Red Hat Enterprise Linux 8

Composing, installing, and managing RHEL for Edge images

Composing, installing, and managing RHEL for Edge Images on Red Hat Enterprise Linux 8

Last Updated: 2022-12-12

Red Hat Enterprise Linux 8 Composing, installing, and managing RHEL for Edge images

Composing, installing, and managing RHEL for Edge Images on Red Hat Enterprise Linux 8

Legal Notice

Abstract

This document is for users who want to compose customized RHEL (rpm-ostree) images using Image Builder, and then remotely install and manage the images on Edge servers.

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 Bugzilla (account required)

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

CHAPTER 1. INTRODUCING RHEL FOR EDGE IMAGES

A RHEL for Edge image is an **rpm-ostree** image that includes system packages to remotely install RHEL on Edge servers.

The system packages include:

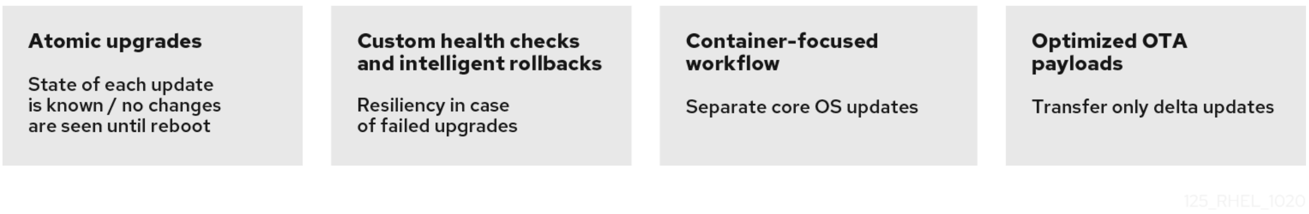
- **Base OS** package
- Podman as the container engine
- Additional RPM content

Differently from RHEL images, RHEL for Edge is an immutable operating system, that is, it contains an **read-only** root directory with the following characteristics:

- The package are isolated from root directory
- Package installs create layers that make it easy to rollback to previous versions
- Efficient updates to disconnected environments
- Supports multiple operating system branches and repositories
- Has a hybrid **rpm-ostree`package system**

You can deploy a RHEL for Edge image on Bare Metal, Appliance, and Edge servers.

With a RHEL for Edge image, you can achieve the following:


Atomic upgrades
State of each update is known / no changes are seen until reboot
Custom health checks and intelligent rollbacks
Resiliency in case of failed upgrades
Container-focused workflow
Separate core OS updates
Optimized OTA payloads
Transfer only delta updates
125_RHEL_1020

1.1. RHEL FOR EDGE—SUPPORTED ARCHITECTURE

Currently, you can deploy RHEL for Edge images on AMD and Intel 64-bit systems.

NOTE

Currently, RHEL for Edge does not support ARM systems.

1.2. HOW TO COMPOSE AND DEPLOY A RHEL FOR EDGE IMAGE

Composing and deploying a RHEL for Edge image involves two phases:

1. Composing a RHEL **rpm-ostree** image using Image Builder. You can access Image Builder through a command-line interface in the **composer-cli** tool, or use a graphical user interface in the RHEL web console.
2. Deploying the image using RHEL installer.

While composing a RHEL for Edge image, you can select any of the following image types. Composing the different RHEL for Edge images might or might not require network access. See the table:

Table 1.1. RHEL for Edge images type

Image type	Description	Suitable for network-based deployments	Suitable for non-network-based deployments
RHEL for Edge Commit (.tar)	Commit image is not directly bootable, even though it contains a full operating system. To boot the Commit image type, you must deploy it.	Yes	No
RHEL for Edge Container (.tar)	The Container creates an **OSTree** commit and embeds it into an OCI container with a web server. When the Container is started, the web server serves the commit as an OSTree repository.	No	Yes
RHEL for Edge Installer (.iso)	The RHEL for Edge Installer image type pulls the commit from the running container and creates an installable boot ISO with a Kickstart file configured to use the embedded OSTree commit.	No	Yes
RHEL for Edge Raw image (.raw.xz)	The compressed raw images consist of a file that contains a partition layout with an existing deployed OSTree commit in it. You can flash the RHEL Raw Images on a hard disk or boot on a virtual machine.	Yes	Yes

Image type	Description	Suitable for network-based deployments	Suitable for non-network-based deployments
RHEL for Edge Simplified Installer (.iso)	The Simplified Installer image type pulls the commit from a running container and creates an installable boot ISO with a Kickstart file configured to use the embedded OSTree commit.	Yes	Yes

The image types vary in terms of their contents, and are therefore suitable for different types of deployment environments.

Additional resources

- Performing a standard RHEL 8 installation .

1.3. NON-NETWORK-BASED DEPLOYMENTS

Use Image Builder to create flexible RHEL **rpm-ostree** images to suit your requirements, and then use Anaconda to deploy them in your environment.

You can access Image Builder through a command-line interface in the **composer-cli** tool, or use a graphical user interface in the RHEL web console.

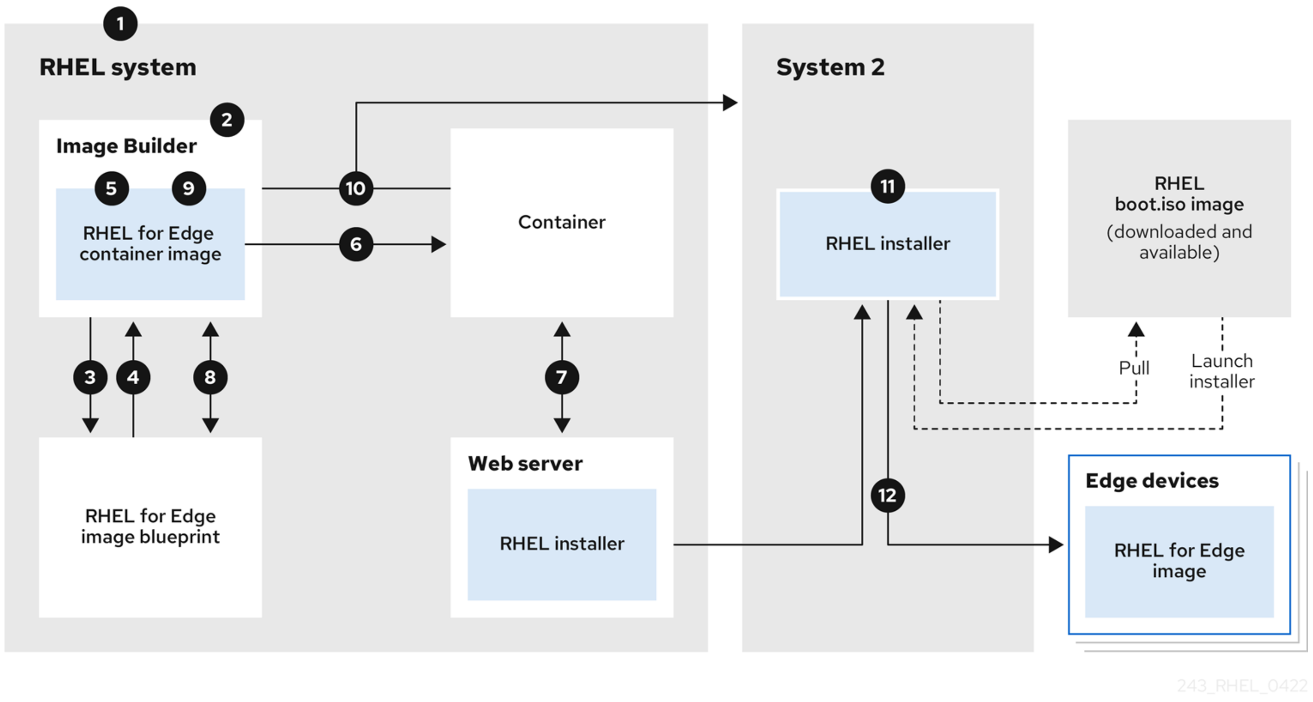
Composing and deploying a RHEL for Edge image in non-network-based deployments involves the following high-level steps:

1. Install and register a RHEL system
2. Install Image Builder
3. Using Image Builder, create a blueprint with customizations for RHEL for Edge Container image
4. Import the RHEL for Edge blueprint in Image Builder
5. Create a RHEL for Edge image embed in an OCI container with a webserver ready to deploy the commit as an OSTree repository
6. Download the RHEL for Edge Container image file
7. Deploy the container serving a repository with the RHEL for Edge Container commit
8. Using Image Builder, create another blueprint for RHEL for Edge Installer image
9. Create a RHEL for Edge Installer image configured to pull the commit from the running container embedded with RHEL for Edge Container image
10. Download the RHEL for Edge Installer image

11. Run the installation

The following diagram represents the RHEL for Edge image non-network deployment workflow:

Figure 1.1. Deploying RHEL for Edge in non-network environment



1.4. NETWORK-BASED DEPLOYMENTS

Use Image Builder to create flexible RHEL **rpm-ostree** images to suit your requirements, and then use Anaconda to deploy them in your environment. Image Builder automatically identifies the details of your deployment setup and generates the image output as an **edge-commit** as a **.tar** file.

