, and **pvs(8)** man pages

### 68.3.5. Moving a volume group to another system

You can move an entire LVM volume group to another system. It is recommended that you use the **vgexport** and **vgimport** commands when you do this.



#### NOTE

You can use the **--force** argument of the **vgimport** command. This allows you to import volume groups that are missing physical volumes and subsequently run the **vgreduce --removemissing** command.

The **vgexport** command makes an inactive volume group inaccessible to the system, which allows you to detach its physical volumes. The **vgimport** command makes a volume group accessible to a machine again after the **vgexport** command has made it inactive.

To move a volume group from one system to another, perform the following steps:

1. Make sure that no users are accessing files on the active volumes in the volume group, then unmount the logical volumes.
2. Use the **-a n** argument of the **vgchange** command to mark the volume group as inactive, which prevents any further activity on the volume group.
3. Use the **vgexport** command to export the volume group. This prevents it from being accessed by the system from which you are removing it.  
After you export the volume group, the physical volume will show up as being in an exported volume group when you execute the **pvscan** command, as in the following example.

```
# pvscan
PV /dev/sda1 is in exported VG myvg [17.15 GB / 7.15 GB free]
PV /dev/sdc1 is in exported VG myvg [17.15 GB / 15.15 GB free]
PV /dev/sdd1 is in exported VG myvg [17.15 GB / 15.15 GB free]
...
```

When the system is next shut down, you can unplug the disks that constitute the volume group and connect them to the new system.

4. When the disks are plugged into the new system, use the **vgimport** command to import the volume group, making it accessible to the new system.
5. Activate the volume group with the **-a y** argument of the **vgchange** command.

6. Mount the file system to make it available for use.

### 68.3.6. Removing LVM volume groups

This procedure describes how to remove an existing volume group.

#### Prerequisites

- The volume group contains no logical volumes. To remove logical volumes from a volume group, see [Removing LVM logical volumes](#).

#### Procedure

1. If the volume group exists in a clustered environment, stop the **lockspace** of the volume group on all other nodes. Use the following command on all nodes except the node where you are performing the removing:

```
# vgchange --lockstop vg-name
```

Wait for the lock to stop.

2. Remove the volume group:

```
# vgrename vg-name  
Volume group "vg-name" successfully removed
```

#### Additional resources

- [vgremove\(8\)](#) man page

## 68.4. MANAGING LVM LOGICAL VOLUMES

A logical volume is a virtual, block storage device that a file system, database, or application can use. To create an LVM logical volume, the physical volumes (PVs) are combined into a volume group (VG). This creates a pool of disk space out of which LVM logical volumes (LVs) can be allocated.

### 68.4.1. Overview of logical volumes

An administrator can grow or shrink logical volumes without destroying data, unlike standard disk partitions. If the physical volumes in a volume group are on separate drives or RAID arrays, then administrators can also spread a logical volume across the storage devices.

You can lose data if you shrink a logical volume to a smaller capacity than the data on the volume requires. Further, some file systems are not capable of shrinking. To ensure maximum flexibility, create logical volumes to meet your current needs, and leave excess storage capacity unallocated. You can safely extend logical volumes to use unallocated space, depending on your needs.



## IMPORTANT

On AMD, Intel, ARM systems, and IBM Power Systems servers, the boot loader cannot read LVM volumes. You must make a standard, non-LVM disk partition for your **/boot** partition. On IBM Z, the **zipl** boot loader supports **/boot** on LVM logical volumes with linear mapping. By default, the installation process always creates the **/** and swap partitions within LVM volumes, with a separate **/boot** partition on a physical volume.

The following are the different types of logical volumes:

### 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.

### 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.

### RAID logical volumes

LVM supports RAID levels 0, 1, 4, 5, 6, and 10. RAID logical volumes are not cluster-aware. 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.

### Thin-provisioned logical volumes (thin volumes)

Using thin-provisioned logical volumes, you can create logical volumes that are larger than the available physical storage. Creating a thinly provisioned set of volumes allows the system to allocate what you use instead of allocating the full amount of storage that is requested.

### 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.

### Thin-provisioned snapshot volumes

Using thin-provisioned snapshot volumes, you can have more virtual devices to be stored on the same data volume. Thinly provisioned snapshots are useful because you are not copying all of the data that you are looking to capture at a given time.

### 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.

## 68.4.2. 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 **lv010** in a volume group called **vg0** can be specified as **vg0/lv010**.
- 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 "lv010" 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 lv010
Creating volume group backup "/etc/lvm/backup/new_vg" (seqno 2).
Activating logical volume new_vg/lv010.
activation/volume_list configuration setting not defined: Checking only host tags for new_vg/lv010.
Creating new_vg-lv010
Loading table for new_vg-lv010 (253:0).
Resuming new_vg-lv010 (253:0).
Wiping known signatures on logical volume "new_vg/lv010"
Initializing 4.00 KiB of logical volume "new_vg/lv010" with value 0.
Logical volume "lv010" 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.

### 68.4.3. Creating LVM logical volume

This procedure describes how to create *mylv* LVM logical volume (LV) from the *myvg* volume group, which is created by using the */dev/vdb1*, */dev/vdb2*, and */dev/vdb3* physical volumes.

#### Prerequisites

- The **lvm2** package is installed.
- The volume group is created. For more information, see [Creating LVM volume group](#).

#### Procedure

1. Create a logical volume:

```
# lvcreate -n mylv -L 500M myvg
```

Use the **-n** option to set the LV name to *mylv*, and the **-L** option to set the size of LV in units of Mb, but it is possible to use any other units. The LV type is linear by default, but the user can specify the desired type by using the **--type** option.



#### IMPORTANT

The command fails if the VG does not have a sufficient number of free physical extents for the requested size and type.

2. View the created logical volumes by using any one of the following commands as per your requirement:
  - a. The **lvs** command provides logical volume information in a configurable form, displaying one line per logical volume:

```
# lvs
LV VG Attr LSize Pool Origin Data% Meta% Move Log Cpy%Sync Convert
mylv myvg -wi-ao---- 500.00m
```

- b. The **lvdisplay** command displays logical volume properties, such as size, layout, and mapping in a fixed format:

```
# lvdisplay -v /dev/myvg/mylv
--- Logical volume ---
LV Path                /dev/myvg/mylv
LV Name                 mylv
VG Name                 myvg
LV UUID                 YTnAk6-kMIT-c4pG-HBFZ-Bx7t-ePMk-7YjhaM
LV Write Access         read/write
[.]
```

- c. The **lvscan** command scans for all logical volumes in the system and lists them:

```
# lvscan
ACTIVE                  '/dev/myvg/mylv' [500.00 MiB] inherit
```

3. Create a file system on the logical volume. The following command creates an **xfs** file system on the logical volume:

```
# mkfs.xfs /dev/myvg/mylv
meta-data=/dev/myvg/mylv  isize=512  agcount=4, agsize=32000 blks
          =                sectsz=512  attr=2, projid32bit=1
          =                crc=1      finobt=1, sparse=1, rmapbt=0
          =                reflink=1
data      =                bsize=4096  blocks=128000, imaxpct=25
          =                sunit=0    swidth=0 blks
naming    =version 2       bsize=4096  ascii-ci=0, ftype=1
log       =internal log   bsize=4096  blocks=1368, version=2
          =                sectsz=512  sunit=0 blks, lazy-count=1
realtime  =none           extsz=4096  blocks=0, rtextents=0
Discarding blocks...Done.
```

4. Mount the logical volume and report the file system disk space usage:

```
# mount /dev/myvg/mylv /mnt

# df -h
Filesystem                1K-blocks  Used  Available Use% Mounted on
/dev/mapper/myvg-mylv 506528 29388 477140   6% /mnt
```

#### Additional resources

- **lvcreate(8)**, **lvdisplay(8)**, **lvs(8)**, **lvscan(8)**, **lvm(8)** and **mkfs.xfs(8)** man pages

#### 68.4.4. Creating a RAID0 striped logical volume

A RAID0 logical volume spreads logical volume data across multiple data subvolumes in units of stripe size. The following procedure creates an LVM RAID0 logical volume called *mylv* that stripes data across the disks.

## Prerequisites

1. You have created three or more physical volumes. For more information on creating physical volumes, see [Creating LVM physical volume](#).
2. You have created the volume group. For more information, see [Creating LVM volume group](#).

## Procedure

1. Create a RAID0 logical volume from the existing volume group. The following command creates the RAID0 volume *mylv* from the volume group *myvg*, which is 2G in size, with three stripes and a stripe size of 4kB:

```
# lvcreate --type raid0 -L 2G --stripes 3 --stripesize 4 -n mylv my_vg
Rounding size 2.00 GiB (512 extents) up to stripe boundary size 2.00 GiB(513 extents).
Logical volume "mylv" created.
```

2. Create a file system on the RAID0 logical volume. The following command creates an ext4 file system on the logical volume:

```
# mkfs.ext4 /dev/my_vg/mylv
```

3. Mount the logical volume and report the file system disk space usage:

```
# mount /dev/my_vg/mylv /mnt

# df
Filesystem          1K-blocks  Used Available Use% Mounted on
/dev/mapper/my_vg-mylv 2002684  6168 1875072  1% /mnt
```

## Verification

- View the created RAID0 striped logical volume:

```
# lvs -a -o +devices,segtype my_vg
LV VG Attr LSize Pool Origin Data% Meta% Move Log Cpy%Sync Convert Devices Type
mylv my_vg rwi-a-r--- 2.00g mylv_rimage_0(0),mylv_rimage_1(0),mylv_rimage_2(0) raid0
[mylv_rimage_0] my_vg iwi-aor--- 684.00m /dev/sdf1(0) linear
[mylv_rimage_1] my_vg iwi-aor--- 684.00m /dev/sdg1(0) linear
[mylv_rimage_2] my_vg iwi-aor--- 684.00m /dev/sdh1(0) linear
```

### 68.4.5. Renaming LVM logical volumes

This procedure describes how to rename an existing logical volume *mylv* to *mylv1*.

## Procedure

1. If the logical volume is currently mounted, unmount the volume:

```
# umount /mnt
```

Replace `/mnt` with the mount point.

- Rename an existing logical volume:

```
# lvrename myvg mylv mylv1
Renamed "mylv" to "mylv1" in volume group "myvg"
```

You can also rename the logical volume by specifying the full paths to the devices:

```
# lvrename /dev/myvg/mylv /dev/myvg/mylv1
```

### Additional resources

- **lvrename(8)** man page

## 68.4.6. Removing a disk from a logical volume

This procedure describes how to 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.

### Procedure

- View the used and free space of physical volumes when using the LV:

```
# pvs -o+pv_used
PV      VG   Fmt  Attr  PSize   PFree   Used
/dev/vdb1 myvg lvm2  a--  1020.00m  0      1020.00m
/dev/vdb2 myvg lvm2  a--  1020.00m  0      1020.00m
/dev/vdb3 myvg lvm2  a--  1020.00m 1008.00m 12.00m
```

- Move the data to other physical volume:
  - If there are enough free extents on the other physical volumes in the existing volume group, use the following command to move the data:

```
# pvmove /dev/vdb3
/dev/vdb3: Moved: 2.0%
...
/dev/vdb3: Moved: 79.2%
...
/dev/vdb3: Moved: 100.0%
```

- If there are no enough free extents on the other physical volumes in the existing volume group, use the following commands to add a new physical volume, extend the volume group using the newly created physical volume, and move the data to this physical volume:

```
# pvcreate /dev/vdb4
Physical volume "/dev/vdb4" successfully created
```



```
# vgextend myvg /dev/vdb4
Volume group "myvg" successfully extended

# pvmove /dev/vdb3 /dev/vdb4
/dev/vdb3: Moved: 33.33%
/dev/vdb3: Moved: 100.00%
```

3. Remove the physical volume:

```
# vgreduce myvg /dev/vdb3
Removed "/dev/vdb3" from volume group "myvg"
```

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:

```
# vgreduce --removemissing myvg
```

#### Additional resources

- **pvmove(8)**, **vgextend(8)**, **vereduce(8)**, and **pvs(8)** man pages

### 68.4.7. Removing LVM logical volumes

This procedure describes how to remove an existing logical volume `/dev/myvg/mylv1` from the volume group `myvg`.

