Red Hat Enterprise Linux 9

Performing an advanced RHEL 9 installation

Installing RHEL using Kickstart

Red Hat Enterprise Linux 9 Performing an advanced RHEL 9 installation

Installing RHEL using Kickstart

Legal Notice

Abstract

You can perform an advanced RHEL installation by using Kickstart. Kickstart helps you to automate the installation with a simple text file.

Table of Contents

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 Jira (account required)

1. Log in to the Jira website.

2. Click **Create** in the top navigation bar

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 **Create** at the bottom of the dialogue.

CHAPTER 1. INTRODUCTION

Red Hat Enterprise Linux 9 delivers a stable, secure, consistent foundation across hybrid cloud deployments with the tools needed to deliver workloads faster with less effort. It can be deployed as a guest on supported hypervisors and Cloud provider environments as well as deployed on physical infrastructure, so your applications can take advantage of innovations in the leading hardware architecture platforms.

1.1. SUPPORTED ARCHITECTURES

Red Hat Enterprise Linux supports the following architectures:

- AMD and Intel 64-bit architectures

- The 64-bit ARM architecture

- IBM Power Systems, Little Endian

- 64-bit IBM Z architectures

> **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. INSTALLATION TERMINOLOGY

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

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

CHAPTER 2. INSTALLATION METHODS

Depending on your requirements, you can install Red Hat Enterprise Linux using several methods. Review the following sections to determine the best installation method for your requirements.

2.1. AVAILABLE 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

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 installation completes. This installation method is supported by the GUI and 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.

- **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.

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. The 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.

- **Register and install RHEL from the Content Delivery Network:**Register and install Red Hat Enterprise Linux on all architectures from the Content Delivery Network (CDN). Registration is performed before the installation packages are downloaded and installed from CDN. This installation method is supported by the graphical user interface and Kickstart.

PART I. PERFORMING AN AUTOMATED INSTALLATION USING KICKSTART

CHAPTER 3. KICKSTART INSTALLATION BASICS

The following provides basic information about Kickstart and how to use it to automate installing Red Hat Enterprise Linux.

3.1. 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 9, see Upgrading from RHEL 8 to RHEL 9 and Considerations in adopting RHEL.

3.2. AUTOMATED INSTALLATION WORKFLOW

Kickstart installations can be performed using a local DVD, a local hard drive, or a NFS, FTP, HTTP, or HTTPS server. This section provides a high level overview of Kickstart usage.

1. Create a Kickstart file. You can write it by hand, copy a Kickstart file saved after a manual installation, or use an online generator tool to create the file, and edit it afterward. See Creating Kickstart files.

2. Make the Kickstart file available to the installation program on removable media, a hard drive or a network location using an HTTP(S), FTP, or NFS server. See Making Kickstart files available to the installation program.

3. Create the boot medium which will be used to begin the installation. See Creating a bootable installation medium and Preparing to install from the network using PXE .

4. Make the installation source available to the installation program. See Creating installation sources for Kickstart installations.

5. Start the installation using the boot medium and the Kickstart file. See Starting Kickstart installations.

If the Kickstart file contains all mandatory commands and sections, the installation finishes automatically. If one or more of these mandatory parts are missing, or if an error occurs, the installation requires manual intervention to finish.

CHAPTER 4. 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 8 Kickstart file for Red Hat Enterprise Linux 9 installation. For more information about 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.

4.1. 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 9** 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.

4.2. 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 9 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.
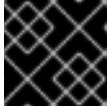
> **IMPORTANT**
>
> The file contains information about users and passwords.

- 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 9 installation

4.3. 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/

4.4. 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.

CHAPTER 5. 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.

5.1. 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 5.1. 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

5.2. 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 9 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:

   ```
   # dnf 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 **/rhel9-install/** directory available as read-only to all clients
   is:

   > /rhel9-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 **/rhel9-
> install/my-ks.cfg**, specify **inst.ks=nfs:myserver.example.com:/rhel9-install/my-ks.cfg**
> as the installation source boot option.

Additional resources

- [Preparing to install from the network using PXE](#)

5.3. 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 9 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:

   ```
   # dnf install httpd
   ```

   To store the Kickstart file on an HTTPS, install **httpd** and **mod_ssl** packages:

   ```
   # dnf 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/rhel9-install/my-ks.cfg**, specify **http://myserver.example.com/rhel9-install/my-ks.cfg** as the file location.

Additional resources

- Deploying Web Servers and Proxies

- Configuring and using Database Servers

5.4. 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 9 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:

   ```
   # dnf 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.

5.5. 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.

5.6. 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.

CHAPTER 6. 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.

6.1. 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 9, 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.

6.2. 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 6.1. 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

6.3. 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 9, 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:

   ```
   # dnf 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 **/rhel9-install/** directory available as read-only to all clients is:

  ```
  /rhel9-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 **/rhel9-install/**, specify **nfs:myserver.example.com:/rhel9-install/** as the installation source.

6.4. 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 9, 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 (LEGACY) is enabled.

> **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/rhel9-install/
   ```

3. Mount the DVD ISO image to the directory:

   ```
   # mount -o loop,ro -t iso9660 /image_directory/image.iso /mnt/rhel9-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/rhel9-install/ /var/www/html/
   ```

   This command creates the **/var/www/html/rhel9-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/rhel9-install/**, specify **http://myserver.example.com/rhel9-install/** as the installation source.