You can access Image Builder through a command-line interface in the **composer-cli** tool, or use a graphical user interface in the RHEL web console.

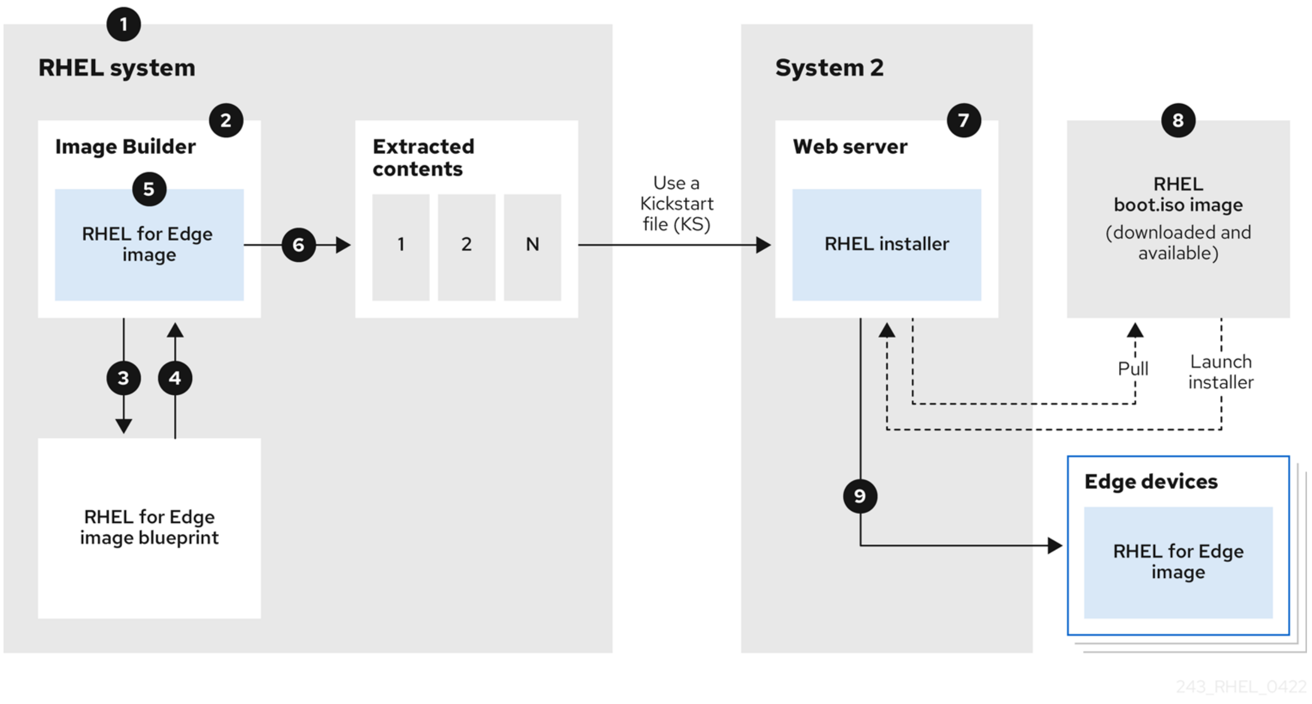
You can compose and deploy the RHEL for Edge image by performing the following high-level steps:

1. Install and register a RHEL system
2. Install Image Builder
3. Using Image Builder, create a blueprint for RHEL for Edge image
4. Import the RHEL for Edge blueprint in Image Builder
5. Create a RHEL for Edge Commit (**.tar**) image
6. Download the RHEL for Edge image file
7. Set up a web server
8. Create a new blueprint for a RHEL for Edge Installer (**.iso**)
9. Create the RHEL for Edge Installer (**.iso**) image, pointing at the OSTree content from the RHEL for Edge Commit (**.tar**) artifact

10. Download the RHEL for Edge installer ISO image you created
11. Boot the edge device using the RHEL for Edge Installer ISO image

The following diagram represents the RHEL for Edge network image deployment workflow:

Figure 1.2. Deploying RHEL for Edge in network-base environment



1.5. DIFFERENCE BETWEEN RHEL RPM IMAGES AND RHEL FOR EDGE IMAGES

You can create RHEL system images in traditional package-based RPM format and also as RHEL for Edge (**rpm-ostree**) images.

You can use the traditional package-based RPMs to deploy RHEL on traditional data centers. However, with RHEL for Edge images you can deploy RHEL on servers other than traditional data center. These servers include systems where processing of large amounts of data is done closest to the source where data is generated—Edge servers.

The RHEL for Edge (**rpm-ostree**) images are not a package manager. They only support complete bootable file system trees, not individual files. These images do not have information regarding the individual files such as how these files were generated or anything related to their origin.

The **rpm-ostree** images need a separate mechanism, the package manager, to install additional applications in the /**var** directory. With that, the **rpm-ostree** image keeps the operation system unchanged, while maintaining the state of the /**var** and /**etc** directories. The atomic updates enable rollbacks and background staging of updates.

Refer to the following table to know how RHEL for Edge images differ from the package-based RHEL RPM images.

Table 1.2. Difference between RHEL RPM images and RHEL for Edge images

Key attributes	RHEL RPM image	RHEL for Edge image
OS assembly	You can assemble the packages locally to form an image.	The packages are assembled in an ostree which you can install on a system.
OS updates	You can use **yum update** to apply the available updates from the enabled repositories.	You can use **rpm-ostree upgrade** to stage an update if any new commit is available in the ostree remote at **/etc/ostree/remotes.d/**. The update takes effect on system reboot.
Repository	The package contains YUM repositories	The package contains Ostree remote repository
User access permissions	Read write	Read-only (**/usr**)
Data persistence	You can mount the image to any non tmpfs mount point	**/etc** & **/var** are read-write enabled and include persisting data.

CHAPTER 2. SETTING UP IMAGE BUILDER

Use Image Builder to create your customized RHEL for Edge images. After you install Image Builder on a RHEL system, Image Builder is available as an application in RHEL web console. You can also access Image Builder through a command line interface in the **composer-cli** tool.

NOTE

It is recommended to install Image Builder on a virtual machine.

In the environment where you want to install Image Builder, ensure that you first meet the system requirements and then install it.

2.1. IMAGE BUILDER SYSTEM REQUIREMENTS

The environment where Image Builder runs, for example a virtual machine, must meet the requirements that are listed in the following table.

Table 2.1. Image Builder system requirements

Parameter	Minimal Required Value
System type	A dedicated virtual machine
Processor	2 cores
Memory	4 GiB
Disk space	20 GiB
Access privileges	Administrator level (root)
Network	Connectivity to Internet

NOTE

The 20 GiB disk space requirement is enough to install and run Image Builder in the host. To build and deploy image builds, you must allocate additional dedicated disk space.

2.2. INSTALLING IMAGE BUILDER

To install Image Builder on a dedicated virtual machine, follow these steps:

Prerequisites

- The virtual machine is created and is powered on.
- You have installed RHEL and you have subscribed to RHSM or Red Hat Satellite.

Procedure

1. Install the following packages on the virtual machine.

   - osbuild-composer
   - composer-cli
   - cockpit-composer
   - bash-completion
   - firewalld

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

   Image Builder is installed as an application in RHEL web console.