#### Procedure

1. If the logical volume is currently mounted, unmount the volume:

```
# umount /mnt
```

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:

```
# lvchange --activate n vg-name/lv-name
```

3. Remove the logical volume using the **lvremove** utility:

```
# lvremove /dev/myvg/mylv1
```

```
Do you really want to remove active logical volume "mylv1"? [y/n]: y
Logical volume "mylv1" 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

- **lvremove(8)** man page

### 68.4.8. 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.

### 68.4.9. 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.

### 68.4.10. Managing LVM logical volumes using RHEL System Roles

Use 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

#### 68.4.10.1. Example Ansible playbook to manage logical volumes

This section provides an example Ansible playbook. This playbook applies the **storage** role to create an LVM logical volume in a volume group.

##### Example 68.1. A playbook that creates a mylv logical volume in the myvg volume group

```
- hosts: all
vars:
  storage_pools:
    - name: myvg
      disks:
```

```

- sda
- sdb
- sdc
volumes:
- name: mylv
  size: 2G
  fs_type: ext4
  mount_point: /mnt/data
roles:
- rhel-system-roles.storage

```

- The **myvg** volume group consists of the following disks:
  - **/dev/sda**
  - **/dev/sdb**
  - **/dev/sdc**
- If the **myvg** volume group already exists, the playbook adds the logical volume to the volume group.
- If the **myvg** volume group does not exist, the playbook creates it.
- The playbook creates an Ext4 file system on the **mylv** logical volume, and persistently mounts the file system at **/mnt**.

#### Additional resources

- The **/usr/share/ansible/roles/rhel-system-roles.storage/README.md** file.

#### 68.4.10.2. Additional resources

- [Managing local storage using RHEL System Roles](#)

### 68.4.11. Removing LVM volume groups

This procedure describes how to remove an existing volume group.

#### Prerequisites

- The volume group contains no logical volumes. To remove logical volumes from a volume group, see [Removing LVM logical volumes](#).

#### Procedure

1. If the volume group exists in a clustered environment, stop the **lockspace** of the volume group on all other nodes. Use the following command on all nodes except the node where you are performing the removing:

```
# vgchange --lockstop vg-name
```

Wait for the lock to stop.

2. Remove the volume group:

```
# vgrename vg-name  
Volume group "vg-name" successfully removed
```

### Additional resources

- **vgremove(8)** man page

## 68.5. MODIFYING THE SIZE OF A LOGICAL VOLUME

After you have created a logical volume, you can modify the size of the volume.

### 68.5.1. Growing a logical volume and file system

This procedure describes how to extend the logical volume and grow a file system on the same logical volume.

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.

#### Prerequisites

1. You have an existing logical volume (LV) with a file system on it. Determine the file system type by using the **df -Th** command.  
For more information on creating LV and a file system, see [Creating LVM logical volume](#).
2. You have sufficient space in the volume group to grow your LV and file system. Use the **vgs -o name,vgfree** command to determine the available space.

#### Procedure

1. Optional: If the volume group has insufficient space to grow your LV, then add a new physical volume to the volume group by using the following command:

```
# vgextend myvg /dev/vdb3  
Physical volume "/dev/vdb3" successfully created.  
Volume group "myvg" successfully extended
```

For more information, see [Creating LVM volume group](#).

2. Now that the volume group is large enough, execute any one of the following steps as per your requirement:
  - a. To extend the LV with the provided size, use the following command:

```
# lvextend -L 3G /dev/myvg/mylv  
Size of logical volume myvg/mylv changed from 2.00 GiB (512 extents) to 3.00 GiB (768  
extents).  
Logical volume myvg/mylv successfully resized.
```

**NOTE**

You can use the **-r** option of the **lvextend** command to extend the logical volume and resize the underlying file system with a single command:

```
# lvextend -r -L 3G /dev/myvg/mylv
```

**WARNING**

You can also extend the logical volume using the **lvresize** command with the same arguments, but this command does not guarantee against accidental shrinkage.

- b. To extend the *mylv* logical volume to fill all of the unallocated space in the *myvg* volume group, use the following command:

```
# lvextend -l +100%FREE /dev/myvg/mylv
Size of logical volume myvg/mylv changed from 10.00 GiB (2560 extents) to 6.35 TiB (1665465 extents).
Logical volume myvg/mylv successfully resized.
```

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.

3. If you are not using the **r** option with the **lvextend** command to extend the LV and resize the file system with a single command, then resize the file system on the logical volume by using the following command:

```
xfs_growfs /mnt/mnt1/
meta-data=/dev/mapper/myvg-mylv isize=512  agcount=4, agsize=65536 blks
          =               sectsz=512  attr=2, projid32bit=1
          =               crc=1      finobt=1, sparse=1, rmapbt=0
          =               reflink=1
data     =               bsize=4096  blocks=262144, imaxpct=25
          =               sunit=0    swidth=0 blks
naming   =version 2      bsize=4096  ascii-ci=0, ftype=1
log      =internal log  bsize=4096  blocks=2560, version=2
          =               sectsz=512  sunit=0 blks, lazy-count=1
realtime =none          extsz=4096  blocks=0, rtextents=0
data blocks changed from 262144 to 524288
```

**NOTE**

Without the **-D** option, **xfs\_growfs** grows the file system to the maximum size supported by the underlying device. For more information, see [Increasing the size of an XFS file system](#).

For resizing an ext4 file system, see [Resizing an ext4 file system](#).

## Verification

- Verify if the file system is growing by using the following command:

```
# df -Th
Filesystem      Type      Size  Used Avail Use% Mounted on
devtmpfs        devtmpfs  1.9G   0 1.9G   0% /dev
tmpfs           tmpfs     1.9G   0 1.9G   0% /dev/shm
tmpfs           tmpfs     1.9G  8.6M 1.9G   1% /run
tmpfs           tmpfs     1.9G   0 1.9G   0% /sys/fs/cgroup
/dev/mapper/rhel-root xfs      45G  3.7G 42G   9% /
/dev/vda1       xfs     1014M  369M 646M  37% /boot
tmpfs           tmpfs     374M   0 374M   0% /run/user/0
/dev/mapper/myvg-mylv xfs      2.0G   47M 2.0G   3% /mnt/mnt1
```

## Additional resources

- [vgextend\(8\)](#), [lvextend\(8\)](#), and [xfs\\_growfs\(8\)](#) man pages

## 68.5.2. 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 **--resizefs** 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 **--resizefs** 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.

## Procedure

- To shrink the *mylv* logical volume in *myvg* volume group to 64 megabytes, use the following command:

```
# lvreduce --resizefs -L 64M myvg/mylv
fsck from util-linux 2.37.2
/dev/mapper/myvg-myLV: clean, 11/25688 files, 4800/102400 blocks
resize2fs 1.46.2 (28-Feb-2021)
Resizing the filesystem on /dev/mapper/myvg-myLV to 65536 (1k) blocks.
The filesystem on /dev/mapper/myvg-myLV is now 65536 (1k) blocks long.

Size of logical volume myvg/myLV changed from 100.00 MiB (25 extents) to 64.00 MiB (16
extents).
Logical volume myvg/myLV successfully resized.
```

In this example, *mylv* contains a file system, which this command resizes together with the logical volume.

- Specifying the **-** sign before the resize value indicates that the value will be subtracted from the logical volume's actual size. To shrink a logical volume to an absolute size of 64 megabytes, use the following command:

```
# lvreduce --resizefs -L -64M myvg/mylv
```

## Additional resources

- lvreduce(8)** man page

### 68.5.3. 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 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
```

## 68.6. 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 **lvreport(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 **lvms fullreport** command. For information on this command and its capabilities, see the **lvms-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 **lvmsreport(7)** man page.

### 68.6.1. 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-ldwM-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-ldwM-01S9-XO8M-
mcpsVe
/dev/sdd1 new_vg lvm2 a- 17.14G 17.14G 17.14G yfvZK-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
```

## 68.6.2. LVM object display fields

You can display additional information about the LVM objects with the **pvs**, **vgs**, and **lvs** commands.

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 68.1, “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 68.1. 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

Argument	Header	Description
<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
<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

By default, the **pvs** command displays the **pv\_name**, **vg\_name**, **pv\_fmt**, **pv\_attr**, **pv\_size** and **pv\_free** fields. 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 view devices detected by LVM that are not 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 68.2, “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 68.2. 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)

Argument	Header	Description
<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)riteable, (r)eadonly, resi(z)eable, e(x)ported, (p)artial and (c)lustered.
<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 **vg\_extent\_size** and **vg\_uuid** fields to the default display.

```
# 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 68.3, “*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 68.3. *lvs* Display Fields

Argument	Header	Description
* <b>chunksize</b> * <b>chunk_size</b>	Chunk	Unit size in a snapshot volume
<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, ®aid without initial sync, (s)napshot, merging (S)napshot, (p)vmove, (v)irtual, mirror or raid (i)mage, mirror or raid (l)mage 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, ®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)uspended snapshot, snapshot (m)erge failed, suspended snapshot (M)erge failed, mapped (d)evice</li> </ul>

Argument	Header	Description
		<p>present without tables, mapped device present with (t)inactive table</p> <ul style="list-style-type: none"> <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 driver, whereas the raid equivalents using the md raid kernel driver all 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).</li> <li>* Bit 8: Newly-allocated data blocks are overwritten with blocks of (z)eros before use.</li> <li>* 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.</li> <li>* Bit 10: s(k)ip activation: this volume is flagged to be skipped during activation.</li> </ul>
<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



Argument	Header	Description
<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
<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> * <b>region_size</b>	Region	The unit size of a mirrored logical volume
<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> * <b>stripe_size</b>	Stripe	Unit size of the stripe in a striped logical volume

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                100.00
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-Ojsl-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 **seg\_start**, **stripesize** and **chunksizes** fields to the default display.

```
# 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
```

### 68.6.3. 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

```

#### 68.6.4. 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.

##### Base 2 units

The default units displayed in powers of 2 (multiples of 1024). You can specify:

- human-readable (**r**) with **<** rounding indicator
- bytes (**b**)
- sectors (**s**)
- kilobytes (**k**)
- megabytes (**m**)
- gigabytes (**g**)
- terabytes (**t**)
- petabytes (**p**)
- exabytes (**e**)
- human-readable (**h**), which is the default unit

The default display is **r**, human-readable. You can override the default by setting the units parameter in the global section of the **/etc/lvm/lvm.conf** file.

##### Base 10 units

You can specify the units to be displayed in multiples of 1000 by capitalizing the unit specification (**R, B, S, K, M, G, T, P, E, H**).

The following example specifies the output of the **pvs**, **vgs** and **lvs** commands in base 2 gigabytes unit:

```

# pvs --units g /dev/sdb
PV    VG  Fmt Attr PSize  PFree
/dev/sdb test lvm2 a-- 931.00g 930.00g

```

```

# vgs --units g test
VG  #PV #LV #SN Attr VSize  VFree
test 1  1  0 wz-n 931.00g 931.00g

```

```

# lvs --units g test
LV  VG  Attr  LSize Pool Origin Data%  Meta%  Move Log Cpy%Sync Convert
lv00 test wi-a---- 1.00g

```

The following example specifies the output of the **pvs**, **vgs** and **lvs** commands in base 10 gigabytes unit:

-

```
# pvs --units G /dev/sdb
PV      VG  Fmt Attr PSize  PFree
/dev/sdb test lvm2 a-- 999.65G 998.58G
```

```
# vgs --units G test
VG  #PV #LV #SN Attr VSize  VFree
test 1  1  0 wz-n 999.65G 998.58G
```

```
# lvs --units G test
LV  VG  Attr  LSize Pool Origin Data%  Meta%  Move Log Cpy%Sync Convert
lv0 test wi-a---- 1.07G
```

You can specify sectors (**s**), defined as 512 bytes, or custom units. The following example displays the output of the **pvs** command as several sectors:

```
# pvs --units s
PV      VG  Fmt Attr PSize    PFree
/dev/sdb test lvm2 a-- 1952440320S 1950343168S
```

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/sdb test lvm2 a-- 238335.00U 238079.00U
```

The purpose of the **r** unit is that it works similarly to **h** (human-readable), but in addition, the reported value gets a prefix of **<** or **>** to indicate that the actual size is slightly more or less than the displayed size. The **r** setting is the default for LVM commands. LVM rounds the decimal value, causing non-exact sizes to be reported. Notice the following:

```
# vgs --units g test
VG  #PV #LV #SN Attr VSize  VFree
test 1  1  0 wz-n 931.00g 930.00g
```

```
# vgs --units r test
VG  #PV #LV #SN Attr VSize  VFree
test 1  1  0 wz-n <931.00g <930.00
```

```
# vgs test
VG  #PV #LV #SN Attr VSize  VFree
test 1  1  0 wz-n <931.00g <930.00g
```

Note that the **r** is the default unit when **--units** is not specified. It also shows how **--units g** (or other **--units**) do not always display exactly correct sizes. It also shows the primary purpose of **r**, which is the **<** to indicate that the displayed size is not exact. In this example, the value is not exact because the VG size is not an exact multiple of gigabytes, and **.01** is also not an exact representation of the fraction.