Additional resources

- Deploying different types of servers

6.5. 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 9, 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/rhel9-install
   ```

5. Mount the DVD ISO image to the directory:

   ```
   # mount -o loop,ro -t iso9660 /image-directory/image.iso /mnt/rhel9-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/rhel9-install
   # cp -r /mnt/rhel9-install/ /var/ftp/
   ```

   This command creates the **/var/ftp/rhel9-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/rhel9-install
# find /var/ftp/rhel9-install -type f -exec chmod 444 {} \;
# find /var/ftp/rhel9-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/rhel9-install/**, specify **ftp://myserver.example.com/rhel9-install/** as the installation source.

CHAPTER 7. INSTALLING RHEL ON ARM WITH KERNEL-64K

By default, RHEL 9 is distributed with a kernel supporting a 4k page size. This 4k kernel is sufficient for efficient memory usage in smaller environments or small cloud instances where the usage of a 64k page kernel is not practical due to space, power, and cost constraints.

> **IMPORTANT**
>
> It is not recommended to move between 4k and 64k page size kernels after the initial boot without reinstallation of the OS.

7.1. INSTALLING KERNEL-64K ON ARM USING KICKSTART

RHEL offers the ARM64 hardware architecture to support workloads that require large physical memory configuration for optimal performance. Such large memory configuration requires the use of a large MMU page size (64k).

While installing RHEL 9, you can select the **kernel-64k** package to install RHEL with kernel supporting 64k page size.

Procedure

- In the **%packages** section of the kickstart file, add the **kernel-64k** package:

  ```
  %packages
  -kernel
  kernel-64k
  %end
  ```

Verification steps

- To verify the page size, after installation is completed and the system is rebooted, open the terminal and run:

  ```
  $ getconf PAGESIZE
  65536
  ```

  The output **65536** indicates that the 64k kernel is in use.

- To verify that the swap partition is enabled, enter:

  ```
  $ free
             total       used       free     shared  buff/cache   available
  Mem:    35756352    3677184   34774848      25792      237120    32079168
  Swap:    6504384          0    6504384
  ```

The total and free columns are non-zero, which indicates the swap is enabled successfully.

7.2. INSTALLING KERNEL-64K ON ARM USING THE COMMAND LINE

If you have already installed RHEL with the default kernel (supporting 4k page size), you can install **kernel-64k** post installation using the command line.

Procedure

1. Open the terminal as the root user, and enter:

   ```
   # dnf -y install kernel-64k
   ```

2. To set the **kernel-64k** as default, enter:

   ```
   # k=$(echo /boot/vmlinuz*64k)
   # grubby --set-default=$k \
           --update-kernel=$k \
           --args="crashkernel=2G-:640M"
   ```

3. Reboot the system:

   ```
   # reboot
   ```

4. Optional: After reboot, remove the 4k kernel:

   ```
   # dnf erase kernel
   ```

   Keeping both versions accidentally can make the 4k kernel default when you update the kernel in future using the **yum update** command.

Verification steps

- To verify the page size, open the terminal and run the following command as any user:

  ```
  $ getconf PAGESIZE
  65536
  ```

  The output **65536** indicates that the 64k kernel is in use.

- To verify swap is enabled, enter:

  ```
  $ free
              total        used        free      shared  buff/cache   available
  Mem:     35756352     3677184    34774848       25792      237120    32079168
  Swap:     6504384           0     6504384
  ```

  The total and free columns are non-zero, which indicates the swap is enabled successfully.

CHAPTER 8. 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.

8.1. 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.

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.

8.2. 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-9/9.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-9/9.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.

8.3. 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.

CHAPTER 9. 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 9.1. Available tmux windows

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

CHAPTER 10. 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.

10.1. INSTALLING KICKSTART MAINTENANCE TOOLS

To use the Kickstart maintenance tools, you must install the package that contains them.

Procedure

- Install the **pykickstart** package:

  ```
  # dnf install pykickstart
  ```

10.2. 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 RHEL9** option in the **ksvalidator** command to acknowledge new commands of the RHEL9 class.

Procedure

- Run **ksvalidator** on your Kickstart file:

  ```
  $ ksvalidator -v RHEL9 /path/to/kickstart.ks
  ```

  Replace */path/to/kickstart.ks* with the path to the Kickstart file you want to verify.
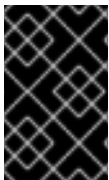
> **IMPORTANT**
>
> The validation tool cannot guarantee the installation will be successful. It ensures only that the syntax is correct and that the file does not include deprecated options. It does not attempt to validate the **%pre**, **%post** and **%packages** sections of the Kickstart file.

Additional resources

- The *ksvalidator(1)* man page

PART II. REGISTERING AND INSTALLING RHEL FROM THE CONTENT DELIVERY NETWORK AND SATELLITE

CHAPTER 11. 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.

11.1. 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.
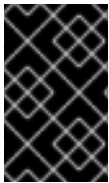
> **IMPORTANT**
>
> The CDN feature is supported by the **Boot ISO** and **DVD ISO** image files. However, it is recommended that you use the **Boot ISO** image file as the installation source defaults to CDN for the Boot ISO image file.

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 9 installation* document.
>
> - Repository configuration is not required in a Kickstart file as your subscription governs which CDN subset and repositories the system can access.

Procedure

1. Open the Kickstart file.

2. Edit the file to add the **rhsm** Kickstart command and its options to the file:

   Organization (required)

   Enter the organization id. An example is:

   ```
   --organization=1234567
   ```

NOTE

For security reasons, Red Hat username and password account details are not supported by Kickstart when registering and installing from the CDN.

Activation Key (required)

Enter the Activation Key. You can enter multiple keys as long as the activation keys are registered to your subscription. An example is:

```
--activation-key="Test_key_1" --activation-key="Test_key_2"
```

Red Hat Insights (optional)

Connect the target system to Red Hat Insights.

NOTE

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

11.2. 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
```

11.3. 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.