2. Reboot the virtual machine

3. Configure the system firewall to allow access to the web console:

   ```
   # firewall-cmd --add-service=cockpit && firewall-cmd --add-service=cockpit --permanent
   ```

4. Enable Image Builder.

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

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

5. Load the shell configuration script so that the autocomplete feature for the **composer-cli** command starts working immediately without reboot:

   ```
   $ source /etc/bash_completion.d/composer-cli
   ```

Additional resources

- Managing repositories.

CHAPTER 3. MANAGING IMAGE BUILDER REPOSITORIES

3.1. IMAGE BUILDER DEFAULT SYSTEM REPOSITORIES

The **osbuild-composer** back end does not inherit the system repositories located in the **/etc/yum.repos.d/** directory. Instead, it has its own set of official repositories defined in the **/usr/share/osbuild-composer/repositories** directory. This includes the Red Hat official repository, which contains the base system RPMs to install additional software or update already installed programs to newer versions. If you want to override the official repositories, you must define overrides in **/etc/osbuild-composer/repositories**. This directory is for user defined overrides and the files located there take precedence over those in the **/usr** directory.

The configuration files are not in the usual YUM repository format known from the files in **/etc/yum.repos.d/**. Instead, they are simple JSON files.

3.2. OVERRIDING A SYSTEM REPOSITORY

You can configure a repository override for image builder in the **/etc/osbuild-composer/repositories** directory with the following steps.

NOTE

Prior to RHEL 8.5 release, the name of the repository overrides is **rhel-8.json**. Starting from RHEL 8.5, the names also respect the minor version: **rhel-84.json**, **rhel-85.json**, and so on.

Prerequisites

- You have a custom repository that is accessible from the host system

Procedure

1. Create a directory where you want to store your repository overrides:

```
$ sudo mkdir -p /etc/osbuild-composer/repositories
```

2. You can create your own JSON file structure.

3. Create a JSON file, using a name corresponding to your RHEL version. Alternatively, you can copy the file for your distribution from **/usr/share/osbuild-composer/** and modify its content.

```
For RHEL 8, use /etc/osbuild-composer/repositories/rhel-85.json
```

4. Add the following structure to your JSON file, for example:

```
{
    "<ARCH>": [
        {
            "name": "baseos",
            "metalink": "",
            "baseurl": "http://mirror.example.com/composes/released/RHEL-
8/8.0/BaseOS/x86_64/os/",
            "mirrorlist": "",
            "gpgkey": "-----BEGIN PGP PUBLIC KEY BLOCK-----\n\n (…)",
            "check_gpg": true,
            "metadata_expire": ""
        }
    ]
}
```

Specify only one of the following attributes:

- **metalink** - offers increased availability and error correction on a repository link.
- **mirrorlist** - a list of URLs that point to package repositories.
- **baseurl** - a link to the repository that contains the packages required for the installation.
  The remaining fields are optional.

  a. Alternatively, you can copy the JSON file for your distribution.

     i. Copy the repository file to the directory you created. In the following command, replace **rhel-8.json** with your RHEL version, for example, **rhel-8.json**.
        In the following command, replace **rhel-version.json** with your RHEL version, for example: rhel-8.json.

```
$ cp /usr/share/osbuild-composer/repositories/rhel-version.json /etc/osbuild-composer/repositories/
```

5. Using a text editor, edit the **baseurl** paths in the **rhel-8.json** file and save it. For example:

```
$ vi /etc/osbuild-composer/repositories/rhel-version.json
```

6. Restart the **osbuild-composer.service**:

```
$ sudo systemctl restart osbuild-composer.service
```

Verification

- Check if the repository points to the correct URLs:

```
$ cat /etc/yum.repos.d/redhat.repo
```

  You can see that the repository points to the correct URLs which are copied from the **/etc/yum.repos.d/redhat.repo** file.

Additional resources

- latest RPMs version available in repository not visible for **osbuild-composer**.

3.3. OVERRIDING A SYSTEM REPOSITORY WITH SUPPORT FOR SUBSCRIPTIONS

The **osbuild-composer** service can use system subscriptions that are defined in the **/etc/yum.repos.d/redhat.repo** file. To use a system subscription in **osbuild-composer**, define a repository override that has:

- The same **baseurl** as the repository defined in **/etc/yum.repos.d/redhat.repo**.
- The value of **"rhsm": true** defined in the JSON object.

Prerequisites

- Your system has a subscription defined in **/etc/yum.repos.d/redhat.repo**
- You have created a repository override. See Overriding a system repository.

Procedure

1. Obtain the **baseurl** from the **/etc/yum.repos.d/redhat.repo** file:

```
# cat /etc/yum.repos.d/redhat.repo
[AppStream]
name = AppStream mirror example
baseurl = https://mirror.example.com/RHEL-8/8.0/AppStream/x86_64/os/
enabled = 1
gpgcheck = 0
sslverify = 1
sslcacert = /etc/pki/ca1/ca.crt
sslclientkey = /etc/pki/ca1/client.key
sslclientcert = /etc/pki/ca1/client.crt
metadata_expire = 86400
enabled_metadata = 0
```

2. Configure the repository override to use the same **baseurl** and set **rhsm** to true:

```
{
    "x86_64": [
        {
            "name": "AppStream mirror example",
            "baseurl": "https://mirror.example.com/RHEL-8/8.0/AppStream/x86_64/os/",
            "gpgkey": "-----BEGIN PGP PUBLIC KEY BLOCK-----\n\n (…)",
            "check_gpg": true,
            "rhsm": true
        }
    ]
}
```

NOTE

osbuild-composer does not automatically use repositories defined in **/etc/yum.repos.d/**. You need to manually specify them either as a system repository override or as an additional **source** using **composer-cli**. System repository overrides are usually used for “BaseOS” and “AppStream” repositories, whereas **composer-cli** sources are used for all the other repositories.

As a result, image builder reads the **/etc/yum.repos.d/redhat.repo** file from the host system and use it as a source of subscriptions.

Additional resources

- image builder uses CDN repositories when host is registered to Satellite 6

CHAPTER 4. COMPOSING A RHEL FOR EDGE IMAGE USING IMAGE BUILDER IN RHEL WEB CONSOLE

Use Image Builder to create a custom RHEL for Edge image (OSTree commit).

To access Image Builder and to create your custom RHEL for Edge image, you can either use the RHEL web console interface or the command-line interface. This section provides information about accessing Image Builder and creating RHEL for Edge images using the RHEL web console.

You can compose RHEL for Edge images using Image Builder in RHEL web console by performing the following high-level steps:

1. Access Image Builder in RHEL web console
2. Create a blueprint for RHEL for Edge image. You can create the following blueprints:
    - Blueprint with customizations for **"RHEL for Edge Commit (.tar)"**or **"RHEL for Edge Container (.tar)"** images
    - Blueprint for **"RHEL for Edge Installer (.iso)"**images
3. Create a user account for the RHEL for Edge image blueprint
4. Create a RHEL for Edge image. You can create the following images:
    - Creating a RHEL for Edge Commit image using Image Builder in RHEL web console
    - Creating a RHEL for Edge Container image using Image Builder in RHEL web console
    - Creating a RHEL for Edge Installer image using Image Builder in RHEL web console
5. Download the RHEL for Edge image

4.1. ACCESSING IMAGE BUILDER IN THE RHEL WEB CONSOLE

To access Image Builder in RHEL web console, ensure that you have met the following prerequisites and then follow the procedure.

Prerequisites

- You have installed a RHEL system.
- You have administrative rights on the system.
- You have subscribed the RHEL system to Red Hat Subscription Manager (RHSM) or to Red Hat Satellite server.
- The system is powered on and accessible over network.
- You have installed Image Builder on the system.

Procedure

1. On your RHEL system, access https://localhost:9090/ in a web browser.

2. For more information about how to remotely access Image Builder, see Managing systems using the RHEL 8 web console document.

3. Log in to the web console using an administrative user account.

4. On the web console, in the left hand menu, click **Apps**.

5. Click **Image Builder**.
   The Image Builder dashboard opens in the right pane. You can now proceed to create a blueprint for the RHEL for Edge images.

4.2. CREATING A BLUEPRINT FOR THE RHEL FOR EDGE COMMIT IMAGE USING IMAGE BUILDER IN RHEL WEB CONSOLE

To create a blueprint for the RHEL for Edge Commit image using Image Builder in RHEL web console, ensure that you have met the following prerequisites and then follow the procedure.