### 68.6.5. 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.

### 68.6.6. 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 **lv010** and **lv011** 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
lv01 4.00m 100.00
lv00 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 **lvreport** man page.

## 68.7. CONFIGURING RAID LOGICAL VOLUMES

You can create, activate, change, remove, display, and use LVM Redundant Array of Independent Disks (RAID) volumes.

### 68.7.1. RAID logical volumes

Logical volume manager (LVM) supports Redundant Array of Independent Disks (RAID) levels 0, 1, 4, 5, 6, and 10. An LVM RAID volume has the following characteristics:

- LVM creates and manages RAID logical volumes that leverage the Multiple Devices (MD) kernel drivers.
- You can temporarily split RAID1 images from the array and merge them back into the array later.
- LVM RAID volumes support snapshots.

Other characteristics include:

#### Clusters

RAID logical volumes are not cluster-aware.

Although you can create and activate RAID logical volumes exclusively on one machine, you cannot activate them simultaneously on more than one machine.

#### Subvolumes

When you create a RAID logical volume (LV), 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 and one implicit parity device, RAID4 results in four metadata subvolumes (**lv\_rmeta\_0**, **lv\_rmeta\_1**, **lv\_rmeta\_2**, and **lv\_rmeta\_3**) and four data subvolumes (**lv\_rimage\_0**, **lv\_rimage\_1**, **lv\_rimage\_2**, and **lv\_rimage\_3**).

#### Integrity

You can lose data when a RAID device fails or when soft corruption occurs. Soft corruption in data storage implies that the data retrieved from a storage device is different from the data written to that device. Adding integrity to a RAID LV reduces or prevent soft corruption. For more information, see [Creating a RAID LV with DM integrity](#).

## 68.7.2. RAID levels and linear support

The following are the supported configurations by RAID, including levels 0, 1, 4, 5, 6, 10, and linear:

### Level 0

RAID level 0, often called striping, is a performance-oriented striped data mapping technique. This means the data being written to the array is broken down into stripes and written across the member disks of the array, allowing high I/O performance at low inherent cost but provides no redundancy. RAID level 0 implementations only stripe the data across the member devices up to the size of the smallest device in the array. This means that if you have multiple devices with slightly different sizes, each device gets treated as though it was the same size as the smallest drive. Therefore, the common storage capacity of a level 0 array is the total capacity of all disks. If the member disks have a different size, then the RAID0 uses all the space of those disks using the available zones.

### Level 1

RAID level 1, or mirroring, provides redundancy by writing identical data to each member disk of the array, leaving a mirrored copy on each disk. Mirroring remains popular due to its simplicity and high level of data availability. Level 1 operates with two or more disks, and provides very good data reliability and improves performance for read-intensive applications but at relatively high costs. RAID level 1 is costly because you write the same information to all of the disks in the array, which provides data reliability, but in a much less space-efficient manner than parity based RAID levels such as level 5. However, this space inefficiency comes with a performance benefit, which is parity-based RAID levels that consume considerably more CPU power in order to generate the parity while RAID level 1 simply writes the same data more than once to the multiple RAID members with very little CPU overhead. As such, RAID level 1 can outperform the parity-based RAID levels on machines where software RAID is employed and CPU resources on the machine are consistently taxed with operations other than RAID activities.

The storage capacity of the level 1 array is equal to the capacity of the smallest mirrored hard disk in a hardware RAID or the smallest mirrored partition in a software RAID. Level 1 redundancy is the highest possible among all RAID types, with the array being able to operate with only a single disk present.

### Level 4

Level 4 uses parity concentrated on a single disk drive to protect data. Parity information is calculated based on the content of the rest of the member disks in the array. This information can then be used to reconstruct data when one disk in the array fails. The reconstructed data can then be used to satisfy I/O requests to the failed disk before it is replaced and to repopulate the failed disk after it has been replaced.

Since the dedicated parity disk represents an inherent bottleneck on all write transactions to the RAID array, level 4 is seldom used without accompanying technologies such as write-back caching. Or it is used in specific circumstances where the system administrator is intentionally designing the software RAID device with this bottleneck in mind such as an array that has little to no write transactions once the array is populated with data. RAID level 4 is so rarely used that it is not available as an option in Anaconda. However, it could be created manually by the user if needed.

The storage capacity of hardware RAID level 4 is equal to the capacity of the smallest member partition multiplied by the number of partitions minus one. The performance of a RAID level 4 array is always asymmetrical, which means reads outperform writes. This is because write operations consume extra CPU resources and main memory bandwidth when generating parity, and then also consume extra bus bandwidth when writing the actual data to disks because you are not only writing the data, but also the parity. Read operations need only read the data and not the parity unless the array is in a degraded state. As a result, read operations generate less traffic to the drives and across the buses of the computer for the same amount of data transfer under normal operating conditions.



## Level 5

This is the most common type of RAID. By distributing parity across all the member disk drives of an array, RAID level 5 eliminates the write bottleneck inherent in level 4. The only performance bottleneck is the parity calculation process itself. Modern CPUs can calculate parity very fast. However, if you have a large number of disks in a RAID 5 array such that the combined aggregate data transfer speed across all devices is high enough, parity calculation can be a bottleneck. Level 5 has asymmetrical performance, and reads substantially outperforming writes. The storage capacity of RAID level 5 is calculated the same way as with level 4.

## Level 6

This is a common level of RAID when data redundancy and preservation, and not performance, are the paramount concerns, but where the space inefficiency of level 1 is not acceptable. Level 6 uses a complex parity scheme to be able to recover from the loss of any two drives in the array. This complex parity scheme creates a significantly higher CPU burden on software RAID devices and also imposes an increased burden during write transactions. As such, level 6 is considerably more asymmetrical in performance than levels 4 and 5.

The total capacity of a RAID level 6 array is calculated similarly to RAID level 5 and 4, except that you must subtract two devices instead of one from the device count for the extra parity storage space.

## Level 10

This RAID level attempts to combine the performance advantages of level 0 with the redundancy of level 1. It also reduces some of the space wasted in level 1 arrays with more than two devices. With level 10, it is possible, for example, to create a 3-drive array configured to store only two copies of each piece of data, which then allows the overall array size to be 1.5 times the size of the smallest devices instead of only equal to the smallest device, similar to a 3-device, level 1 array. This avoids CPU process usage to calculate parity similar to RAID level 6, but it is less space efficient.

The creation of RAID level 10 is not supported during installation. It is possible to create one manually after installation.

## Linear RAID

Linear RAID is a grouping of drives to create a larger virtual drive.

In linear RAID, the chunks are allocated sequentially from one member drive, going to the next drive only when the first is completely filled. This grouping provides no performance benefit, as it is unlikely that any I/O operations split between member drives. Linear RAID also offers no redundancy and decreases reliability. If any one member drive fails, the entire array cannot be used and data can be lost. The capacity is the total of all member disks.

### 68.7.3. LVM RAID segment types

To create a RAID logical volume, you can specify a RAID type by using the **--type** argument of the **lvcreate** command. For most users, specifying one of the five available primary types, which are **raid1**, **raid4**, **raid5**, **raid6**, and **raid10**, should be sufficient.

The following table describes the possible RAID segment types.

**Table 68.4. LVM RAID segment types**

Segment type	Description
--------------	-------------

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> argument without specifying striping.
<b>raid4</b>	RAID4 dedicated parity disk.
<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>● It is same as <b>raid5</b>.</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_zr</b>	<ul style="list-style-type: none"> <li>● RAID6 zero restart.</li> <li>● It is same as <b>raid6</b>.</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>
<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>

Segment type	Description
<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> argument along with the 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 is lost if any of the data subvolumes fail.

### 68.7.4. Creating RAID logical volumes

You can create RAID1 arrays with multiple numbers of copies, according to the value you specify for the **-m** argument. Similarly, you can specify the number of stripes for a RAID 0, 4, 5, 6, and 10 logical volume with the **-i** argument. You can also specify the stripe size with the **-l** argument. The following procedure describes different ways to create different types of RAID logical volume.

#### Procedure

- Create a 2-way RAID. The following command creates a 2-way RAID1 array, named *my\_lv*, in the volume group *my\_vg*, that is 1G in size:

```
# lvcreate --type raid1 -m 1 -L 1G -n my_lv my_vg
Logical volume "my_lv" created.
```

- Create a RAID5 array with stripes. The following command creates a RAID5 array with three stripes and one implicit parity drive, named *my\_lv*, in the volume group *my\_vg*, that is 1G in size. Note that you can specify the number of stripes similar to 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
```

- Create a RAID6 array with stripes. The following command creates a RAID6 array with three 3 stripes and two implicit parity drives, named *my\_lv*, in the volume group *my\_vg*, that is 1G one gigabyte in size:

```
# lvcreate --type raid6 -i 3 -L 1G -n my_lv my_vg
```

#### Verification

- Display 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)
```

### Additional resources

- [lvcreate\(8\)](#) and [lvraid\(7\)](#) man pages

## 68.7.5. Creating a RAID0 striped logical volume

A RAID0 logical volume spreads logical volume data across multiple data subvolumes in units of stripe size. The following procedure creates an LVM RAID0 logical volume called *mylv* that stripes data across the disks.

### Prerequisites

1. You have created three or more physical volumes. For more information on creating physical volumes, see [Creating LVM physical volume](#).
2. You have created the volume group. For more information, see [Creating LVM volume group](#).

### Procedure

1. Create a RAID0 logical volume from the existing volume group. The following command creates the RAID0 volume *mylv* from the volume group *myvg*, which is 2G in size, with three stripes and a stripe size of 4kB:

```
# lvcreate --type raid0 -L 2G --stripes 3 --stripesize 4 -n mylv my_vg
Rounding size 2.00 GiB (512 extents) up to stripe boundary size 2.00 GiB(513 extents).
Logical volume "mylv" created.
```

2. Create a file system on the RAID0 logical volume. The following command creates an ext4 file system on the logical volume:

```
# mkfs.ext4 /dev/my_vg/mylv
```

3. Mount the logical volume and report the file system disk space usage:

```
# mount /dev/my_vg/mylv /mnt

# df
Filesystem          1K-blocks  Used Available Use% Mounted on
/dev/mapper/my_vg-mylv 2002684   6168 1875072   1% /mnt
```

### Verification

- View the created RAID0 striped logical volume:

```
# lvs -a -o +devices,segtype my_vg
LV VG Attr LSize Pool Origin Data% Meta% Move Log Cpy%Sync Convert Devices Type
mylv my_vg rwi-a-r--- 2.00g mylv_rimage_0(0),mylv_rimage_1(0),mylv_rimage_2(0) raid0
```

```
[mylv_rimage_0] my_vg iwi-aor--- 684.00m /dev/sdf1(0) linear
[mylv_rimage_1] my_vg iwi-aor--- 684.00m /dev/sdg1(0) linear
[mylv_rimage_2] my_vg iwi-aor--- 684.00m /dev/sdh1(0) linear
```

### 68.7.6. Parameters for creating a RAID0

You can create a RAID0 striped logical volume using the **lvcreate --type raid0[meta] --stripes *Stripes* --stripesize *StripeSize* *VolumeGroup* [*PhysicalVolumePath*]** command.

The following table describes different parameters, which you can use while creating a RAID0 striped logical volume.

**Table 68.5. Parameters for creating a RAID0 striped logical volume**

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. Since RAID0 is non-resilient, it does not 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. 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.
<b><i>PhysicalVolumePath</i></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.

### 68.7.7. Soft data corruption

Soft corruption in data storage implies that the data retrieved from a storage device is different from the data written to that device. The corrupted data can exist indefinitely on storage devices. You might not discover this corrupted data until you retrieve and attempt to use this data.

Depending on the type of configuration, a Redundant Array of Independent Disks (RAID) logical volume(LV) prevents data loss when a device fails. If a device consisting of a RAID array fails, the data can be recovered from other devices that are part of that RAID LV. However, a RAID configuration does not ensure the integrity of the data itself. Soft corruption, silent corruption, soft errors, and silent errors are terms that describe data that has become corrupted, even if the system design and software continues to function as expected.