CHAPTER 12. REGISTERING AND INSTALLING RHEL FROM SATELLITE USING KICKSTART

This section contains information about how to register your system, attach RHEL subscriptions, and install from the Red Hat Satellite using Kickstart.

12.1. REGISTERING AND INSTALLING RHEL FROM SATELLITE

This procedure describes how to register your system, attach RHEL subscriptions, and install from a Satellite instance using the **rhsm** Kickstart command. It also shows how to configure system purpose and connect the system to Red Hat Insights. The **rhsm** Kickstart command removes the requirement of using custom **%post** scripts when registering the system.

> **IMPORTANT**
>
> - Satellite installation 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 Satellite for the Boot ISO image file.
>
> - 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 the system registration.
>
> - Repository configuration is not required in a Kickstart file as your subscription governs which satellite hosted repositories the system can access.

Prerequisites

- Your system is connected to a network that can access the Satellite instance.

- The Red Hat Satellite Server version is 6.11 or later

- 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 an organization ID, activation key, and the URL of the Satellite 6.11 instance you want to use.

- You have enabled, synced, and added the required BaseOS and AppStream RPMs repositories to the content-view.

- The activation key has the release version set to 9.x, and the concerned content-view is selected in it.

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 satellite.

Activation Key (required)

Enter the Activation Key. You can enter multiple keys as long as the activation keys are registered to your subscription. An example is:

> --activation-key="Test_key_1" --activation-key="Test_key_2"

Red Hat Insights (optional)

Connect the target system to Red Hat Insights.

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

Server hostname

> **NOTE**
>
> The Server hostname does not require the HTTP protocol, for example, **nameofhost.com**.

Set the server hostname if want to register to a Satellite instance. An example is:

```
--server-hostname="nameofhost.com"
```

System Purpose (optional)

Set the System Purpose role, SLA, and usage using the command:

```
syspurpose --role="Red Hat Enterprise Linux Server" --sla="Premium" --usage="Production"
```

Example

The following example displays a minimal Kickstart file with all **rhsm** Kickstart command options.

```
graphical
lang en_US.UTF-8
keyboard us
rootpw 12345
timezone America/New_York
zerombr
clearpart --all --initlabel
autopart
syspurpose --role="Red Hat Enterprise Linux Server" --sla="Premium" --usage="Production"
rhsm --organization="12345" --activation-key="test_key" --connect-to-insights --server-hostname="nameofhost.com" --proxy="user:password@hostname:9000"
reboot
%packages
vim
%end
```

3. Save the Kickstart file and start the installation process.

Verification steps

Once the system is installed, has been rebooted, and terminal window is opened, you can verify that your system is registered to satellite:

1. From the terminal window, enter the following command as a root user:

```
# subscription-manager list
Installed Product Status
Product Name: Red Hat Enterprise Linux for x86_64
Product ID: 486
Version: 9
Arch: x86_64
Status: Subscribed
Status Details
Starts: 11/4/2019
Ends: 11/4/2020
```

2. To view a detailed report:

```
# subscription-manager list --consumed
```

Additional resources

- Configuring System Purpose

- Starting Kickstart installations

- Red Hat Insights product documentation

- Understanding Activation Keys of Using Red Hat Subscription Management   For information about setting up an HTTP proxy for Subscription Manager, see the **PROXY CONFIGURATION** section in the **subscription-manager** man page.

12.2. UNREGISTERING YOUR SYSTEM FROM SATELLITE

This procedure describes how to unregister your system from the satellite.

Prerequisites

- You have completed the registration and installation process.

- You have started the Kickstart installation.

- The installed system has rebooted and a terminal window is open.

Procedure

- From the terminal window, enter the following command as a root user:

  ```
  # subscription-manager unregister
  ```

  The attached subscription is unregistered from the system and the connection to the satellite is removed.

PART III. ADVANCED CONFIGURATION OPTIONS

CHAPTER 13. CONFIGURING SYSTEM PURPOSE

You use System Purpose to record the intended use of a Red Hat Enterprise Linux 9 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.

13.1. 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 **satpurpose Kickstart** command

- After installation using the **subscription-manager** 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 9 installation](#)

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

13.2. 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 **subscription-manager** command-line tool. The **subscription-manager** 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=

13.3. ADDITIONAL RESOURCES

- Configuring System Purpose using the **subscription-manager** command-line tool

CHAPTER 14. 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.

14.1. 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.

14.2. 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.

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.

14.3. 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.

14.4. 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.

14.5. 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.
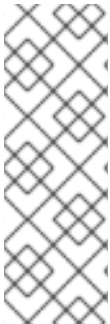
> **NOTE**
>
> If you burned an ISO image file to a CD or DVD but it does not have the **OEMDRV** volume label, you can use the **inst.dd** option with no arguments. The installation program provides an option to scan and select drivers from the CD or DVD. In this scenario, the installation program does 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.

14.6. 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

14.7. 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.

CHAPTER 15. 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.

15.1. 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.

15.2. 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.

15.3. 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**.

PART IV. KICKSTART REFERENCES

APPENDIX A. KICKSTART SCRIPT FILE FORMAT REFERENCE

This reference describes in detail the kickstart file format.

A.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.

A.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.