Prerequisites

- On a RHEL system, you have opened the Image Builder dashboard.

Procedure

1. On the Image Builder dashboard, click **Create Blueprint**.
   The **Create Blueprint** dialogue box opens.

2. Specify a name and a description for the blueprint that you want to create.

3. Click **Create**.
   The dashboard displays a list with the **Available components**.

   NOTE

   Your system must be subscribed to RHSM, otherwise, the list of available components appears as “Loading”.

4. To search for a specific component, enter the component name in the **Filter By Name** text box, and press **Enter**. The **Component Details** pane displays the component details and its dependent components.

5. In the **Components Details** pane, click **Add**.
   The web console selects the latest version by default. To select the required component version:

   a. Click the component you want to add.

   b. Click the button next to the component name and select **View**.

   c. From the **Version** dropdown menu under **Component Options**, select the required version.

   d. Click **Apply Change**.
      If you want to remove a component from the blueprint, in the **Available Components** pane, click **-** against the component name.

6. Click **Commit** to save the blueprint.

The dialog box with the blueprint summary opens.

7. Click **Commit**.
8. Click **Back to Blueprints**.
   The Image Builder dashboard lists the blueprints that you created.

4.3. CREATING A BLUEPRINT FOR THE RHEL FOR EDGE CONTAINER IMAGE USING IMAGE BUILDER IN RHEL WEB CONSOLE

You can create a blueprint for the RHEL for Edge Container image using Image Builder in RHEL web console, and add customizations, such as packages. The blueprint with customizations will be used to create a RHEL for Edge Container image.

Prerequisites

- On a RHEL system, you have opened the Image Builder dashboard.

Procedure

1. On the Image Builder dashboard, click **Create Blueprint**.
   A **Create Blueprint** dialogue box opens.
2. Specify a name and description for the blueprint that you want to create.
3. Click **Create**.
   The dashboard displays a list of available components.

   NOTE

   Your system must be subscribed to RHSM, otherwise, the list of available components appears as “Loading”.

4. Optional: Customize the blueprint with components.
   a. To search for a specific component, enter the component name in the **Filter By Name** text box, and press **Enter**. The **Component Details** page displays the component details and its dependent components.
   b. In the **Components Details** pane, click **Add**.
      The web console selects the latest version by default. To select the required component version:
   c. Click the component you want to add.
   d. Click the button next to the component name and select **View**.
   e. From the **Version** dropdown menu under **Component Options**, select the required version.
   f. Click **Apply Change**.
      If you want to remove a component from the blueprint, in the **Available Components** pane, click **-** against the component name.
5. Click **Commit** to save the pending changes in the blueprint.

A dialog box with the selected components changes appears.

6. Click **Commit**.
7. Click **Back to Blueprints**.
   The Image Builder dashboard lists the blueprints that you created.

4.4. CREATING A BLUEPRINT FOR THE RHEL FOR EDGE INSTALLER IMAGE USING IMAGE BUILDER IN RHEL WEB CONSOLE

You can create a blueprint to build a **RHEL for Edge Installer (.iso)**image, and specify user accounts to automatically create one or more users on the system at installation time. See Creating an administrative user account for a RHEL for Edge image blueprint.

Prerequisites

- On a RHEL system, you have opened the Image Builder dashboard.

Procedure

1. On the Image Builder dashboard, click **Create Blueprint**.
   The **Create Blueprint** dialogue box opens.
2. Specify a name and description for the blueprint that you want to create.
3. Click **Create**.
4. Click **Back to Blueprints**.
   The Image Builder dashboard lists the blueprints that you created.
5. Click the blueprint you created for the RHEL for Edge Installer image.
6. On the **Customization** tab, click **Create user account**.
   A wizard window opens.
7. Enter the user account details and click **Create**.
   Image Builder lists the user account you created.

4.5. ADDING A SOURCE TO RHEL FOR EDGE IMAGE BLUEPRINT

By default, the **appstream** and **baseos yum** official sources are available. You can use the sources defined in Image Builder, **yum repository**, **mirrorlist** and **metalink**, to include the RPMs packages from custom third parties repositories and you can add them to blueprints. These sources are global and therefore available to all blueprints.

The **System sources** repositories are set up locally on your computer and cannot be disabled or removed from Image Builder. You can add additional custom sources and thus be able to access other contents than the **System sources** available on your system. Any RPM repository that is accessible from the host system is valid as a source. It is not possible to disable a **System Source**.

Perform the following steps to add a Source to your blueprints.

Prerequisites

- You have opened the Image Builder interface of the RHEL web console in a browser.

Procedure

1. Click the **Manage Sources** button in the upper right corner.
   A pop-up window appears with the available sources, their names and descriptions.
2. On the right side of the pop-up window, click the **Add Source** button.
3. Add the desired **Source name**, the **Source path**, and the **Source Type**.
   Optionally, check the boxes related to the **Security field**.
   a. **SSL Certificate** - to verify a repository's identity and enable an encrypted connection.
   b. **GPG Key** - to validate signatures of RPM packages available in this repository.
4. Click **Add Source**. The screen shows the available sources window and lists the source you have added.
   As a result, the new System source is available for use or any changes that you may want to make.

4.6. CREATING AN ADMINISTRATIVE USER ACCOUNT FOR A RHEL FOR EDGE IMAGE BLUEPRINT

The RHEL for Edge images you create using Image Builder have the root account locked and no other accounts included. Image Builder enables you to create a user account with password for a RHEL for Edge blueprint so that you can log in to the RHEL for Edge image created from the blueprint. For the administrative user account you can either have a password-based access or an SSH-key based access.

NOTE

For Network-based installation that uses kickstart, you can also create a user account with Kickstart.

Prerequisites

- You have created an SSH key that you can use for the user account to be created.
- You have accessed the Image Builder dashboard in the RHEL web console.
- You have created a blueprint for RHEL for Edge image.

Procedure

1. On the Image Builder dashboard, find the blueprint for RHEL for Edge image.
   To search a required blueprint, specify the blueprint name in the Filter by Name text box and then press **Enter**.
2. Click the blueprint name.
   Image Builder displays the blueprint details.
3. On the Customizations tab, click **Create User Account**.
4. On the Create User Account dialogue box, specify the required details and a password for the user account.

For password-based access, specify a password for the user account.

For SSH-based access, specify an SSH key for the user account.

NOTE

Ensure that you select the **Server administrator** check box, if you want to provide administrators rights to the user account you are creating.

Image Builder creates a specified user account and displays the details.

You can create additional user accounts, if required.

4.7. CREATING A RHEL FOR EDGE COMMIT IMAGE USING IMAGE BUILDER IN RHEL WEB CONSOLE

You can create RHEL for Edge Commit images for network-based deployment by selecting **“RHEL for Edge Commit (.tar)”**. The **“RHEL for Edge Commit (.tar)”**commit image type is not directly bootable, even though it contains a full operating system. To boot commit image type, you must deploy it in a running container.

To create a RHEL for Edge Commit image using Image Builder in RHEL web console, follow the steps:

Prerequisites

- On a RHEL system, you have accessed the Image Builder dashboard.
- You have created a blueprint for RHEL for Edge Commit image.

Procedure

1. On the Image Builder dashboard, for the blueprint that you have created for **RHEL for Edge Commit** image, click **Create Image**.
   To search for a specific blueprint, enter the blueprint name in the **Filter By Name** text box, and then press **Enter**.

2. On the **Create Image** window, perform the following steps:

3. From the **Type** dropdown list, select **“RHEL for Edge Commit (.tar)”**for network-based deployment.

4. Specify **one** of the following arguments: **Repository URL** or **Parent commit**.

a. **Repository URL**: specify the URL to the OSTree repository of the commit to embed in the image. For example, http://10.0.2.2:8080/repository/.

b. **Parent commit**: specify a previous commit, or leave it empty if you do not have a commit at this time.

   1. In the **Ref** textbox, specify a reference path for where your commit is going to be created. By default, the web console specifies **rhel/8/$ARCH/edge**. The "$ARCH" value is determined by the host machine.

   2. Click **Create**.

Image Builder starts to create a RHEL for Edge Commit image for the blueprint that you created.