Device mapper (DM) integrity is used with RAID levels 1, 4, 5, 6, and 10 to mitigate or prevent data loss due to soft corruption. The RAID layer ensures that a non-corrupted copy of the data can fix the soft corruption errors. The integrity layer sits above each RAID image while an extra sub LV stores the integrity metadata or data checksums for each RAID image. When you retrieve data from an RAID LV

with integrity, the integrity data checksums analyze the data for corruption. If corruption is detected, the integrity layer returns an error message, and the RAID layer retrieves a non-corrupted copy of the data from another RAID image. The RAID layer automatically rewrites non-corrupted data over the corrupted data to repair the soft corruption.

When creating a new RAID LV with DM integrity or adding integrity to an existing RAID LV, consider the following points:

- The integrity metadata requires additional storage space. For each RAID image, every 500MB data requires 4MB of additional storage space because of the checksums that get added to the data.
- While some RAID configurations are impacted more than others, adding DM integrity impacts performance due to latency when accessing the data. A RAID1 configuration typically offers better performance than RAID5 or its variants.
- The RAID integrity block size also impacts performance. Configuring a larger RAID integrity block size offers better performance. However, a smaller RAID integrity block size offers greater backward compatibility.
- There are two integrity modes available: **bitmap** or **journal**. The **bitmap** integrity mode typically offers better performance than **journal** mode.

## TIP

If you experience performance issues, either use RAID1 with integrity or test the performance of a particular RAID configuration to ensure that it meets your requirements.

### 68.7.8. Creating a RAID LV with DM integrity

When you create a RAID LV with device mapper (DM) integrity or add integrity to an existing RAID LV, it mitigates the risk of losing data due to soft corruption. Wait for the integrity synchronization and the RAID metadata to complete before using the LV. Otherwise, the background initialization might impact the LV's performance.

#### Procedure

1. Create a RAID LV with DM integrity. The following example creates a new RAID LV with integrity named *test-lv* in the *my\_vg* volume group, with a usable size of *256M* and RAID level *1*:

```
# lvcreate --type raid1 --raidintegrity y -L 256M -n test-lv my_vg
Creating integrity metadata LV test-lv_rimage_0_imeta with size 8.00 MiB.
Logical volume "test-lv_rimage_0_imeta" created.
Creating integrity metadata LV test-lv_rimage_1_imeta with size 8.00 MiB.
Logical volume "test-lv_rimage_1_imeta" created.
Logical volume "test-lv" created.
```



#### NOTE

To add DM integrity to an existing RAID LV, use the following command:

```
# lvconvert --raidintegrity y my_vg/test-lv
```

Adding integrity to a RAID LV limits the number of operations that you can perform on that RAID LV.

2. Optional: Remove the integrity before performing certain operations.

```
# lvconvert --raidintegrity n my_vg/test-lv
Logical volume my_vg/test-lv has removed integrity.
```

## Verification

- View information about the added DM integrity:
  - View information about the test-lv RAID LV that was created in the *my\_vg* volume group:

```
# lvs -a my_vg
LV          VG   Attr   LSize  Origin              Cpy%Sync
test-lv     my_vg rwi-a-r--- 256.00m              2.10
[test-lv_rimage_0] my_vg gwi-aor--- 256.00m [test-lv_rimage_0_iorig] 93.75
[test-lv_rimage_0_imeta] my_vg ewi-ao---- 8.00m
[test-lv_rimage_0_iorig] my_vg -wi-ao---- 256.00m
[test-lv_rimage_1] my_vg gwi-aor--- 256.00m [test-lv_rimage_1_iorig] 85.94
[...]
```

The following describes different options from this output:

### g attribute

It is the list of attributes under the Attr column indicates that the RAID image is using integrity. The integrity stores the checksums in the **\_imeta** RAID LV.

### Cpy%Sync column

It indicates the synchronization progress for both the top level RAID LV and for each RAID image.

### RAID image

It is indicated in the LV column by **raid\_image\_N**.

### LV column

It ensures that the synchronization progress displays 100% for the top level RAID LV and for each RAID image.

- Display the type for each RAID LV:

```
# lvs -a my_vg -o+segtype
LV          VG   Attr   LSize  Origin              Cpy%Sync Type
test-lv     my_vg rwi-a-r--- 256.00m              87.96  raid1
[test-lv_rimage_0] my_vg gwi-aor--- 256.00m [test-lv_rimage_0_iorig] 100.00
integrity
[test-lv_rimage_0_imeta] my_vg ewi-ao---- 8.00m              linear
[test-lv_rimage_0_iorig] my_vg -wi-ao---- 256.00m              linear
[test-lv_rimage_1] my_vg gwi-aor--- 256.00m [test-lv_rimage_1_iorig] 100.00
integrity
[...]
```

- There is an incremental counter that counts the number of mismatches detected on each RAID image. View the data mismatches detected by integrity from **rimage\_0** under *my\_vg/test-lv*:

```
# lvs -o+integritymismatches my_vg/test-lv_rimage_0
LV          VG   Attr   LSize  Origin              Cpy%Sync IntegMismatches
[test-lv_rimage_0] my_vg gwi-aor--- 256.00m [test-lv_rimage_0_iorig] 100.00
0
```

In this example, the integrity has not detected any data mismatches and thus the **IntegMismatches** counter shows zero (0).

- View the data integrity information in the `/var/log/messages` log files, as shown in the following examples:

#### Example 68.2. Example of dm-integrity mismatches from the kernel message logs

```
device-mapper: integrity: dm-12: Checksum failed at sector 0x24e7
```

#### Example 68.3. Example of dm-integrity data corrections from the kernel message logs

```
md/raid1:mdX: read error corrected (8 sectors at 9448 on dm-16)
```

### Additional resources

- lvcreate(8)** and **lvraid(7)** man pages

## 68.7.9. Minimum and maximum I/O rate options

When you create a RAID logical volumes, the background I/O required to initialize the logical volumes with the sync operation can expel 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. To control the rate at which **sync** operations are performed, set the minimum and maximum I/O rate for those operations with the **--minrecoveryrate** and **--maxrecoveryrate** options of the **lvcreate** command.

You can specify these options as follows:

#### **--maxrecoveryrate Rate[bBsSkKmMgG]**

Sets the maximum recovery rate for a RAID logical volume so that it will not expel nominal I/O operations. Specify the Rate as an amount per second for each device in the array. If you do not provide a suffix, then it assumes kiB/sec/device. Setting the recovery rate to 0 means it will be unbounded.

#### **--minrecoveryrate Rate[bBsSkKmMgG]**

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. Specify the Rate as an amount per second for each device in the array. If you do not give a suffix, then it assumes kiB/sec/device.

For example, use the **lvcreate --type raid10 -i 2 -m 1 -L 10G --maxrecoveryrate 128 -n my\_lv my\_vg** command to create a 2-way RAID10 array `my_lv`, which is in the volume group `my_vg` with 3 stripes that is 10G in size with a maximum recovery rate of 128 kiB/sec/device. You can also specify minimum and maximum recovery rates for a RAID scrubbing operation.



### 68.7.10. Converting a Linear device to a RAID logical volume

You can convert an existing linear logical volume to a RAID logical volume. To perform this operation, use the **--type** argument of the **lvconvert** command.

RAID logical volumes are composed of metadata and data subvolume pairs. When you convert a linear device to a RAID1 array, it creates a new metadata subvolume and associates it with the original logical volume on one of the same physical volumes that the linear volume is on. The additional images are added in a metadata/data subvolume pair. If the metadata image that pairs with the original logical volume cannot be placed on the same physical volume, the **lvconvert** fails.

#### Procedure

1. View the logical volume device that needs to be converted:

```
# lvs -a -o name,copy_percent,devices my_vg
LV   Copy%  Devices
my_lv      /dev/sde1(0)
```

2. Convert the linear logical volume to a RAID device. 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
Are you sure you want to convert linear LV my_vg/my_lv to raid1 with 2 images enhancing
resilience? [y/n]: y
Logical volume my_vg/my_lv successfully converted.
```

#### Verification

- Ensure if the logical volume is converted to a RAID device:

```
# 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)
```

#### Additional resources

- The **lvconvert(8)** man page

### 68.7.11. 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. To perform this operation, use the **lvconvert** command and specify 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.

#### Procedure

1. Display 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)
```

- Convert an existing RAID1 LVM logical volume to an LVM linear logical volume. The following command converts the LVM RAID1 logical volume `my_vg/my_lv` to an LVM linear device:

```
# lvconvert -m0 my_vg/my_lv
Are you sure you want to convert raid1 LV my_vg/my_lv to type linear losing all resilience?
[y/n]: y
Logical volume my_vg/my_lv successfully converted.
```

When you convert an LVM RAID1 logical volume to an LVM linear volume, you can also specify which physical volumes to remove. In the following example, the **lvconvert** command specifies that you want to remove `/dev/sde1`, leaving `/dev/sdf1` as the physical volume that makes up the linear device:

```
# lvconvert -m0 my_vg/my_lv /dev/sde1
```

## Verification

- Verify if the RAID1 logical volume was converted to an LVM linear device:

```
# lvs -a -o name,copy_percent,devices my_vg
LV Copy% Devices
my_lv /dev/sdf1(1)
```

## Additional resources

- The **lvconvert(8)** man page

## 68.7.12. Converting a mirrored LVM device to a RAID1 logical volume

You can convert an existing mirrored LVM device with a segment type mirror to a RAID1 LVM device. To perform this operation, use the **lvconvert** command with the **--type raid1** argument. This renames the mirror subvolumes named **mimage** to RAID subvolumes named **rimage**.

In addition, it also removes the mirror log and creates metadata subvolumes named **rmeta** for the data subvolumes on the same physical volumes as the corresponding data subvolumes.

## Procedure

- View 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)
```

- Convert the mirrored logical volume `my_vg/my_lv` to a RAID1 logical volume:

```
# lvconvert --type raid1 my_vg/my_lv
Are you sure you want to convert mirror LV my_vg/my_lv to raid1 type? [y/n]: y
Logical volume my_vg/my_lv successfully converted.
```

## Verification

- Verify if the mirrored logical volume is converted to a 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(0)
[my_lv_rimage_1]    /dev/sdf1(0)
[my_lv_rmeta_0]     /dev/sde1(125)
[my_lv_rmeta_1]     /dev/sdf1(125)
```

## Additional resources

- The **lvconvert(8)** man page

### 68.7.13. 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 **lvraid(7)** man page.

### 68.7.14. Changing the number of images in an existing RAID1 device

You can change the number of images in an existing RAID1 array, similar to the way you can change the number of images in the implementation of LVM mirroring.

When you add images to a RAID1 logical volume with the **lvconvert** command, you can perform the following operations:

- specify the total number of images for the resulting device,
- how many images to add to the device, and
- can optionally specify on which physical volumes the new metadata/data image pairs reside.