A.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 9 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 9. 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.
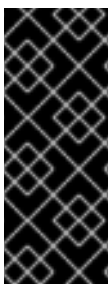
IMPORTANT

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**

A.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 9 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 9, modules are present only in the AppStream repository. To list available modules, use the **dnf module list** command on an installed Red Hat Enterprise Linux 9 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
```
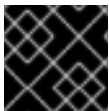
> **IMPORTANT**
>
> Installing all available packages using only ***** in a Kickstart file is not supported.

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

- [Managing Software with the DNF Tool](#)

A.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.

--inst-langs

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.
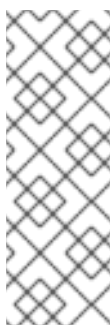
> **NOTES**
>
> - Using **-@Core** to exclude packages in the **@Core** package group does not work. The only way to exclude the **@Core** package group is with the **--nocore** option.
>
> - The **@Core** package group is defined as a minimal set of packages needed for installing a working system. It is not related in any way to core packages as defined in the Package Manifest and Scope of Coverage Details.

--exclude-weakdeps

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 DNF will attempt to download packages (retries). The default value is 10. This option only applies during the installation, and will not affect DNF configuration on the installed system.

--timeout=

Sets the DNF timeout in seconds. The default value is 30. This option only applies during the installation, and will not affect DNF configuration on the installed system.

A.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.

A.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

A.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.
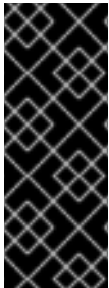
> ### IMPORTANT
>
> The **%pre** section of Kickstart is executed at the stage of installation which happens after the installer image (**inst.stage2**) is fetched: it means **after** root switches to the installer environment (the installer image) and **after** the **Anaconda** installer itself starts. 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.

A.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 *Installing and using dynamic programming languages*.

--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

A.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.

A.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 *Installing and using dynamic programming languages*.

--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
```

A.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.

A.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 *Installing and using dynamic programming languages*.

--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
```

A.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
```

A.4. 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 occured. 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 *Installing and using dynamic programming languages*.

--log=

Logs the script's output into the specified log file.

A.5. 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 B. KICKSTART COMMANDS AND OPTIONS REFERENCE

This reference is a complete list of all Kickstart commands supported by the Red Hat Enterprise Linux installation program 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.

B.1. KICKSTART CHANGES

The following sections describe the changes in Kickstart commands and options in Red Hat Enterprise Linux 9.

B.1.1. auth or authconfig is deprecated in RHEL 8

The **auth** or **authconfig** Kickstart command is deprecated in Red Hat Enterprise Linux 8 because the **authconfig** tool and package have been removed.

Similarly to **authconfig** commands issued on command line, **authconfig** commands in Kickstart scripts now use the **authselect-compat** tool to run the new **authselect** tool. For a description of this compatibility layer and its known issues, see the manual page **authselect-migration(7)**. The installation program will automatically detect use of the deprecated commands and install on the system the **authselect-compat** package to provide the compatibility layer.

B.1.2. 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.

B.1.3. Deprecated Kickstart commands and options

The following Kickstart commands and options have been deprecated in 9.

- **timezone --ntpservers** – use the **timesource** command instead

- **timezone --nontp**

- **logging --level**

- **%packages --excludeWeakdeps** – use **--exclude-weakdeps** instead

- **%packages --instLangs** – use **--inst-langs** instead

- **%anaconda**

- **pwpolicy** – use the Anaconda configuration files instead

- **syspurpose** – use **subscription-manager syspurpose** instead

Where only specific options are listed, the base command and its other options are still available and not deprecated. Using the deprecated 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.

B.1.4. Removed Kickstart commands and options

The following Kickstart commands and options have been completely removed in 9. 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**

- **autostep**

Where only specific options and values are listed, the base command and its other options are still available and not removed.

B.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.

B.2.1. cdrom

The **cdrom** Kickstart command is optional. It performs the installation from the first optical drive on the system.

Syntax

```
cdrom
```

Notes

- This command has no options.

- To actually run the installation, you must specify one of **cdrom**, **harddrive**, **hmc**, **nfs**, **liveimg**, **ostreesetup**, **rhsm**, or **url** unless the **inst.repo** option is specified on the kernel command line.

B.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.

B.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.

B.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.

B.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.

B.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.

B.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.

B.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, you must specify one of **cdrom**, **harddrive**, **hmc**, **nfs**, **liveimg**, **ostreesetup**, **rhsm**, or **url** unless the **inst.repo** option is specified on the kernel command line.

B.2.9. 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, you must specify one of **cdrom**, **harddrive**, **hmc**, **nfs**, **liveimg**, **ostreesetup**, **rhsm**, or **url** unless the **inst.repo** option is specified on the kernel command line.

B.2.10. 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.

B.2.11. 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.

B.2.12. 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

- To actually run the installation, you must specify one of **cdrom**, **harddrive**, **hmc**, **nfs**, **liveimg**, **ostreesetup**, **rhsm**, or **url** unless the **inst.repo** option is specified on the kernel command line.

B.2.13. 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/

B.2.14. 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 about you system's APM/ACPI abilities.

- This command has no options.

B.2.15. 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.
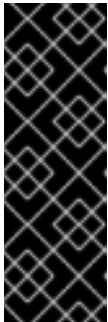
  > **IMPORTANT**
  >
  > This option is deprecated and available as a Technology Preview only. For information about Red Hat scope of support for Technology Preview features, see the Technology Preview Features Support Scope document.
  >
  > When **kexec** is used, device registers (which would normally 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.

B.2.16. 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.

- **--server-hostname=** – Sets the Satellite instance hostname for registration.