3. To check the RHEL for Edge Commit image creation progress:

c. Click the **blueprint name** from the breadcrumbs.

d. Click the **Images** tab.

> NOTE
>
> The image creation process takes up to 20 minutes to complete. To stop the image creation process, click **Stop** from the More Options menu.

After the image creation process is complete, you can download the resulting **"RHEL for Edge Commit (.tar)"** image.

Additional resources

- Downloading a RHEL for Edge image

4.8. CREATING A RHEL FOR EDGE CONTAINER IMAGE USING IMAGE BUILDER IN RHEL WEB CONSOLE

You can create RHEL for Edge images for non-network-based deployment by selecting **"RHEL for Edge Container (.tar)"**. The **RHEL for Edge Container (.tar)** image type creates an OSTree commit and embeds it into an OCI container with a web server. When the container is started, the web server serves the commit as an OSTree repository.

Follow the steps in this procedure to create a RHEL for Edge Container image using Image Builder in RHEL web console.

Prerequisites

- On a RHEL system, you have accessed the Image Builder dashboard.
- You have created a blueprint for RHEL for Edge Container image. See link:See Creating a RHEL for Edge Container image using Image Builder in RHEL web console.

Procedure

1. On the Image Builder dashboard, for the blueprint that you have created for RHEL for Edge image, **click Create Image**. To search for a specific blueprint, enter the blueprint name in the **Filter By Name** text box, and then press **Enter**.

2. On the **Create Image** window, perform the following steps:

3. Select **"RHEL for Edge Container (.tar)"**, from the **Type** dropdown list,.

4. In the **Repository URL** textbox, specify the URL to the OSTree repository of the commit to embed in the image. For example, http://10.0.2.2:8080/repository/.

5. In the **Parent commit** textbox, specify a previous commit, or leave it empty; if you do not have a commit at this time.

6. In the **Ref** textbox, specify a reference path for where your commit is going to be created. By default, the web console specifies **rhel/8/$ARCH/edge**. The "$ARCH" value is determined by the host machine.

7. Click **Create**.
   Image Builder starts to create a RHEL for Edge Container image for the blueprint that you created.

8. To check the RHEL for Edge Container image creation progress:

   a. Click the **blueprint name** from the breadcrumbs.

   b. Click the **Images** tab.

   NOTE

   The image creation process takes up to 20 minutes to complete. To abort the image creation process, click **Stop** from the More Options menu.

After the image creation process is complete, you can download the resulting **"RHEL for Edge Container (.tar)"** image.

Additional resources

- Downloading a RHEL for Edge image

4.9. CREATING A RHEL FOR EDGE INSTALLER IMAGE USING IMAGE BUILDER IN RHEL WEB CONSOLE

You can create RHEL for Edge Installer images for non-network-based deployment by selecting **"RHEL for Edge Installer (.iso)"**. The **RHEL for Edge Installer (.iso)**image type pulls the OSTree commit repository from the running container served by the **RHEL for Edge Container (.tar)** and creates an installable boot ISO image with a kickstart file that is configured to use the embedded OSTree commit.

Follow the steps in this procedure to create a RHEL for Edge image using Image Builder in RHEL web console.

Prerequisites

- On a RHEL system, you have accessed the Image Builder dashboard.

- You created a blueprint for RHEL for Edge Installer image. See Creating a RHEL for Edge Installer image using Image Builder in RHEL web console.

- You created a RHEL for Edge Container image and loaded it into a running container. See Creating a RHEL for Edge Container image for non-network-based deployments .

Procedure

1. On the Image Builder dashboard, for the blueprint that you have created for **RHEL for Edge Installer** image, click **Create Image**.
   To search for a specific blueprint, enter the blueprint name in the **Filter By Name** text box, and then press **Enter**.

2. On the **Create Image** window, perform the following steps:

    a. From the **Type** dropdown list, select **“RHEL for Edge Installer (.iso)”**.

    b. In the **Repository URL** textbox, specify the URL to the running container OSTree repository of the commit to embed in the image. For example, http://10.0.2.2:8080/repository/.

    c. The **Parent commit** textbox, you can specify a previous commit, or leave it empty; if you do not have a commit at this time.

    d. In the **Ref** textbox, the reference path must match the **Ref** from the RHEL for Edge Container image compose.

3. Click **Create**.
   Image Builder starts to create a **RHEL for Edge Installer** image for the blueprint that you created.

4. To check the RHEL for Edge Installer image creation progress:

    a. Click the **blueprint name** from the breadcrumbs.

    b. Click the **Images** tab.

    NOTE

    The image creation process takes up to 20 minutes to complete. To abort the image creation process, click **Stop** from the **More Options** menu.

After the image creation process is complete, you can download the resulting **RHEL for Edge Installer (.iso)** image and boot the ISO image into a device.

Additional resources

- Downloading a RHEL for Edge image .

4.10. DOWNLOADING A RHEL FOR EDGE IMAGE

After Image Builder successfully creates the RHEL for Edge image, download the image on the local host.

Procedure

To download an image:

1. From the **More Options** menu, click **Download**.
   The Image Builder downloads the file at your default download location.

The downloaded file consists of a **.tar** file with an OSTree repository for RHEL for Edge Commit and RHEL for Edge Container images, or a **.iso** file for RHEL for Edge Installer images, with an OSTree repository. This repository contains the commit and a **json** file which contains information metadata about the repository content.

4.11. ADDITIONAL RESOURCES

- Composing a RHEL for Edge image using Image Builder command-line .

CHAPTER 5. COMPOSING A RHEL FOR EDGE IMAGE USING IMAGE BUILDER COMMAND-LINE

You can use Image Builder to create a customized RHEL for Edge image (OSTree commit).

To access Image Builder and to create your custom RHEL for Edge image, you can either use the RHEL web console interface or the command-line interface. This chapter provides information about creating RHEL for Edge images using the CLI.

For Network-based deployments, the workflow to compose RHEL for Edge images using the CLI, involves the following high-level steps:

1. Create a blueprint for RHEL for Edge image
2. Create a RHEL for Edge Commit image
3. Download the RHEL for Edge Commit image

For Non-Network-based deployments, the workflow to compose RHEL for Edge images using the CLI, involves the following high-level steps:

1. Create a blueprint for RHEL for Edge image
2. Create a blueprint for the RHEL for Edge Installer image
3. Create a RHEL for Edge Container image
4. Create a RHEL for Edge Installer image
5. Download the RHEL for Edge image

To perform the steps, use the **composer-cli** package.

NOTE

To run the **composer-cli** commands as non-root, you must be part of the **weldr** group or you must have administrator access to the system.

5.1. NETWORK-BASED DEPLOYMENTS WORKFLOW

This provides steps on how to build **OSTree** commits. These **OSTree** commits contain a full operating system, but are not directly bootable. To boot them, you need to deploy them using a Kickstart file.

5.1.1. Creating a RHEL for Edge Commit image blueprint using Image Builder command-line interface

Create a blueprint for RHEL for Edge Commit image using the CLI.

Prerequisite

- You do not have an existing blueprint. To verify that, list the existing blueprints:

```
$ sudo composer-cli blueprints list
```

Procedure

1. Create a plain text file in the TOML format, with the following content:

   ```
   name = "blueprint-name"
   description = "blueprint-text-description"
   version = "0.0.1"
   modules = [ ]
   groups = [ ]
   ```

   Where,

   - *blueprint-name* is the name and blueprint-text-description is the description for your blueprint.
   - *0.0.1* is the version number according to the Semantic Versioning scheme.
   - *Modules* describe the package name and matching version glob to be installed into the image, for example, the package name = "tmux" and the matching version glob is version = "2.9a".
     Notice that currently there are no differences between packages and modules.
   - *Groups* are packages groups to be installed into the image, for example the group package anaconda-tools.
     At this time, if you do not know the modules and groups, leave them empty.

2. Include the required packages and customize the other details in the blueprint to suit your requirements.
   For every package that you want to include in the blueprint, add the following lines to the file:

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

   Where,

   - package-name is the name of the package, such as httpd, gdb-doc, or coreutils.
   - package-version is the version number of the package that you want to use.
     The package-version supports the following dnf version specifications:
   - For a specific version, use the exact version number such as 8.0.
   - For the latest available version, use the asterisk *.
   - For the latest minor version, use formats such as 8.*.