## Procedure

1. Display 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)
```

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.

2. Convert the 2-way RAID1 logical volume `my_vg/my_lv` to a 3-way RAID1 logical volume:

```
# lvconvert -m 2 my_vg/my_lv
Are you sure you want to convert raid1 LV my_vg/my_lv to 3 images enhancing resilience?
[y/n]: y
Logical volume my_vg/my_lv successfully converted.
```

The following are a few examples of changing the number of images in an existing RAID1 device:

- You can also specify which physical volumes to use while adding an image to RAID. The following command converts the 2-way RAID1 logical volume `my_vg/my_lv` to a 3-way RAID1 logical volume, specifying that the physical volume `/dev/sdd1` be used for the array:

```
# lvconvert -m 2 my_vg/my_lv /dev/sdd1
```

- Convert the 3-way RAID1 logical volume into a 2-way RAID1 logical volume:

```
# lvconvert -m1 my_vg/my_lv
Are you sure you want to convert raid1 LV my_vg/my_lv to 2 images reducing resilience?
[y/n]: y
Logical volume my_vg/my_lv successfully converted.
```

- Convert the 3-way RAID1 logical volume into a 2-way RAID1 logical volume by specifying the physical volume `/dev/sde1`, which contains the image to remove:

```
# lvconvert -m1 my_vg/my_lv /dev/sde1
```

Additionally, when you remove an image and its associated metadata subvolume volume, any higher-numbered images will be shifted down to fill the slot. Removing **lv\_rimage\_1** from a 3-way RAID1 array that consists of **lv\_rimage\_0**, **lv\_rimage\_1**, and **lv\_rimage\_2**

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`.

## Verification

- View the RAID1 device after changing the number of images in an existing RAID1 device:

```
# 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/sdd1(1)
[my_lv_rimage_1] /dev/sde1(1)
[my_lv_rimage_2] /dev/sdf1(1)
[my_lv_rmeta_0] /dev/sdd1(0)
[my_lv_rmeta_1] /dev/sde1(0)
[my_lv_rmeta_2] /dev/sdf1(0)
```

## Additional resources

- The `lvconvert(8)` man page

### 68.7.15. 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. When you are removing a RAID image from an existing RAID1 logical volume or removing 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.

## Procedure

- Display 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       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)
```

- Split the RAID image into a separate logical volume. The following example splits a 2-way RAID1 logical volume, `my_lv`, into two linear logical volumes, `my_lv` and `new`:

```
# lvconvert --splitmirror 1 -n new my_vg/my_lv
Are you sure you want to split raid1 LV my_vg/my_lv losing all resilience? [y/n]: y
```

Split a 3-way RAID1 logical volume, `my_lv`, into a 2-way RAID1 logical volume, `my_lv`, and a linear logical volume, `new`:

-

```
# lvconvert --splitmirror 1 -n new my_vg/my_lv
```

## Verification

- View the logical volume after you split off an image of a RAID logical volume:

```
# lvs -a -o name,copy_percent,devices my_vg
LV      Copy%  Devices
my_lv   100.00 /dev/sde1(1)
new     100.00 /dev/sdf1(1)
```

## Additional resources

- The **lvconvert(8)** man page

## 68.7.16. Splitting and Merging a RAID Image

You can temporarily split off an image of a RAID1 array for read-only use while tracking any changes by using the **--trackchanges** argument with the **--splitmirrors** argument of the **lvconvert** command. Using this feature, you can merge the image into an array at a later time while resyncing only those portions of the array that have changed since the image was split.

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. When you merge the image, only the portions of the array that have changed since the image was split are resynced.

## Procedure

- Create a RAID logical volume:

```
# lvcreate --type raid1 -m 2 -L 1G -n my_lv my_vg
Logical volume "my_lv" created
```

- Optional: View the created RAID 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_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)
```

- Split an image from the created RAID logical volume and track the changes to the remaining array:

```
# 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
```

- Optional: View the logical volume after splitting the image:

```
# 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)
```

- Merge the volume back into the array:

```
# lvconvert --merge my_vg/my_lv_rimage_1
my_vg/my_lv_rimage_1 successfully merged back into my_vg/my_lv
```

## Verification

- View the merged 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/sdc1(1)
[my_lv_rimage_1]  /dev/sdd1(1)
[my_lv_rmeta_0]   /dev/sdc1(0)
[my_lv_rmeta_1]   /dev/sdd1(0)
```

## Additional resources

- The **lvconvert(8)** man page

### 68.7.17. 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.

### 68.7.17.1. 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.

### 68.7.17.2. 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.

### 68.7.18. Replacing a RAID device in a logical volume

You can replace a RAID device in a logical volume.

- If there has been no failure on the RAID device, follow [Section 68.7.18.1, “Replacing a RAID device that has not failed”](#).
- If the RAID device has failed, follow [Section 68.7.18.4, “Replacing a failed RAID device in a logical volume”](#).

#### 68.7.18.1. 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.

#### Prerequisites

- The RAID device has not failed. The following commands will not work if the RAID device has failed.

#### Procedure

- Replace the RAID device:

```
# lvconvert --replace dev_to_remove vg/lv possible_replacements
```

- Replace *dev\_to\_remove* with the path to the physical volume that you want to replace.
- Replace *vg/lv* with the volume group and logical volume name of the RAID array.
- Replace *possible\_replacements* with the path to the physical volume that you want to use as a replacement.

### Example 68.4. Replacing a RAID1 device

The following example creates a RAID1 logical volume and then replaces a device in that volume.

1. Create the RAID1 array:

```
# lvcreate --type raid1 -m 2 -L 1G -n my_lv my_vg

Logical volume "my_lv" created
```

2. Examine the RAID1 array:

```
# 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)
```

3. Replace the **/dev/sdb2** physical volume:

```
# lvconvert --replace /dev/sdb2 my_vg/my_lv
```

4. Examine the RAID1 array with the replacement:

```
# 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)
```

### Example 68.5. Specifying the replacement physical volume

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.

1. Create the RAID1 array:

```
# lvcreate --type raid1 -m 1 -L 100 -n my_lv my_vg

Logical volume "my_lv" created
```

2. Examine the RAID1 array:

■

```
# 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)
```

3. Examine the physical volumes:

```
# 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
```

4. Replace the **/dev/sdb1** physical volume with **/dev/sdd1**:

```
# lvconvert --replace /dev/sdb1 my_vg/my_lv /dev/sdd1
```

5. Examine the RAID1 array with the replacement:

```
# 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)
```

### Example 68.6. Replacing multiple RAID devices

You can replace more than one RAID device at a time by specifying multiple **replace** arguments, as in the following example.

1. Create a RAID1 array:

```
# lvcreate --type raid1 -m 2 -L 100 -n my_lv my_vg

Logical volume "my_lv" created
```

2. Examine the RAID1 array:

```
# 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)
```

3. Replace the **/dev/sdb1** and **/dev/sdc1** physical volumes:

```
# lvconvert --replace /dev/sdb1 --replace /dev/sdc1 my_vg/my_lv
```

4. Examine the RAID1 array with the replacements:

```
# 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)
```

### 68.7.18.2. Failed devices in LVM RAID

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.

### 68.7.18.3. Recovering a failed RAID device in a logical volume

If the LVM RAID device failure is a transient failure or you are able to repair the device that failed, you can initiate recovery of the failed device.

#### Prerequisites

- The previously failed device is now working.

#### Procedure

- Refresh the logical volume that contains the RAID device:

```
# lvchange --refresh my_vg/my_lv
```

#### Verification steps

- Examine the logical volume with the recovered device:

```
# lvs --all --options name,devices,lv_attr,lv_health_status my_vg
```

#### 68.7.18.4. Replacing a failed RAID device in a logical volume

This procedure replaces a failed device that serves as a physical volume in an LVM RAID logical volume.

##### Prerequisites

- The volume group includes a physical volume that provides enough free capacity to replace the failed device.  
If no physical volume with sufficient free extents is available on the volume group, add a new, sufficiently large physical volume using the **vgextend** utility.

##### Procedure

1. In the following example, a RAID logical volume is laid out as follows:

```
# lvs --all --options 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)
```

2. If the **/dev/sdc** device fails, the output of the **lvs** command is as follows:

```
# lvs --all --options name,copy_percent,devices my_vg

/dev/sdc: open failed: No such device or address
Couldn't find device with uuid A4kRI2-vlZA-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)
```

3. 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-vlZA-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.

Optional: To manually specify the physical volume that replaces the failed device, add the physical volume at the end of the command:

```
# lvconvert --repair my_vg/my_lv replacement_pv
```

4. Examine the logical volume with the replacement:

```
# lvs --all --options 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-vlZA-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)
```

Until you remove the failed device from the volume group, LVM utilities still indicate that LVM cannot find the failed device.

5. Remove the failed device from the volume group:

```
# vgreduce --removemissing VG
```

### 68.7.19. 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.

#### Procedure

1. Optional: Limit the I/O bandwidth that the scrubbing process uses.  
When you perform a RAID scrubbing operation, the background I/O required by the **sync** operations can crowd out other I/O to LVM devices, such as updates to volume group metadata. This might cause the other LVM operations to slow down. You can control the rate of the scrubbing operation by implementing recovery throttling.

Add the following options to the **lvchange --syncaction** commands in the next steps:

**--maxrecoveryrate *Rate*[bBsSkKmMgG]**

Sets the maximum recovery rate so that the operation does not crowd out nominal I/O operations. Setting the recovery rate to 0 means that the operation is unbounded.

**--minrecoveryrate *Rate*[bBsSkKmMgG]**

Sets the minimum recovery rate to ensure that I/O for **sync** operations achieves a minimum throughput, even when heavy nominal I/O is present.

Specify the *Rate* value as an amount per second for each device in the array. If you provide no suffix, the options assume kiB per second per device.

2. Display the number of discrepancies in the array, without repairing them:

```
# lvchange --syncaction check vg/raid_lv
```

3. Correct the discrepancies in the array:

```
# lvchange --syncaction repair vg/raid_lv
```



#### NOTE

The **lvchange --syncaction repair** operation does not perform the same function as the **lvconvert --repair** operation:

- The **lvchange --syncaction repair** operation initiates a background synchronization operation on the array.
- The **lvconvert --repair** operation repairs or replaces failed devices in a mirror or RAID logical volume.

4. Optional: Display information about the scrubbing operation:

```
# 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 displays the progress of the **sync** operations.
- The **lv\_attr** field provides additional indicators. Bit 9 of this field displays the health of the logical volume, and it supports the following indicators:

- **m** (mismatches) 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** (refresh) 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. Refresh the logical volume to notify the kernel that the device is now available, or replace the device if you suspect that it failed.

### Additional resources

- For more information, see the **lvchange(8)** and **lvraid(7)** man pages.

## 68.7.20. 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 **lvraid(7)** man page.

## 68.7.21. 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 **lvraid(7)** man page.

## 68.7.22. 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.

## 68.7.23. 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.
```

## 68.8. SNAPSHOT OF LOGICAL VOLUMES

Using the LVM snapshot feature, you can create virtual images of a volume, for example, */dev/sda*, at a particular instant without causing a service interruption.

### 68.8.1. Overview of snapshot volumes

When you modify the original volume (the origin) after you take a snapshot, the snapshot feature makes a copy of the modified data area as it was prior to the change so that it can reconstruct the state of the volume. When you create a snapshot, full read and write access to the origin stays possible.

Since 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. It does not provide a substitute for a backup procedure. Snapshot copies are virtual copies and are not an actual media backup.

The size of the snapshot controls the amount of space set aside for storing the changes to the origin volume. For example, if you create a snapshot and then completely overwrite the origin, the snapshot should be at least as big as the origin volume to hold the changes. You should regularly monitor the size of the snapshot. 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 because it contains many writes, such as **/home**.

If a snapshot is full, the snapshot becomes invalid because it can no longer track changes on the origin volume. But you can configure LVM to automatically extend a snapshot whenever its usage exceeds the **snapshot\_autoextend\_threshold** value to avoid snapshot becoming invalid. Snapshots are fully resizable and you can perform the following operations:

- If you have the storage capacity, you can increase the size of the snapshot volume to prevent it from getting dropped.
- If 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.

The snapshot volume provide the following benefits:

- Most typically, you take a snapshot 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 if the original file system requires file system repair.
- Since the snapshot is read/write, you can test applications against production data by taking a snapshot and running tests against the snapshot without touching the real data.
- You can create LVM volumes for use with Red Hat Virtualization. You can use LVM snapshots 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.

## 68.8.2. Creating a snapshot of the original volume

Use **lvcreate** command with the **-s** or **--size** argument followed by the required size to create a snapshot of the original volume (the origin). A snapshot of a volume is writable. By default, a snapshot volume is activated with the origin during normal activation commands as compared to the thinly-provisioned snapshots. LVM does not support creating a snapshot volume that is larger than the sum of the origin volume's size and the required metadata size for the volume. If you specify a snapshot volume that is larger than this, LVM creates a snapshot volume that is required for the size of the origin.



### NOTE

The nodes in a cluster do not support LVM snapshots. 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.

The following procedure creates an origin logical volume named *origin* and a snapshot volume of this original volume named *snap*.

### Prerequisites

- You have created volume group *vg001*. For more information, see [Creating LVM volume group](#).

### Procedure

- Create a logical volume named *origin* from the volume group *vg001*:

```
# lvcreate -L 1G -n origin vg001
Logical volume "origin" created.
```

- Create a snapshot logical volume named *snap* of */dev/vg001/origin* that is 100 MB in size:

```
# lvcreate --size 100M --name snap --snapshot /dev/vg001/origin
Logical volume "snap" created.
```

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.