- To switch the installation source repository to the CDN by using the **rhsm** Kickstart command, you must meet the following conditions:

  - On the kernel command line, you have used **inst.stage2=<URL>** to fetch the installation image but have not specified an installation source using **inst.repo=**.

  - In the Kickstart file, you have not specified an installation source by using the **url**, **cdrom**, **harddrive**, **liveimg**, **nfs** and **ostree** setup commands.

- 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.

B.2.17. 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.

B.2.18. 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 is 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

B.2.19. 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.

B.2.20. url

The **url** Kickstart command is optional. It is used to install from an installation tree image on a remote server by using the FTP, HTTP, or HTTPS protocol. You can only specify one URL.

You must specify one of the **--url**, **--metalink** or **--mirrorlist** options.

Syntax

```
url --url=FROM [OPTIONS]
```

Options

- **--url=FROM** - Specifies the **HTTP**, **HTTPS**, **FTP**, or **file** location to install from.

- **--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
    ```

Notes

- To actually run the installation, you must specify one of **cdrom**, **harddrive**, **hmc**, **nfs**, **liveimg**, **ostreesetup**, **rhsm**, or **url** unless the **inst.repo** option is specified on the kernel command line.

B.2.21. 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](#)

B.2.22. %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
```

B.2.23. %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*

B.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.

B.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.

B.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.

B.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.

B.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.

B.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

B.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

B.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 9, modules are present only in the AppStream repository. To list available modules, use the **dnf module list** command on an installed Red Hat Enterprise Linux 9 system with a valid subscription.

Additional resources

- [Managing Software with the DNF Tool](#)

B.3.8. repo

The **repo** Kickstart command is optional. It configures additional dnf 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 dnf 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.

B.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.

- **--allow-ssh** – If this option is present, the root user can login to the system using SSH with a password. This option is available in RHEL 9.1 and later only.

Add the following line to the kickstart file during the kickstart installation method to enable **password-based SSH root logins**. The option **--allow-ssh** is not available in RHEL 9.0.

```
%post
echo "PermitRootLogin yes" > /etc/ssh/sshd_config.d/01-permitrootlogin.conf
%end
```

B.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](#)

B.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
  ```

B.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.

B.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.

B.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 9.0 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 9.0 and is planned to be removed in post 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.

B.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. This option is deprecated.

- **--ntpservers=** - Specify a list of NTP servers to be used as a comma-separated list without spaces. This option is deprecated, use the **timesource** command instead.

Notes

In Red Hat Enterprise Linux 9, 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.

B.3.16. timesource (optional)

The **timesource** kickstart command is optional. Use it to set NTP, NTS servers, and pools that provide time data, as well as control whether NTP services are enabled or disabled on the system.

Syntax

> timesource [--ntp-server *NTP_SERVER* | --ntp-pool *NTP_POOL* | --ntp-disable] [--nts]

Mandatory options

It is mandatory to specify one of the following options when you use the **timesource** command:

- **--ntp-server** - adds one NTP server as a time source. This option can be added only once to a single command in order to add a one NTP time source server. To add multiple sources, add multiple timesource commands each with a single --ntp-server or --ntp-pool option each time. For example, to add multiple sources for *Europe* timezone

  > **timezone *Europe***
  > **timesource --ntp-server *0.rhel.pool.ntp.org***
  > **timesource --ntp-server *1.rhel.pool.ntp.org***
  > **timesource --ntp-server *2.rhel.pool.ntp.org***

- **--ntp-pool** - adds a NTP server pool as a time source. This option can be added only once to add a single NTP time source pool. Repeat the timesource command to add multiple sources.

- **--ntp-disable** - disables NTP time sources for the installed system.

Optional options

- **--nts** - the server or pool added with this command uses the NTS protocol. Note that this option can be added even with --ntp-disable, but it has no effect.

Notes

- The **--ntpservers** option from the **timezone** command is deprecated. Red Hat recommends using this new option for expressive capabilities of the **timesource** command.

- Only **timesource** command can mark servers and pools as using **NTS** instead of plain **NTP** protocol.

B.3.17. 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.

- 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.

B.3.18. 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 9 does not include the KDE Desktop Environment, do not use the **--defaultdesktop=** documented in upstream.

B.4. KICKSTART COMMANDS FOR NETWORK CONFIGURATION

The Kickstart commands in this list let you configure networking on the system.

B.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 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.
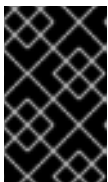
  > **IMPORTANT**
  >
  > Network teaming is deprecated in Red Hat Enterprise Linux 9. Consider using the network bonding driver as an alternative. For details, see Configuring network bonding.

  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**.

  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 the device specified in **--device=** as a parent. For example, **network --device=em1 --vlanid=171** creates a virtual LAN device **em1.171**.

- **--interfacename=** – Specify a custom interface name for a virtual LAN device. This option should be used when the default name generated by the **--vlanid=** option is not desirable. This option must be used along with **--vlanid=**. For example:

> network --device=em1 --vlanid=171 --interfacename=vlan171

The above command creates a virtual LAN interface named **vlan171** on the **em1** device with an ID of **171**.

The interface name can be arbitrary (for example, **my-vlan**), but in specific cases, the following conventions must be followed:

- If the name contains a dot (**.**), it must take the form of *NAME.ID*. The *NAME* is arbitrary, but the *ID* must be the VLAN ID. For example: **em1.171** or **my-vlan.171**.

- Names starting with **vlan** must take the form of **vlan***ID* – for example, **vlan171**.