3. Push (import) the blueprint to the Image Builder server:

   ```
   # composer-cli blueprints push blueprint-name.toml
   ```

4. List the existing blueprints to check whether the created blueprint is successfully pushed and exists.

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

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

```
# composer-cli blueprints depsolve blueprint-name
```

Additional resources

- Supported Image Customizations.

5.1.2. Creating a RHEL for Edge Commit image using Image Builder command-line interface

To create a RHEL for Edge Commit image using Image Builder command-line interface, ensure that you have met the following prerequisites and follow the procedure.

Prerequisites

- You have created a blueprint for RHEL for Edge Commit image.

Procedure

1. Create the RHEL for Edge Commit image.

```
# composer-cli compose start blueprint-name image-type
```

   Where,

   - *blueprint-name* is the RHEL for Edge blueprint name.
   - *image-type* is **edge-commit** for **network-based deployment**.
     A confirmation that the composer process has been added to the queue appears. It also shows a Universally Unique Identifier (UUID) number for the image created. Use the UUID number to track your build. Also keep the UUID number handy for further tasks.

2. Check the image compose status.

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

   The output displays the status in the following format:

```
<UUID> RUNNING date blueprint-name blueprint-version image-type
```

   NOTE

   The image creation process takes up to 20 minutes to complete.

   To interrupt the image creation process, run:

```
# composer-cli compose cancel <UUID>
```

   To delete an existing image, run:

```
# composer-cli compose delete <UUID>
```

After the image is ready, you can download it and use the image on your **network deployments**.

Additional resources

- Composing a RHEL for Edge image using Image Builder command-line

5.1.3. Creating a RHEL for Edge image update with a parent commit using Image Builder command-line interface

If you performed a change in an existing blueprint, for example, you added a new package, and want to update an existing RHEL for Edge image with this new package, you can use a parent commit ID to generate an updated RHEL for Edge Commit (.tar) image.

To create a RHEL for Edge image with a parent commit using Image Builder command line interface, ensure that you have met the following prerequisites and then follow the procedure.

Prerequisites

- You have updated an existing blueprint for RHEL for Edge image.
- You have an existing RHEL for Edge image (OSTree commit). See Extracting RHEL for Edge image commit.

Procedure

1. Create the RHEL for Edge image.

   ```
   # composer-cli compose start-ostree --ref rhel/8/x86_64/edge --parent parent-OSTree-commit-id blueprint-name image-type
   ```

   Where,

   - *--ref* is the same value used by to build ostree repository
   - *--parent* is the OSTree parent commit
   - *blueprint-name* is the RHEL for Edge blueprint name.
   - *image-type* is **edge-commit** for **network-based deployment**

   NOTE

   The **--parent** argument can only be used for RHEL for Edge Commit (.tar) image type. Using the **--url** and **--parent** arguments together results in errors.

   A confirmation that the composer process has been added to the queue appears. It also shows a Universally Unique Identifier (UUID) number for the image created. Use the UUID number to track your build. Also keep the UUID number handy for further tasks.

2. Check the image compose status.

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

The output displays the status in the following format:

```
<UUID> RUNNING date blueprint-name blueprint-version image-type
```

NOTE

The image creation process takes a few minutes to complete.

(Optional) To interrupt the image creation process, run:

```
# composer-cli compose cancel <UUID>
```

(Optional) To delete an existing image, run:

```
# composer-cli compose delete <UUID>
```

After the image creation is complete, to upgrade an existing ostree deployment, you need:

- Set up a repository. See Deploying a RHEL for Edge image .
- Add this repository as a remote, that is, the http or https endpoint that hosts the ostree content.
- Pull the new OSTree commit onto their existing running instance. See Deploying RHEL for Edge image updates manually .

Additional resources

- Creating a system image with Image Builder in the command-line interface
- Downloading a RHEL for Edge image using the Image Builder command-line interface

5.1.4. Downloading a RHEL for Edge image using the Image Builder command-line interface

To download a RHEL for Edge image using Image Builder command line interface, ensure that you have met the following prerequisites and then follow the procedure.

Prerequisites

- You have created a RHEL for Edge image.

Procedure

1. Review the RHEL for Edge image status.

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

   The output must display the following:

   ```
   $ <UUID> FINISHED date blueprint-name blueprint-version image-type
   ```

2. Download the image.

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

   Image Builder downloads the image as a **tar** file to the current directory.

   The UUID number and the image size is displayed alongside.

   ```
   $ <UUID>-commit.tar: size MB
   ```

The image contains a commit and a **json** file with information metadata about the repository content.

Additional resources

- Deploying a RHEL for Edge image in a network-base environment .

5.2. NON-NETWORK-BASED DEPLOYMENTS WORKFLOW

This provides steps on how to build a boot ISO image that installs an OSTree-based system using the "RHEL for Edge Container" and the "RHEL for Edge Installer" images and that can be later deployed to a device in disconnected environments.

5.2.1. Creating a RHEL for Edge Container blueprint using Image Builder CLI

To create a blueprint for RHEL for Edge Container image, perform the following steps:

Procedure

1. Create a plain text file in the TOML format, with the following content:

   ```
   name = "blueprint-name"
   description = "blueprint-text-description"
   version = "0.0.1"
   modules = [ ]
   groups = [ ]
   ```

   Where,

   - *blueprint-name* is the name and blueprint-text-description is the description for your blueprint.
   - *0.0.1* is the version number according to the Semantic Versioning scheme.
   - *Modules* describe the package name and matching version glob to be installed into the image, for example, the package name = "tmux" and the matching version glob is version = "2.9a".
     Notice that currently there are no differences between packages and modules.
   - *Groups* are packages groups to be installed into the image, for example the group package anaconda-tools.
     At this time, if you do not know the modules and groups, leave them empty.

2. Include the required packages and customize the other details in the blueprint to suit your requirements.

For every package that you want to include in the blueprint, add the following lines to the file:

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

Where,

- package-name is the name of the package, such as httpd, gdb-doc, or coreutils.
- package-version is the version number of the package that you want to use.
  The package-version supports the following dnf version specifications:
- For a specific version, use the exact version number such as 8.0.
- For the latest available version, use the asterisk *.
- For the latest minor version, use formats such as 8.*.

3. Push (import) the blueprint to the Image Builder server:

   ```
   # composer-cli blueprints push blueprint-name.toml
   ```

4. List the existing blueprints to check whether the created blueprint is successfully pushed and exists.

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

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

   ```
   # composer-cli blueprints depsolve blueprint-name
   ```

Additional resources

- Supported Image Customizations.

5.2.2. Creating a RHEL for Edge Installer blueprint using Image Builder CLI

You can create a blueprint to build a **RHEL for Edge Installer (.iso)** image, and specify user accounts to automatically create one or more users on the system at installation time. See Creating an administrative user account for a RHEL for Edge image blueprint. It creates a user on the system at installation time.

WARNING

When you create a user in the blueprint with the **customizations.user** customization, the blueprint creates the user under the **/usr/lib/passwd** directory and the password, under the **/usr/etc/shadow** directory. Note that you cannot change the password in further versions of the image in a running system using **OSTree** updates. The users you create with blueprints must be used only to gain access to the created system. After you access the system, you need to create users, for example, using the **useradd** command.

To create a blueprint for RHEL for Edge Installer image, perform the following steps:

Procedure

1. Create a plain text file in the TOML format, with the following content:

   ```
   name = "blueprint-installer"
   description = "blueprint-for-installer-image"
   version = "0.0.1"

   [[customizations.user]]
   name = "user"
   description = "account"
   password = "user-password"
   key = "user-ssh-key "
   home = "path"
   groups = ["user-groups"]
   ```

   Where,

   - *blueprint-name* is the name and blueprint-text-description is the description for your blueprint.
   - *0.0.1* is the version number according to the Semantic Versioning scheme.

2. Push (import) the blueprint to the Image Builder server:

   ```
   # composer-cli blueprints push blueprint-name.toml
   ```

3. List the existing blueprints to check whether the created blueprint is successfully pushed and exists.

   ```
   # composer-cli blueprints show blueprint-name
   ```

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

   ```
   # composer-cli blueprints depsolve blueprint-name
   ```

Additional resources

- Supported Image Customizations.

5.2.3. Creating a RHEL for Edge Container image using Image Builder CLI

To create a RHEL for Edge Container image using Image Builder command-line interface, ensure that you have met the following prerequisites and follow the procedure.