- Display the origin volume and the current percentage of the snapshot volume being used:

```
# lvs -a -o +devices
LV VG Attr LSize Pool Origin Data% Meta% Move Log Cpy%Sync Convert
Devices
origin vg001 owi-a-s--- 1.00g /dev/sde1(0)
snap vg001 swi-a-s--- 100.00m origin 0.00 /dev/sde1(256)
```

You can also display the status of logical volume */dev/vg001/origin* with all the snapshot logical volumes and their status, such as active or inactive by using the **lvdisplay /dev/vg001/origin** command.

**WARNING**

Since 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 become full. 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.

- You can configure LVM to automatically extend a snapshot when its usage exceeds the **snapshot\_autoextend\_threshold** value to avoid the snapshot becoming invalid when it is 100% full. View the existing values for the **snapshot\_autoextend\_threshold** and **snapshot\_autoextend\_percent** options from the `/etc/lvm.conf` file and edit them as per your requirements.

The following example, sets the **snapshot\_autoextend\_threshold** option to value less than 100 and **snapshot\_autoextend\_percent** option to the value depending on your requirement to extend the snapshot volume:

```
# vi /etc/lvm.conf
snapshot_autoextend_threshold = 70
snapshot_autoextend_percent = 20
```

You can also extend this snapshot manually by executing the following command:

```
# lvextend -L+100M /dev/vg001/snap
```

**NOTE**

This feature requires unallocated space in the volume group. An automatic extension of a snapshot does 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.

**Additional resources**

- lvcreate(8)**, **lvextend(8)**, and **lvs(8)** man pages
- `/etc/lvm/lvm.conf` file

**68.8.3. Merging snapshot to its original volume**

Use the **lvconvert** command with the **--merge** option to merge a snapshot into its original (the origin) volume. You can perform a system rollback if you have lost data or files, or otherwise you have to restore your system to a previous state. After you merge the snapshot volume, the resulting logical volume has the origin volume'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.

If both the origin and snapshot volume are not open and active, the merge starts immediately.

Otherwise, the merge starts after either the origin or snapshot are activated and both are closed. You can merge a snapshot into an origin that cannot be closed, for example a **root** file system, after the origin volume is activated.

### Procedure

1. Merge the snapshot volume. The following command merges snapshot volume `vg001/snap` into its *origin*:

```
# lvconvert --merge vg001/snap
Merging of volume vg001/snap started.
vg001/origin: Merged: 100.00%
```

2. View the origin volume:

```
# lvs -a -o +devices
LV   VG   Attr   LSize Pool Origin Data% Meta% Move Log Cpy%Sync Convert
Devices
origin vg001 owi-a-s--- 1.00g                /dev/sde1(0)
```

### Additional resources

- **lvconvert(8)** man page

## 68.9. CREATING AND MANAGING THIN PROVISIONED VOLUMES (THIN VOLUMES)

Red Hat Enterprise Linux supports thin provisioned snapshot volumes and logical volumes.

Logical volumes and snapshot volumes can be thinly provisioned:

- Using thin-provisioned logical volumes, you can create logical volumes that are larger than the available physical storage.
- Using thin-provisioned snapshot volumes, you can store more virtual devices on the same data volume.

### 68.9.1. Overview of thin provisioning

Many modern storage stacks now provide the ability to choose between thick provisioning and thin provisioning:

- Thick provisioning provides the traditional behavior of block storage where blocks are allocated regardless of their actual usage.
- Thin provisioning grants the ability to provision a larger pool of block storage that may be larger in size than the physical device storing the data, resulting in over-provisioning. Over-provisioning is possible because individual blocks are not allocated until they are actually used. If you have multiple thin-provisioned devices that share the same pool, then these devices can be over-provisioned.

By using thin provisioning, you can over-commit the physical storage, and instead can manage a pool of free space known as a thin pool. You can allocate this thin pool to an arbitrary number of devices when needed by applications. You can expand the thin pool dynamically when needed for cost-effective

allocation of storage space.

For example, if ten users each request a 100GB file system for their application, then you 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.



## NOTE

When using thin provisioning, it is important that you monitor the storage pool and add more capacity as the available physical space runs out.

The following are a few advantages of using thin-provisioned devices:

- You can create logical volumes that are larger than the available physical storage.
- You can have more virtual devices to be stored on the same data volume.
- You can create file systems that can grow logically and automatically to support the data requirements and the unused blocks are returned to the pool for use by any file system in the pool

The following are the potential drawbacks of using thin-provisioned devices:

- Thin-provisioned volumes have an inherent risk of running out of available physical storage. If you have over-provisioned your underlying storage, it could possibly result in an outage due to the lack of available physical storage. For example, if you create 10T of thinly provisioned storage with only 1T physical storage for backing, the volumes will become unavailable or unwritable after the 1T is exhausted.
- If volumes are not sending discards to the layers after thin-provisioned devices, then the accounting for usage will not be accurate. For example, placing a file system without the **-o discard mount** option and not running **fstrim** periodically on top of thin-provisioned devices will never unallocate previously used storage. In such cases, you end up using the full provisioned amount over time even if you are not really using it.
- You must monitor the logical and physical usage so as to not run out of available physical space.
- Copy on Write (CoW) operation can be slower on file systems with snapshots.
- Data blocks can be intermixed between multiple file systems leading to random access limitations of the underlying storage even when it does not appear that way to the end user.

### 68.9.2. Creating thinly-provisioned logical volumes

Using thin-provisioned logical volumes, you can create logical volumes that are larger than the available physical storage. Creating a thinly provisioned set of volumes allows the system to allocate what you use instead of allocating the full amount of storage that is requested.

Using the **-T** or **--thin** option of the **lvcreate** command, you can create either a thin pool or a thin volume. You can also use the **-T** option of the **lvcreate** command to create both a thin pool and a thin volume at the same time with a single command. This procedure describes how to create and grow thinly-provisioned logical volumes.

#### Prerequisites

- You have created a volume group. For more information, see [Creating LVM volume group](#).

## Procedure

1. Create a thin pool:

```
# lvcreate -L 100M -T vg001/mythinpool
Thin pool volume with chunk size 64.00 KiB can address at most 15.81 TiB of data.
Logical volume "mythinpool" created.
```

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 determines what type of device is to be created from the other options that are added with the command. You can also create thin pool using additional parameters as shown in the following examples:

- You can also create a thin pool using the **--thinpool** parameter of the **lvcreate** command. Unlike the **-T** option, the **--thinpool** parameter requires that you specify the name of the thin pool logical volume you are creating. The following example uses the **--thinpool** parameter to create a thin pool named *mythinpool* in the volume group *vg001* that is *100M* in size:

```
# lvcreate -L 100M --thinpool mythinpool vg001
Thin pool volume with chunk size 64.00 KiB can address at most 15.81 TiB of data.
Logical volume "mythinpool" created.
```

- As striping is supported for pool creation, you can use the **-i** and **-l** options to create stripes. The following command creates a *100M* thin pool named as *thinpool* in volume group *vg001* with two *64 kB* stripes and a chunk size of *256 kB*. It also creates a *1T* thin volume named *vg001/thinvolume*.



### NOTE

Ensure that there are two physical volumes with sufficient free space in the volume group or you cannot create the thin pool.

```
# lvcreate -i 2 -l 64 -c 256 -L 100M -T vg001/thinpool -V 1T --name thinvolume
```

2. Create a thin volume:

```
# lvcreate -V 1G -T vg001/mythinpool -n thinvolume
WARNING: Sum of all thin volume sizes (1.00 GiB) exceeds the size of thin pool
vg001/mythinpool (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.
```

In this case, you are specifying virtual size for the volume that is greater than the pool that contains it. You can also create thin volumes using additional parameters as shown in the following examples:

- To create both a thin volume and a thin pool, use the **-T** option of the **lvcreate** command and specify both the size and virtual size argument:

```
# lvcreate -L 100M -T vg001/mythinpool -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/mythinpool (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.
```

- To use the remaining free space to create a thin volume and thin pool, use the **100%FREE** option:

```
# lvcreate -V 1G -l 100%FREE -T vg001/mythinpool -n thinvolume
```

```
Thin pool volume with chunk size 64.00 KiB can address at most <15.88 TiB of data.
```

```
Logical volume "thinvolume" created.
```

- To convert an existing logical volume to a thin pool volume, use the **--thinpool** parameter of the **lvconvert** command. You must also use the **--poolmetadata** parameter in conjunction with the **--thinpool** parameter to convert an existing logical volume to a thin pool volume's metadata volume.

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.
```



#### NOTE

Converting a logical volume to a thin pool volume or a thin pool metadata volume destroys the content of the logical volume, as **lvconvert** does not preserve the content of the devices but instead overwrites the content.

- By default, the **lvcreate** command approximately sets the size of the thin pool metadata logical volume by using the following formula:

```
Pool_LV_size / Pool_LV_chunk_size * 64
```

If you have large numbers of snapshots or if you have small chunk sizes for your thin pool and therefore 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 using 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.

The following example illustrates how to increase the default value of the thin pools' metadata volume:

```
# lvcreate -V 1G -l 100%FREE -T vg001/mythinpool --poolmetadatasize 16M -n
thinvolume
```

```
Thin pool volume with chunk size 64.00 KiB can address at most 15.81 TiB of data.
```

```
Logical volume "thinvolume" created.
```

3. View the created thin pool and thin volume:

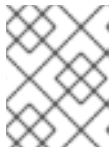
```
# lvs -a -o +devices
```

```

LV          VG  Attr   LSize  Pool      Origin Data%  Meta%  Move Log Cpy%Sync
Convert Devices
[lvol0_pmspare]  vg001 ewi----- 4.00m                               /dev/sda(0)
mythinpool      vg001 twi-aotz-- 100.00m          0.00  10.94
mythinpool_tdata(0)
[mythinpool_tdata] vg001 Twi-ao---- 100.00m
/dev/sda(1)
[mythinpool_tmeta] vg001 ewi-ao---- 4.00m
/dev/sda(26)
thinvolume      vg001 Vwi-a-tz-- 1.00g mythinpool 0.00

```

- Optional: Extend the size of a thin pool with the **lvextend** command. You cannot, however, reduce the size of a thin pool.



#### NOTE

This command fails if you use **-l 100%FREE** argument while creating a thin pool and thin volume.

The following command resizes an existing thin pool that is *100M* in size by extending it another *100M*:

```

# lvextend -L+100M vg001/mythinpool
Size of logical volume vg001/mythinpool_tdata changed from 100.00 MiB (25 extents) to
200.00 MiB (50 extents).
WARNING: Sum of all thin volume sizes (1.00 GiB) exceeds the size of thin pool
vg001/mythinpool (200.00 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 *vg001/mythinpool* successfully resized

```

# lvs -a -o +devices
LV          VG  Attr   LSize  Pool      Origin Data%  Meta%  Move Log Cpy%Sync
Convert Devices
[lvol0_pmspare]  vg001 ewi----- 4.00m                               /dev/sda(0)
mythinpool      vg001 twi-aotz-- 200.00m          0.00  10.94
mythinpool_tdata(0)
[mythinpool_tdata] vg001 Twi-ao---- 200.00m
/dev/sda(1)
[mythinpool_tdata] vg001 Twi-ao---- 200.00m
/dev/sda(27)
[mythinpool_tmeta] vg001 ewi-ao---- 4.00m
/dev/sda(26)
thinvolume      vg001 Vwi-a-tz-- 1.00g mythinpool 0.00

```

- Optional: To rename the thin pool and thin volume, use the following command:

```

# lvrename vg001/mythinpool vg001/mythinpool1
Renamed "mythinpool" to "mythinpool1" in volume group "vg001"

# lvrename vg001/thinvolume vg001/thinvolume1
Renamed "thinvolume" to "thinvolume1" in volume group "vg001"

```



View the thin pool and thin volume after renaming:

```
# lvs
LV      VG      Attr  LSize  Pool      Origin Data%  Move Log Copy%  Convert
mythinpool1 vg001  twi-a-tz 100.00m          0.00
thinvolume1 vg001  Vwi-a-tz 1.00g mythinpool1 0.00
```

6. Optional: To remove the thin pool, use the following command:

```
# lvremove -f vg001/mythinpool1
Logical volume "thinvolume1" successfully removed.
Logical volume "mythinpool1" successfully removed.
```

### Additional resources

- **lvcreate(8)**, **lvrename(8)**, **lvs(8)**, and **lvconvert(8)** man pages

### 68.9.3. Overview of chunk size

A chunk is the largest unit of physical disk dedicated to snapshot storage.