- **--teamslaves=** – Team device specified by the **--device=** option will be created using 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.

> **IMPORTANT**
>
> Network teaming is deprecated in Red Hat Enterprise Linux 9. Consider using the network bonding driver as an alternative. For details, see Configuring network bonding.

- **--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\"}}"

  > **IMPORTANT**
  >
  > Network teaming is deprecated in Red Hat Enterprise Linux 9. Consider using the network bonding driver as an alternative. For details, see Configuring network bonding.

- **--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 9 installation* document.

B.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.

B.5. KICKSTART COMMANDS FOR HANDLING STORAGE

The Kickstart commands in this section configure aspects of storage such as devices, disks, partitions, LVM, and filesystems.

> IMPORTANT
>
> The **sdX** (or **/dev/sdX**) format does not guarantee consistent device names across reboots, which can complicate the usage of some Kickstart commands. When a command requires a device node name, you can use any item from **/dev/disk** as an alternative. For example, instead of using the following device name:
>
> **part / --fstype=xfs --onpart=sda1**
>
> You can use an entry similar to one of the following:
>
> **part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1**
>
> **part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1**
>
> By using this approach, the command always targets the same storage device. This is especially useful in large storage environments. To explore the available device names on the system, you can use the **ls -lR /dev/disk** command during the interactive installation. For more information about different ways to consistently refer to storage devices, see [Overview of persistent naming attributes](#).

B.5.1. 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.

- **--backuppassphrase** - 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 **--backuppassphrase** options.

- Ensure that the disk sector sizes are consistent when using **autopart**, **autopart --type=lvm**, or **autopart=thinp**.

B.5.2. 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.
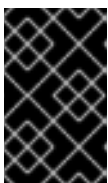
  > **IMPORTANT**
  >
  > The **--boot-drive=** option is currently being ignored in Red Hat Enterprise Linux installations on 64-bit IBM Z systems using the **zipl** boot loader. When **zipl** is installed, it determines the boot drive on its own.

- **--leavebootorder** – The installation program adds Red Hat Enterprise Linux 9 to the list of the installed systems in UEFI. It does not add the installed system to the boot order. All existing boot entries as well as their order are preserved.
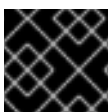
  > **IMPORTANT**
  >
  > This option is applicable for Power and UEFI systems.

- **--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 9 installation* document.

- The **sdX** (or **/dev/sdX**) format does not guarantee consistent device names across reboots, which can complicate the usage of some Kickstart commands. When a command requires a device node name, you can use any item from **/dev/disk** as an alternative. For example, instead of using the following device name:

> part / --fstype=xfs --onpart=sda1

You can use an entry similar to one of the following:

> part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
>
> part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1

By using this approach, the command always targets the same storage device. This is especially useful in large storage environments. To explore the available device names on the system, you can use the **ls -lR /dev/disk** command during the interactive installation. For more information about different ways to consistently refer to storage devices, see Overview of persistent naming attributes.

B.5.3. 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.

B.5.4. 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

- The **sdX** (or **/dev/sdX**) format does not guarantee consistent device names across reboots, which can complicate the usage of some Kickstart commands. When a command requires a device node name, you can use any item from **/dev/disk** as an alternative. For example, instead of using the following device name:

  > part / --fstype=xfs --onpart=sda1

  You can use an entry similar to one of the following:

  > part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
  >
  > part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1

  By using this approach, the command always targets the same storage device. This is especially useful in large storage environments. To explore the available device names on the system, you can use the **ls -lR /dev/disk** command during the interactive installation. For more information about different ways to consistently refer to storage devices, see Overview of persistent naming attributes.

- If the **clearpart** command is used, then the **part --onpart** command cannot be used on a logical partition.

B.5.5. 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.

B.5.6. 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

- 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.

- The **sdX** (or /**dev**/**sdX**) format does not guarantee consistent device names across reboots, which can complicate the usage of some Kickstart commands. When a command requires a device node name, you can use any item from /**dev**/**disk** as an alternative. For example, instead of using the following device name:

> part / --fstype=xfs --onpart=sda1

You can use an entry similar to one of the following:

> part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
>
> part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1

By using this approach, the command always targets the same storage device. This is especially useful in large storage environments. To explore the available device names on the system, you can use the **ls -lR /dev/disk** command during the interactive installation. For more information about different ways to consistently refer to storage devices, see Overview of persistent naming attributes.

B.5.7. 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**.

B.5.8. 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.

B.5.9. logvol

The **logvol** Kickstart command is optional. It creates a logical volume for Logical Volume Management (LVM).

Syntax

```
logvol mntpoint --vgname=name --name=name [OPTIONS]
```

Mandatory options

mntpoint

The mount point where the partition is mounted. Must be of one of the following forms:

- /*path*
  For example, / or /**home**

- **swap**
  The partition is used as swap space.

  To determine the size of the swap partition automatically, use the **--recommended** option:

  ```
  swap --recommended
  ```

  To determine the size of the swap partition automatically and also allow extra space for your system to hibernate, use the **--hibernation** option:

> swap --hibernation

The size assigned will be equivalent to the swap space assigned by **--recommended** plus the amount of RAM on your system.

--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.
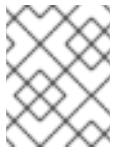
--useexisting

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.

--resize

Resize a logical volume. If you use this option, you must also specify **--useexisting** and **--size**.