Prerequisites

- You have created a blueprint for RHEL for Edge Container image.

Procedure

1. Create the RHEL for Edge Container image.

   ```
   # composer-cli compose start-ostree --ref rhel/8/x86_64/edge --url URL-OSTree-repository
   blueprint-name image-type
   ```

   Where,

   - *--ref* is the same value that customer used to build ostree repository
   - *--url* is the URL to the OSTree repository of the commit to embed in the image. For example, http://10.0.2.2:8080/repository/. See Setting up a web server to install RHEL for Edge image.
   - *blueprint-name* is the RHEL for Edge blueprint name.
   - *image-type* is **edge-container** for **non-network-based deployment**.
     A confirmation that the composer process has been added to the queue appears. It also shows a Universally Unique Identifier (UUID) number for the image created. Use the UUID number to track your build. Also keep the UUID number handy for further tasks.

2. Check the image compose status.

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

   The output displays the status in the following format:

   ```
   <UUID> RUNNING date blueprint-name blueprint-version image-type
   ```

   NOTE

   The image creation process takes up to 20 minutes to complete.

   To interrupt the image creation process, run:

   ```
   # composer-cli compose cancel <UUID>
   ```

   To delete an existing image, run:

   ```
   # composer-cli compose delete <UUID>
   ```

After the image is ready, it can be used for **non-network deployments**. See Creating a RHEL for Edge Container image for non-network-based deployments.

Additional resources

- Composing a RHEL for Edge image using Image Builder command-line

5.2.4. Creating a RHEL for Edge Installer image using command-line interface for non-network-based deployments

To create a RHEL for Edge Installer image that embeds the **OSTree** commit, using Image Builder command-line interface, ensure that you have met the following prerequisites and then follow the procedure.

Prerequisites

- You have created a blueprint for RHEL for Edge Installer image.
- You have created a RHEL for Edge Edge Container image and deployed it using a web server.

Procedure

1. Begin to create the RHEL for Edge Installer image.

```
# composer-cli compose start-ostree --ref rhel/8/x86_64/edge --url URL-OSTree-repository
blueprint-name image-type
```

   Where,

   - *ref* is the same value that customer used to build ostree repository
   - *URL-OSTree-repository* is the URL to the OSTree repository of the commit to embed in the image. For example, http://10.0.2.2:8080/repository/. See Creating a RHEL for Edge Container image for non-network-based deployments.
   - *blueprint-name* is the RHEL for Edge Installer blueprint name.
   - *image-type* is **edge-installer**.
     A confirmation that the composer process has been added to the queue appears. It also shows a Universally Unique Identifier (UUID) number for the image created. Use the UUID number to track your build. Also keep the UUID number handy for further tasks.

2. Check the image compose status.

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

   The command output displays the status in the following format:

```
<UUID> RUNNING date blueprint-name blueprint-version image-type
```

NOTE

The image creation process takes a few minutes to complete.

To interrupt the image creation process, run:

```
# composer-cli compose cancel <UUID>
```

To delete an existing image, run:

```
# composer-cli compose delete <UUID>
```

After the image is ready, you can use it for **non-network deployments**. See Installing the RHEL for Edge image for non-network-based deployments.

5.2.5. Downloading a RHEL for Edge Installer image using the Image Builder CLI

To download a RHEL for Edge Installer image using Image Builder command line interface, ensure that you have met the following prerequisites and then follow the procedure.

Prerequisites

- You have created a RHEL for Edge Installer image.

Procedure

1. Review the RHEL for Edge image status.

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

   The output must display the following:

   ```
   $ <UUID> FINISHED date blueprint-name blueprint-version image-type
   ```

2. Download the image.

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

   Image Builder downloads the image as an **.iso** file to the current directory.

   The UUID number and the image size is displayed alongside.

   ```
   $ <UUID>-boot.iso: size MB
   ```

The resulting image is a bootable ISO image.

Additional resources

- Deploying a RHEL for Edge image in a non-network-base environment .

5.3. SUPPORTED IMAGE CUSTOMIZATIONS

You can use several image customizations within blueprints. To make use of these options, you must initially configure the customizations in the blueprint and import (push) it to Image Builder.

NOTE

These customizations are not supported within the web console.

Select a package group

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

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

Set the image hostname

```
[customizations]
hostname = "baseimage"
```

User specifications for the resulting system image

```
[[customizations.user]]
name = "USER-NAME"
description = "USER-DESCRIPTION"
password = "PASSWORD-HASH"
key = "PUBLIC-SSH-KEY"
home = "/home/USER-NAME/"
shell = "/usr/bin/bash"
groups = ["users", "wheel"]
uid = NUMBER
gid = NUMBER
```

NOTE

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

IMPORTANT

To generate the **password hash**, you must install **python3** on your system. The following command installs the **python3** package.

```
# yum install python3
```

Replace *PASSWORD-HASH* with the actual **password hash**. To generate the **password hash**, use a command such as:

```
$ python3 -c 'import crypt,getpass;pw=getpass.getpass();print(crypt.crypt(pw) if
(pw==getpass.getpass("Confirm: ")) else exit())'
```

Replace *PUBLIC-SSH-KEY* with the actual public key.

Replace the other placeholders with suitable values.

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

Repeat this block for every user to include.

Group specifications for the resulting system image

```
[[customizations.group]]
name = "GROUP-NAME"
gid = NUMBER
```

Repeat this block for every group to include.

Set an existing users SSH key

```
[[customizations.sshkey]]
user = "root"
key = "PUBLIC-SSH-KEY"
```

NOTE

This option is only applicable for existing users. To create a user and set an SSH key, see the **User specifications for the resulting system image**customization in this section.

Append a kernel boot parameter option to the defaults

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

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

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

Define a kernel name to use in an image

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

Set the timezone and the*Network Time Protocol*(NTP) servers for the resulting system image

```
[customizations.timezone]
timezone = "TIMEZONE"
ntpservers = "NTP_SERVER"
```

If you do not set a timezone, the system uses *Universal Time, Coordinated* **(UTC)** as default. Setting NTP servers is optional.

Set the locale settings for the resulting system image

```
[customizations.locale]
languages = ["LANGUAGE"]
keyboard = "KEYBOARD"
```

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

Set the firewall for the resulting system image

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

You can use the numeric ports, or their names from the **/etc/services** file to enable lists.

Customize the firewall services

Review the available firewall services.

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

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

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

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

You can optionally customize the firewall services for the system image that you plan to create.

NOTE

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

Set which services to enable during the boot time

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

You can control which services to enable during the boot time. Some image types already have services enabled or disabled to ensure that the image works correctly and this setup cannot be overridden.

NOTE

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

Specify a custom filesystem configuration

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

- security benchmark compliance
- protection against out-of-disk errors
- performance
- consistency with existing setups
  Customize the filesystem configuration in your blueprint:

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

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

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

    NOTE

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

    If you have more than one partition, you can create images with a customized file system partition on LVM and resize those partitions at runtime. For that, you can specify a customized filesystem configuration in your blueprint and therefore create images with the desired disk layout. The default filesystem layout remains unchanged - if you use plain images without file system customization, and **cloud-init** resizes the root partition.

NOTE

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

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

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

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

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

NOTE

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

For example:

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

or

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

Additional resources

- Blueprint import fails after adding filesystem customization "size" .

5.4. ADDITIONAL RESOURCES

- Composing a RHEL for Edge image using Image Builder in RHEL web console .

CHAPTER 6. BUILDING SIMPLIFIED INSTALLER IMAGES TO PROVISION A RHEL FOR EDGE IMAGE

You can build a RHEL for Edge Simplified Installer image, which is optimized for unattended installation to a device, and provision the image to a RHEL for Edge image.

6.1. SIMPLIFIED INSTALLER IMAGE BUILD AND DEPLOYMENT

Build a RHEL for Edge Simplified Installer image by using the new image type, **edge-simplified-installer**.

To build a RHEL for Edge Simplified Installer image, provide an existing **OSTree** commit. The resulting simplified image contains a raw image that has the OSTree commit deployed. After you boot the Simplified installer ISO image, it provisions a RHEL for Edge system that you can use on a hard drive or as a boot image in a virtual machine.