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 estimates good metadata size for such chunk size. The minimal metadata size **lvm2** can create and use is 2 MiB. If the metadata size needs to be larger than 128 MiB it begins to increase the chunk size, so the metadata size stays 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 bigger metadata size is a better option.

To specify the chunk size according to your requirement, use the **-c** or **--chunksize** parameter to overrule **lvm2** estimated chunk size. Be aware that you cannot change the chunk size once the thinpool is created.

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 according to 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.



#### NOTE

Using the inappropriate combination of chunk size and metadata size may result in a potentially problematic situation, when user runs out of space in **metadata** or they may not further grow their thin-pool size because of limited maximum addressable thin-pool data size.

### Additional resources

- **lvmthin(7)** man page

## 68.9.4. Thinly-provisioned snapshot volumes

Red Hat Enterprise Linux supports thinly-provisioned snapshot volumes. A snapshot of a thin logical volume also creates a thin logical volume (LV). 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.



### NOTE

Similarly to all LVM snapshot volumes, and 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.

Traditional snapshots must allocate new space for each snapshot created, where data is preserved as changes are made to the origin. But thin-provisioning snapshots share the same space with the origin. Snapshots of thin LVs are efficient because the data blocks common to a thin LV and any of its snapshots are shared. You can create snapshots of thin LVs or from the other thin snapshots. Blocks common to recursive snapshots are also shared in the thin pool.

Thin snapshot volumes provide the following benefits:

- Increasing the number of snapshots of the origin has a negligible impact on performance.
- A thin snapshot volume can reduce disk usage because only the new data is written and is not copied to each snapshot.
- There is no need to simultaneously activate the thin snapshot volume with the origin, which is a requirement of traditional snapshots.
- When restoring an origin from a snapshot, it is not required to merge the thin snapshot. You can remove the origin and instead use the snapshot. Traditional snapshots have a separate volume where they store changes that must be copied back, that is, merged to the origin to reset it.
- There is a significantly higher limit on the number of allowed snapshots as compared to the traditional snapshots.

Although there are many advantages for using thin snapshot volumes, there are some use cases for which the traditional LVM snapshot volume feature might be more appropriate to your needs. You can use traditional snapshots with all types of volumes. However, to use thin-snapshots requires you to use thin-provisioning.



### NOTE

You cannot limit the size of a thin snapshot volume; the snapshot uses all of the space in the thin pool, if necessary. In general, you should consider the specific requirements of your site when deciding which snapshot format to use.

By default, a thin snapshot volume is skipped during normal activation commands.

## 68.9.5. Creating thinly-provisioned snapshot volumes

Using thin-provisioned snapshot volumes, you can have more virtual devices stored on the same data volume.



## IMPORTANT

When creating a thin snapshot volume, 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. The following procedure describes different ways to create a thinly-provisioned snapshot volume.

### Prerequisites

- You have created a thinly-provisioned logical volume. For more information, see [Overview of thin provisioning](#).

### Procedure

- Create a thinly-provisioned snapshot volume. The following command creates a thinly-provisioned snapshot volume named as *mynsnapshot1* of the thinly-provisioned logical volume *vg001/thinvolume*:

```
# lvcreate -s --name mynsnapshot1 vg001/thinvolume
Logical volume "mynsnapshot1" created
```

```
# lvs
LV      VG      Attr  LSize Pool   Origin  Data% Move Log Copy% Convert
mynsnapshot1 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
```



## NOTE

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. For information on extending the size of a thin volume, see [Creating thinly-provisioned logical volumes](#).

- 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. The following example creates a thin snapshot volume of the read-only, inactive logical volume named *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 uses the existing thin pool *vg001/pool*. The origin volume must be in the same volume group as the snapshot volume. Do not 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 by executing the following command.

```
# lvcreate -s vg001/mysnapshot1 --name mysnapshot2
Logical volume "mysnapshot2" created.
```

To create a third thinly-provisioned snapshot volume, use the following command:

```
# lvcreate -s vg001/mysnapshot2 --name mysnapshot3
Logical volume "mysnapshot3" created.
```

## Verification

- Display a list of all ancestors and descendants of a thin snapshot logical volume:

```
$ lvs -o name,lv_ancestors,lv_descendants vg001
LV      Ancestors          Descendants
mysnapshot2  mysnapshot1,thinvolume  mysnapshot3
mysnapshot1  thinvolume             mysnapshot2,mysnapshot3
mysnapshot3  mysnapshot2,mysnapshot1,thinvolume
mythinpool
thinvolume                mysnapshot1,mysnapshot2,mysnapshot3
```

Here,

- thinvolume* is an origin volume in volume group *vg001*.
- mysnapshot1* is a snapshot of *thinvolume*
- mysnapshot2* is a snapshot of *mysnapshot1*
- mysnapshot3* is a snapshot of *mysnapshot2*



### NOTE

The **lv\_ancestors** and **lv\_descendants** fields display existing dependencies. However, they do not track removed entries which can break a dependency chain if the entry was removed from the middle of the chain.

## Additional resources

- lvcreate(8)** man page

## 68.10. 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.

### 68.10.1. 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**.

### **Additional resources**

- **lvmcache(7)** man page

## **68.10.2. LVM caching components**

LVM provides support for adding a cache to LVM logical volumes. LVM caching uses the following LVM logical volume types:

### **Main LV**

The larger, slower, and original volume.

### **Cache pool LV**

A composite LV that you can use for caching data from the main LV. It has two sub-LVs: data for holding cache data and metadata for managing the cache data. You can configure specific disks for data and metadata. You can use the cache pool only with **dm-cache**.

### **Cachevol LV**

A linear LV that you can use for caching data from the main LV. You cannot configure separate disks for data and metadata. **cachevol** can be only used with either **dm-cache** or **dm-writecache**.

All of these associated LVs must be in the same volume group.

You can combine a main logical volume (LV) with a faster, usually smaller, LV that holds the cached data. The fast LV is created from fast block devices, such as SSD drives. 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.

### Additional resources

- [lvmcache\(7\)](#) man page
- [Creating and managing thin provisioned volumes \(thin volumes\)](#)

### 68.10.3. 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/sdf**

#### Example 68.7. Creating a **cachevol** volume

```
# lvcreate --size 5G --name fastvol vg /dev/sdf
Logical volume "fastvol" created.
```

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

**Example 68.8. Attaching the cachevol volume to the main LV**

```
# lvconvert --type cache --cachevol fastvol vg/main-lv
Erase all existing data on vg/fastvol? [y/n]: y
Logical volume vg/main-lv is now cached.
```

**Verification steps**

- Verify if the newly created logical volume has **dm-cache** enabled:

```
# 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**

- **lvmdcache(7)** man page

**68.10.4. Enabling dm-cache caching with a cachepool for a logical volume**

This procedure enables you to create the cache data and the cache metadata logical volumes individually and then combine the volumes into a cache pool.

**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 **cachepool** volume on the fast device:

```
# lvcreate --type cache-pool --size <cachepool-size> --name <fastpool> <vg /dev/fast>
```

Replace the following values:

**cachepool-size**

The size of the **cachepool**, such as **5G**

**fastpool**

A name for the **cachepool** volume

**vg**

The volume group name

**/dev/fast**The path to the fast block device, such as **/dev/sdf1****NOTE**

You can use **--poolmetadata** option to specify the location of the pool metadata when creating the cache-pool.

**Example 68.9. Creating a cachepool volume**

```
# lvcreate --type cache-pool --size 5G --name fastpool vg /dev/sde
Logical volume "fastpool" created.
```

2. Attach the **cachepool** to the main logical volume to begin caching:

```
# lvconvert --type cache --cachepool <fastpool> <vg/main>
```

Replace the following values:

**fastpool**The name of the **cachepool** volume**vg**

The volume group name

**main**

The name of the slow logical volume

**Example 68.10. Attaching the cachepool to the main LV**

```
# lvconvert --type cache --cachepool fastpool vg/main
Do you want wipe existing metadata of cache pool vg/fastpool? [y/n]: y
Logical volume vg/main is now cached.
```

**Verification steps**

- Examine the newly created devicevolume with the **cache-pool** type:

```
# lvs --all --options +devices <vg>
```

LV	Pool	Type	Devices
[fastpool_cpool]		cache-pool	fastpool_pool_cdata(0)
[fastpool_cpool_cdata]		linear	/dev/sdf1(4)
[fastpool_cpool_cmeta]		linear	/dev/sdf1(2)
[lv00_pmspare]		linear	/dev/sdf1(0)
main	[fastpool_cpool]	cache	main_corig(0)
[main_corig]		linear	/dev/sdf1(0)



-

### Additional resources

- **lvcreate(8)** man page
- **lvmcache(7)** man page
- **lvconvert(8)** man page

### 68.10.5. 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.
- If the slow logical volume is active, deactivate it.

#### 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/sdf**

**Example 68.11. Creating a deactivated cachevol volume**

```
# lvcreate --activate n --size 5G --name fastvol vg /dev/sdf
WARNING: Logical volume vg/fastvol not zeroed.
Logical volume "fastvol" created.
```

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

#### **Example 68.12. Attaching the cachevol volume to the main LV**

```
# lvconvert --type writecache --cachevol fastvol vg/main-lv
Erase all existing data on vg/fastvol? [y/n]?: y
Using writecache block size 4096 for unknown file system block size, logical block
size 512, physical block size 512.
WARNING: unable to detect a file system block size on vg/main-lv
WARNING: using a writecache block size larger than the file system block size may
corrupt the file system.
Use writecache block size 4096? [y/n]: y
Logical volume vg/main-lv now has writecache.
```

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_cvool] [main-lv_wcorig] 0.00
main-lv_wcorig(0)
[fastvol_cvool] vg Cwi-aoC--- 252.00m
/dev/sdc1(0)
[main-lv_wcorig] vg owi-aoC--- 500.00m
/dev/sdb1(0)

```

### Additional resources

- **lvmcache(7)** man page

### 68.10.6. 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 *vg* with the volume group name, and *main-lv* with 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:

Replace *vg* with the volume group name, and *main-lv* with the name of the logical volume where caching is enabled.

#### Example 68.13. Detaching the **cachevol** or **cachepool** volume

```

# lvconvert --splitcache vg/main-lv
Detaching writecache already clean.
Logical volume vg/main-lv writecache has been detached.

```

#### Verification steps

- Check that the logical volumes are no longer attached together:

```

# lvs --all --options +devices <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

## 68.11. 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 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
```



### NOTE

During manual activation, the **systemd** automatically mounts LVM volumes with the corresponding mount point from the **/etc/fstab** file unless the **systemd-mount** unit is masked.

### 68.11.1. 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.

### 68.11.2. 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
```

### 68.11.3. 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.

### 68.11.4. 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 **lvraid(7)** man page.

## 68.12. LIMITING LVM DEVICE VISIBILITY AND USAGE

You can limit the devices that are visible and usable to Logical Volume Manager (LVM) by controlling the devices that LVM can scan.

To adjust the configuration of LVM device scanning, edit the LVM device filter settings in the `/etc/lvm/lvm.conf` file. The filters in the `lvm.conf` file consist of a series of simple regular expressions. The system applies these expressions to each device name in the `/dev` directory to decide whether to accept or reject each detected block device.

### 68.12.1. The LVM device filter

The Logical Volume Manager (LVM) device filter is a list of device name patterns.

Patterns are regular expressions delimited by any character and preceded by **a** for accepting, or **r** for rejecting. The first regular expression in the list that matches a device determines if LVM accepts or rejects (ignores) a specific device. A device can have several names through symlinks. If the filter accepts any one of those device names, LVM uses the device. LVM also accepts devices that do not match any patterns.

The default device filter accepts all devices on the system. An ideal user configured device filter accepts one or more patterns and rejects everything else. For example, in such cases, the pattern list can end with `r|.*`.

You can find the LVM devices filter configuration in the `devices/filter` and `devices/global_filter` fields in the `lvm.conf` file.

#### 68.12.1.1. Additional resources

- `lvm.conf(5)` man page

#### 68.12.1.2. Examples of LVM device filter configurations

The list below shows filter configurations that control which devices LVM scans and can later use. Configure the device filter in the `lvm.conf` file.

- The following is the default filter configuration, which scans all devices:

```
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 Integrated Development Environment (IDE) devices and removes all other block devices:

```
filter = [ "a|loop|", "a|/dev/hd.*|", "r|.*" ]
```

- The following filter adds only partition 8 on the first IDE drive and removes all other block devices:

```
filter = [ "a|^/dev/hda8$", "r|.*)" ]
```

## Additional resources

- **lvm.conf(5)** man page

## 68.13. 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.

### 68.13.1. 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**

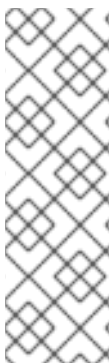


configuration parameter is set to 0, it will then allow the log to share physical volume(s) with part of the data.