--encrypted

Specifies that this logical volume should be encrypted with Linux Unified Key Setup (LUKS), using the passphrase provided in the **--passphrase=** option. If you do not specify a passphrase, the installation program uses the default, system-wide passphrase set with the **autopart --passphrase** command, or stops the installation and prompts you to provide a passphrase if no default is set.

> **NOTE**
>
> When encrypting one or more partitions, Anaconda attempts to gather 256 bits of entropy to ensure the partitions are encrypted securely. Gathering entropy can take some time - the process will stop after a maximum of 10 minutes, regardless of whether sufficient entropy has been gathered.
>
> The process can be sped up by interacting with the installation system (typing on the keyboard or moving the mouse). If you are installing in a virtual machine, you can also attach a **virtio-rng** device (a virtual random number generator) to the guest.

--passphrase=

Specifies the passphrase to use when encrypting this logical volume. You must use this option together with the **--encrypted** option; it has no effect by itself.

--cipher=

Specifies the type of encryption to use if the Anaconda default **aes-xts-plain64** is not satisfactory. You must use this option together with the **--encrypted** option; by itself it has no effect. Available types of encryption are listed in the Security hardening document, but Red Hat strongly recommends using either **aes-xts-plain64** or **aes-cbc-essiv:sha256**.

--escrowcert=_URL_of_X.509_certificate_

Store data encryption keys of all encrypted volumes as files in **/root**, encrypted using the X.509 certificate from the URL specified with *URL_of_X.509_certificate*. The keys are stored as a separate file for each encrypted volume. This option is only meaningful if **--encrypted** is specified.

--luks-version=*LUKS_VERSION*

Specifies which version of LUKS format should be used to encrypt the filesystem. This option is only meaningful if **--encrypted** is specified.

--backuppassphrase

Add a randomly-generated passphrase to each encrypted volume. Store these passphrases in separate files in **/root**, encrypted using the X.509 certificate specified with **--escrowcert**. This option is only meaningful if **--escrowcert** is specified.

--pbkdf=*PBKDF*

Sets Password-Based Key Derivation Function (PBKDF) algorithm for LUKS keyslot. See also the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified.

--pbkdf-memory=*PBKDF_MEMORY*

Sets the memory cost for PBKDF. See also the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified.

--pbkdf-time=*PBKDF_TIME*

Sets the number of milliseconds to spend with PBKDF passphrase processing. See also **--iter-time** in the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified, and is mutually exclusive with **--pbkdf-iterations**.

--pbkdf-iterations=*PBKDF_ITERATIONS*

Sets the number of iterations directly and avoids PBKDF benchmark. See also **--pbkdf-force-iterations** in the man page *cryptsetup(8)*. This option is only meaningful if **--encrypted** is specified, and is mutually exclusive with **--pbkdf-time**.

--thinpool

Creates a thin pool logical volume. (Use a mount point of **none**)

--metadatasize=*size*

Specify the metadata area size (in MiB) for a new thin pool device.

--chunksize=*size*

Specify the chunk size (in KiB) for a new thin pool device.

--thin

Create a thin logical volume. (Requires use of **--poolname**)

--poolname=*name*

Specify the name of the thin pool in which to create a thin logical volume. Requires the **--thin** option.

--profile=*name*

Specify the configuration profile name to use with thin logical volumes. If used, the name will also be included in the metadata for the given logical volume. By default, the available profiles are **default** and **thin-performance** and are defined in the **/etc/lvm/profile/** directory. See the **lvm(8)** man page for additional information.

--cachepvs=

A comma-separated list of physical volumes which should be used as a cache for this volume.

--cachemode=

Specify which mode should be used to cache this logical volume - either **writeback** or **writethrough**.
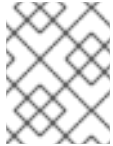
> **NOTE**
>
> For more information about cached logical volumes and their modes, see the **lvmcache(7)** man page.

--cachesize=

Size of cache attached to the logical volume, specified in MiB. This option requires the **--cachepvs=** option.

Notes

- Do not use the dash (**-**) character in logical volume and volume group names when installing Red Hat Enterprise Linux using Kickstart. If this character is used, the installation finishes normally, but the **/dev/mapper/** directory will list these volumes and volume groups with every dash doubled. For example, a volume group named **volgrp-01** containing a logical volume named **logvol-01** will be listed as **/dev/mapper/volgrp—01-logvol—01**.
  This limitation only applies to newly created logical volume and volume group names. If you are reusing existing ones using the **--noformat** option, their names will not be changed.

- If you lose the LUKS passphrase, any encrypted partitions and their data is completely inaccessible. There is no way to recover a lost passphrase. However, you can save encryption passphrases with the **--escrowcert** and create backup encryption passphrases with the **--backuppassphrase** 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

B.5.10. 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.

B.5.11. 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.

B.5.12. part or partition

The **part** or **partition** Kickstart command is required. It creates a partition on the system.

Syntax

> part|partition *mntpoint* [*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.

- **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).
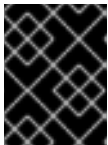
> **IMPORTANT**
>
> If the **--size** value is too small, the installation fails. Set the **--size** value as the minimum amount of space you require.

- **--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.

> **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.

- 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.

- The **sdX** (or /**dev**/**sdX**) format does not guarantee consistent device names across reboots, which can complicate the usage of some Kickstart commands. When a command requires a device node name, you can use any item from /**dev**/**disk** as an alternative. For example, instead of using the following device name:

```
part / --fstype=xfs --onpart=sda1
```

You can use an entry similar to one of the following:

```
part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
```

```
part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1
```

By using this approach, the command always targets the same storage device. This is especially useful in large storage environments. To explore the available device names on the system, you

can use the **ls -lR /dev/disk** command during the interactive installation. For more information about different ways to consistently refer to storage devices, see Overview of persistent naming attributes.