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

Be aware that future updates can bring code changes in layout behaviour according to the defined allocation policies. 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.

### 68.13.2. 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.

### 68.13.3. 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)
```

### 68.13.4. Differentiating between LVM RAID objects using tags

You can assign tags to LVM RAID objects to group them, so that you can automate the control of LVM RAID behavior, such as activation, by group.

The physical volume (PV) tags are responsible for the allocation control in the LVM raid, as opposed to logical volume (LV) or volume group (VG) tags, because allocation in lvm occurs at the PV level based on allocation policies. To distinguish storage types by their different properties, tag them appropriately (e.g. NVMe, SSD, HDD). Red Hat recommends that you tag each new PV appropriately after you add it to a VG.

This procedure adds object tags to your logical volumes, assuming `/dev/sda` is an SSD, and `/dev/sd[b-f]` are HDDs with one partition.

#### Prerequisites

- The **lvm2** package is installed.
- Storage devices to use as PVs are available.

#### Procedure

1. Create a volume group.

```
# vgcreate MyVG /dev/sd[a-f]1
```

2. Add tags to your physical volumes.

```
# pvchange --addtag ssds /dev/sda1
# pvchange --addtag hdds /dev/sd[b-f]1
```

3. Create a RAID6 logical volume.

```
# lvcreate --type raid6 --stripes 3 -L1G -nr6 MyVG @hdds
```

4. Create a linear cache pool volume.

```
# lvcreate -nr6pool -L512m MyVG @ssds
```

5. Convert the RAID6 volume to be cached.

```
# lvconvert --type cache --cachevol MyVG/r6pool MyVG/r6
```

#### Additional resources

- The **lvcreate(8)**, **lvconvert(8)**, **lvmraid(7)** and **lvmcache(7)** man pages.

## 68.14. TROUBLESHOOTING LVM

You can use LVM tools to troubleshoot a variety of issues in LVM volumes and groups.

### 68.14.1. 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 **-v** argument to any LVM command to increase the verbosity level of the command output. Verbosity can be further increased by adding additional **v's**. A maximum of four such **v's** is allowed, for example, **-vvvv**.
  - 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. Execute 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

- **lvmdump(8)** man page

### 68.14.2. 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 68.14. Failed volume groups

In this example, one of the devices that made up the volume group *myvg* failed. The volume group is unusable but you can see information about the failed device.

```
# vgs --options +devices
/dev/vdb1: open failed: No such device or address
/dev/vdb1: open failed: No such device or address
WARNING: Couldn't find device with uuid 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-
z1lf4s.
WARNING: VG myvg is missing PV 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s (last
written to /dev/sdb1).
WARNING: Couldn't find all devices for LV myvg/mylv while checking used and assumed
devices.

VG  #PV #LV #SN Attr  VSize VFree Devices
myvg 2  2  0 wz-pn- <3.64t <3.60t [unknown](0)
myvg 2  2  0 wz-pn- <3.64t <3.60t [unknown](5120),/dev/vdb1(0)
```

#### Example 68.15. Failed logical volume

In this example, one of the devices failed due to which the logical volume in the volume group failed. The command output shows the failed logical volumes.

```
# lvs --all --options +devices

/dev/vdb1: open failed: No such device or address
/dev/vdb1: open failed: No such device or address
WARNING: Couldn't find device with uuid 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-
z1lf4s.
WARNING: VG myvg is missing PV 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s (last
written to /dev/sdb1).
WARNING: Couldn't find all devices for LV myvg/mylv while checking used and assumed
devices.

LV  VG Attr  LSize Pool Origin Data% Meta% Move Log Cpy%Sync Convert
Devices
mylv myvg -wi-a---p- 20.00g                [unknown](0)
[unknown](5120),/dev/sdc1(0)
```

**Example 68.16. 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)
```

**68.14.3. 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 myvg
```

2. Check which logical volumes will be removed:

```
# vgreduce --removemissing --test myvg
```

3. Remove all the logical volumes that used the lost physical volume from the volume group:

```
# vgreduce --removemissing --force myvg
```

4. Optional: If you accidentally removed logical volumes that you wanted to keep, you can reverse the **vgreduce** operation:

```
# vgcfgrestore myvg
```

**WARNING**

If you remove a thin pool, LVM cannot reverse the operation.

#### 68.14.4. Finding the metadata of a missing LVM physical volume

If the volume group's 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.

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:

```
# cat /etc/lvm/archive/myvg_00000-1248998876.vg
```

Replace `00000-1248998876` with the backup-number. 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-iVI8-7qTD-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 myvg
```

```
PARTIAL MODE. Incomplete logical volumes will be processed.
```

```
WARNING: Couldn't find device with uuid 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s.
```

```
WARNING: VG myvg is missing PV 42B7bu-YCMp-CEVD-CmKH-2rk6-fiO9-z1lf4s (last written to /dev/vdb1).
```

```
0 logical volume(s) in volume group "myvg" now active
```

#### 68.14.5. 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. You might be able to recover the data from the physical volume by rewriting the metadata area on the physical volume.

**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 [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 68.17. Restoring a physical volume on `/dev/vdb1`**

The following example labels the `/dev/vdb1` 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/vdb1
```

...

Physical volume `"/dev/vdb1"` successfully created

- Restore the metadata of the volume group:

```
# vgcfgrestore myvg

Restored volume group myvg
```

- Display the logical volumes on the volume group:



```
# lvs --all --options +devices myvg
```

The logical volumes are currently inactive. For example:

```
LV   VG   Attr LSize  Origin Snap%  Move Log Copy%  Devices
mylv myvg -wi--- 300.00G                /dev/vdb1 (0),/dev/vdb1(0)
mylv myvg -wi--- 300.00G                /dev/vdb1 (34728),/dev/vdb1(0)
```

- If the segment type of the logical volumes is RAID, resynchronize the logical volumes:

```
# lvchange --resync myvg/mylv
```

- Activate the logical volumes:

```
# lvchange --activate y myvg/mylv
```

- 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
mylv myvg -wi--- 300.00G                /dev/vdb1 (0),/dev/vdb1(0)
mylv myvg -wi--- 300.00G                /dev/vdb1 (34728),/dev/vdb1(0)
```

### 68.14.6. 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.

### 68.14.7. Preventing the rounding error when creating an LVM volume

When creating an LVM logical volume, you can specify the number of logical extents of the logical volume to avoid rounding error.

#### Procedure

1. Find the number of free physical extents in the volume group:

```
# vdisplay myvg
```

#### Example 68.18. Free extents in a volume group

For example, the following volume group has 8780 free physical extents:

```
--- Volume group ---
VG Name          myvg
System ID
Format           lvm2
Metadata Areas   4
Metadata Sequence No 6
VG Access        read/write
[...]
Free PE / Size   8780 / 34.30 GB
```

2. Create the logical volume. Enter the volume size in extents rather than bytes.

#### Example 68.19. Creating a logical volume by specifying the number of extents

```
# lvcreate --extents 8780 --name mylv myvg
```

#### Example 68.20. Creating a logical volume to occupy all the remaining space

Alternatively, 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 mylv myvg
```

### 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
myvg 2  1  0 wz--n- 34.30G  0  0  8780
```

## 68.14.8. Troubleshooting LVM RAID

You can troubleshoot various issues in LVM RAID devices to correct data errors, recover devices, or replace failed devices.

### 68.14.8.1. 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.

## Procedure

- Optional: Limit the I/O bandwidth that the scrubbing process uses.  
When you perform a RAID scrubbing operation, the background I/O required by the **sync** operations can crowd out other I/O to LVM devices, such as updates to volume group metadata. This might cause the other LVM operations to slow down. You can control the rate of the scrubbing operation by implementing recovery throttling.

Add the following options to the **lvchange --syncaction** commands in the next steps:

### **--maxrecoveryrate** *Rate*[bBsSkKmMgG]

Sets the maximum recovery rate so that the operation does not crowd out nominal I/O operations. Setting the recovery rate to 0 means that the operation is unbounded.

### **--minrecoveryrate** *Rate*[bBsSkKmMgG]

Sets the minimum recovery rate to ensure that I/O for **sync** operations achieves a minimum throughput, even when heavy nominal I/O is present.

Specify the *Rate* value as an amount per second for each device in the array. If you provide no suffix, the options assume kiB per second per device.

- Display the number of discrepancies in the array, without repairing them:

```
# lvchange --syncaction check vg/raid_lv
```

- Correct the discrepancies in the array:

```
# lvchange --syncaction repair vg/raid_lv
```



## NOTE

The **lvchange --syncaction repair** operation does not perform the same function as the **lvconvert --repair** operation:

- The **lvchange --syncaction repair** operation initiates a background synchronization operation on the array.
- The **lvconvert --repair** operation repairs or replaces failed devices in a mirror or RAID logical volume.

- Optional: Display information about the scrubbing operation:

```
# 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 displays the progress of the **sync** operations.
- The **lv\_attr** field provides additional indicators. Bit 9 of this field displays the health of the logical volume, and it supports the following indicators:
  - **m** (mismatches) 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** (refresh) 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. Refresh the logical volume to notify the kernel that the device is now available, or replace the device if you suspect that it failed.

**Additional resources**

- For more information, see the **lvchange(8)** and **lvraid(7)** man pages.

**68.14.8.2. Failed devices in LVM RAID**

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.

**68.14.8.3. Recovering a failed RAID device in a logical volume**

If the LVM RAID device failure is a transient failure or you are able to repair the device that failed, you can initiate recovery of the failed device.

**Prerequisites**

- The previously failed device is now working.

**Procedure**

- Refresh the logical volume that contains the RAID device:

```
# lvchange --refresh my_vg/my_lv
```

**Verification steps**

- Examine the logical volume with the recovered device:

```
# lvs --all --options name,devices,lv_attr,lv_health_status my_vg
```

#### 68.14.8.4. Replacing a failed RAID device in a logical volume

This procedure replaces a failed device that serves as a physical volume in an LVM RAID logical volume.

##### Prerequisites

- The volume group includes a physical volume that provides enough free capacity to replace the failed device.

If no physical volume with sufficient free extents is available on the volume group, add a new, sufficiently large physical volume using the **vgextend** utility.

##### Procedure

1. In the following example, a RAID logical volume is laid out as follows:

```
# lvs --all --options 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)
```

2. If the **/dev/sdc** device fails, the output of the **lvs** command is as follows:

```
# lvs --all --options name,copy_percent,devices my_vg

/dev/sdc: open failed: No such device or address
Couldn't find device with uuid A4kRI2-vlzA-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)
```

3. 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-vlZA-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.

```

Optional: To manually specify the physical volume that replaces the failed device, add the physical volume at the end of the command:

```
# lvconvert --repair my_vg/my_lv replacement_pv
```

4. Examine the logical volume with the replacement:

```

# lvs --all --options 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-vlZA-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)

```

Until you remove the failed device from the volume group, LVM utilities still indicate that LVM cannot find the failed device.

5. Remove the failed device from the volume group:

```
# vgreduce --removemissing VG
```

### 68.14.9. 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/sddlmap not /dev/sdf

```

You can troubleshoot these warnings to understand why LVM displays them, or to hide the warnings.

#### 68.14.9.1. 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 68.6. Example device mappings in different multipath software**

Multipath software	SCSI paths to a LUN	Multipath device mapping to paths
DM Multipath	<code>/dev/sdb</code> and <code>/dev/sdc</code>	<code>/dev/mapper/mpath1</code> or <code>/dev/mapper/mpatha</code>
EMC PowerPath		<code>/dev/emcpowera</code>
HDLM		<code>/dev/sddlmap</code>

As a result of the multiple device nodes, LVM tools find the same metadata multiple times and report them as duplicates.

### 68.14.9.2. Cases of duplicate PV warnings

LVM displays the duplicate PV warnings in either of the following cases:

#### Single paths to the same device

The two devices displayed in the output are both 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 two devices displayed in the output are both 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.

### 68.14.9.3. 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|.*)" ]
```

### 68.14.9.4. 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
```

#### 68.14.9.5. Additional resources

- [Limiting LVM device visibility and usage](#)
- [The LVM device filter](#)