- If you lose the LUKS passphrase, any encrypted partitions and their data is completely inaccessible. There is no way to recover a lost passphrase. However, you can save encryption passphrases with the **--escrowcert** and create backup encryption passphrases with the **--backuppassphrase** options.

B.5.13. 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.
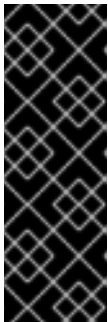
  > **IMPORTANT**
  >
  > - On IBM Power Systems, if a RAID device has been prepared and has not been reformatted during the installation, ensure that the RAID metadata version is **0.90** or **1.0** if you intend to put the **/boot** and PReP partitions on the RAID device. The **mdadm** metadata versions **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).

- **--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/md***X* 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.

- **--backuppassphrase** - Add a randomly-generated passphrase to this device. Store the passphrase in a file in /**root**, encrypted using the X.509 certificate specified with **--escrowcert**. This option is only meaningful if **--escrowcert** is specified.

- **--pbkdf=**_PBKDF_ - Sets Password-Based Key Derivation Function (PBKDF) algorithm for LUKS keyslot. See also the man page _cryptsetup(8)_. This option is only meaningful if **--encrypted** is specified.

- **--pbkdf-memory=**_PBKDF_MEMORY_ - Sets the memory cost for PBKDF. See also the man page _cryptsetup(8)_. This option is only meaningful if **--encrypted** is specified.

- **--pbkdf-time=**_PBKDF_TIME_ - Sets the number of milliseconds to spend with PBKDF passphrase processing. See also **--iter-time** in the man page _cryptsetup(8)_. This option is only meaningful if **--encrypted** is specified, and is mutually exclusive with **--pbkdf-iterations**.

- **--pbkdf-iterations=**_PBKDF_ITERATIONS_ - Sets the number of iterations directly and avoids PBKDF benchmark. See also **--pbkdf-force-iterations** in the man page _cryptsetup(8)_. This option is only meaningful if **--encrypted** is specified, and is mutually exclusive with **--pbkdf-time**.

Example

The following example shows how to create a RAID level 1 partition for /, and a RAID level 5 for **/home**, assuming there are three SCSI disks on the system. It also creates three swap partitions, one on each drive.

```
part raid.01 --size=6000 --ondisk=sda
part raid.02 --size=6000 --ondisk=sdb
part raid.03 --size=6000 --ondisk=sdc
part swap --size=512 --ondisk=sda
part swap --size=512 --ondisk=sdb
part swap --size=512 --ondisk=sdc
part raid.11 --size=1 --grow --ondisk=sda
part raid.12 --size=1 --grow --ondisk=sdb
part raid.13 --size=1 --grow --ondisk=sdc
raid / --level=1 --device=rhel8-root --label=rhel8-root raid.01 raid.02 raid.03
raid /home --level=5 --device=rhel8-home --label=rhel8-home raid.11 raid.12 raid.13
```

Notes

- If you lose the LUKS passphrase, any encrypted partitions and their data is completely inaccessible. There is no way to recover a lost passphrase. However, you can save encryption passphrases with the **--escrowcert** and create backup encryption passphrases with the **--backuppassphrase** options.

B.5.14. 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 toegether 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.

B.5.15. 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 **snaphost** 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.

B.5.16. 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.

B.5.17. 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.

B.5.18. zfcp

The **zfcp** Kickstart command is optional. It defines a Fibre channel device.

This option only applies on 64-bit IBM Z.

Syntax

```
zfcp --devnum=devnum [--wwpn=wwpn --fcplun=lun]
```

Options

- **--devnum=** – The device number (zFCP adapter device bus ID).

- **--wwpn=** – The device's World Wide Port Name (WWPN). Takes the form of a 16-digit number, preceded by **0x**.

- **--fcplun=** – The device's Logical Unit Number (LUN). Takes the form of a 16-digit number, preceded by **0x**.

> **NOTE**
>
> It is sufficient to specify an FCP device bus ID if automatic LUN scanning is available and when installing 9 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 **zfcp.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

```
zfcp --devnum=0.0.4000 --wwpn=0x5005076300C213e9 --fcplun=0x5022000000000000
zfcp --devnum=0.0.4000
```

B.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.

B.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).

B.6.2. %addon com_redhat_oscap

The **%addon com_redhat_oscap** 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.
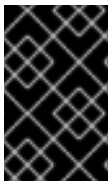
> **IMPORTANT**
>
> Applying a security policy is not necessary on all systems. This command should only be used when a specific policy is mandated by your organization rules or government regulations.

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 com_redhat_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 com_redhat_oscap** section which uses content from the **scap-security-guide** on the installation media:

  Example B.1. Sample OpenSCAP Add-on Definition Using SCAP Security Guide

  ```
  %addon com_redhat_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 B.2. Sample OpenSCAP Add-on Definition Using a Datastream

  ```
  %addon com_redhat_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](#)

B.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.

B.7.1. pwpolicy (deprecated)

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|--notstrict] [--emptyok|--notempty] [--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.

B.8. KICKSTART COMMANDS FOR SYSTEM RECOVERY

The Kickstart command in this section repairs an installed system.

B.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. nvdimm

   e. zfcp

4. Mount the system

   > rescue [--nomount|--romount]

5. Run %post script
   This step is run only if the installed system is mounted in read-write mode.

6. Start shell

7. Reboot system