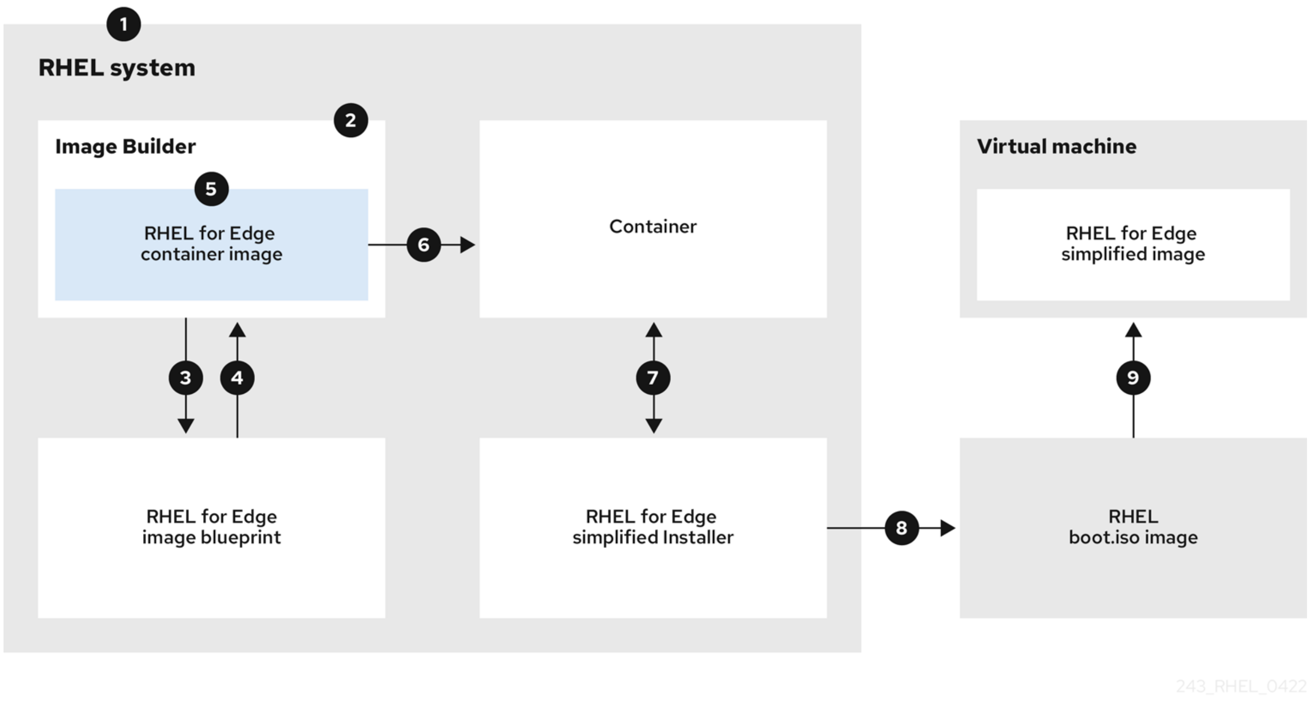
The RHEL for Edge Simplified Installer image is optimized for unattended installation to a device and supports both network-base deployment and non-network-based deployments. However, for network-based deployment, it supports only UEFI HTTP boot.

Composing and deploying a simplified RHEL for Edge image involves the following high-level steps:

1. Install and register a RHEL system
2. Install Image Builder
3. Using Image Builder, create a blueprint with customizations for RHEL for Edge Container image
4. Import the RHEL for Edge blueprint in Image Builder
5. Create a RHEL for Edge image embed in an OCI container with a webserver ready to deploy the commit as an OSTree repository
6. Create a blueprint for **edge-simplified-installer**
7. Build a simplified RHEL for Edge image
8. Download the RHEL for Edge simplified image
9. Install the raw image with virt-install

The following diagram represents the RHEL for Edge Simplified building and provisioning workflow:

Figure 6.1. Building and provisioning RHEL for Edge in network-base environment


1
RHEL system
2
Image Builder
5
RHEL for Edge container image
6
Container
Virtual machine
RHEL for Edge simplified image
3
4
7
9
RHEL for Edge image blueprint
RHEL for Edge simplified Installer
8
RHEL boot.iso image
243_RHEL_0422

6.2. CREATING A BLUEPRINT FOR A SIMPLIFIED IMAGE USING IMAGE BUILDER CLI

To create a blueprint for a simplified RHEL for Edge image, perform the following steps:

Procedure

1. Create a plain text file in the Tom’s Obvious, Minimal Language (TOML) format, with the following content:

```
name = "simplified-installer-blueprint"
description = "blueprint for the simplified installer image"
version = "0.0.1"
packages = []
modules = []
groups = []
distro = ""

[customizations]
installation_device = "/dev/vda"

[customizations.fdo]
manufacturing_server_url = "http://10.0.0.2:8080"
diun_pub_key_insecure = "true"
```

   Where,

   - *name* is the name and *description* is the description for your blueprint.
   - *0.0.1* is the version number according to the Semantic Versioning scheme.

- *Modules* describe the package name and matching version glob to be installed into the image, for example, the package name = "tmux" and the matching version glob is version = "2.9a". Notice that currently there are no differences between packages and modules.
- *Groups* are packages groups to be installed into the image, for example the **anaconda-tools** group package. If you do not know the modules and groups, leave them empty.
- *installation-device* is the customization to enable an unattended installation to your device.
- *manufacturing_server_url* is the url to to perform the initial device credential exchange.

2. Push (import) the blueprint to the Image Builder server:

```
# composer-cli blueprints push blueprint-name.toml
```

3. List the existing blueprints to check whether the created blueprint is successfully pushed and exists.

```
# composer-cli blueprints show blueprint-name
```

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

```
# composer-cli blueprints depsolve blueprint-name
```

Additional resources

- Composing a RHEL for Edge image using Image Builder command-line .

6.3. CREATING A RHEL FOR EDGE SIMPLIFIED INSTALLER IMAGE USING IMAGE BUILDER CLI

To create a RHEL for Edge Simplified image using Image Builder command-line interface, ensure that you have met the following prerequisites and then follow the procedure.

Prerequisites

- You created a blueprint for the RHEL for Edge Simplified image.
- You served an OSTree repository of the commit to embed in the image. For example, http://10.0.2.2:8080/repo. See Setting up a web server to install RHEL for Edge image .

Procedure

1. Create the bootable ISO image.

```
# composer-cli compose start-ostree \
blueprint-name \
edge-simplified-installer \
--ref rhel/8/x86_64/edge \
--url URL-OSTree-repository \
```

Where,

- *blueprint-name* is the RHEL for Edge blueprint name.
- *edge-simplified-installer* is the image-type .
- *--ref* is the reference for where your commit is going to be created.
- *--url* is the URL to the OSTree repository of the commit to embed in the image. For example, http://10.0.2.2:8080/repository/. You can either start a RHEL for Edge Container or set up a web server. See Creating a RHEL for Edge Container image for non-network-based deployments and Setting up a web server to install RHEL for Edge image .
  A confirmation that the composer process has been added to the queue appears. It also shows a Universally Unique Identifier (UUID) number for the image created. Use the UUID number to track your build. Also keep the UUID number handy for further tasks.

2. Check the image compose status.

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

   The output displays the status in the following format:

   ```
   <UUID> RUNNING date blueprint-name blueprint-version image-type
   ```

   NOTE

   The image creation processes can take up to ten minutes to complete.

   To interrupt the image creation process, run:

   ```
   # composer-cli compose cancel <UUID>
   ```

   To delete an existing image, run:

   ```
   # composer-cli compose delete <UUID>
   ```

Additional resources

- Composing a RHEL for Edge image using Image Builder command-line .

6.4. DOWNLOADING A SIMPLIFIED RHEL FOR EDGE IMAGE USING THE IMAGE BUILDER COMMAND-LINE INTERFACE

To download a RHEL for Edge image using Image Builder command line interface, ensure that you have met the following prerequisites and then follow the procedure.

Prerequisites

- You have created a RHEL for Edge image.

Procedure

1. Review the RHEL for Edge image status.

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

The output must display the following:

```
$ <UUID> FINISHED date blueprint-name blueprint-version image-type
```

2. Download the image.

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

Image Builder downloads the image as an **.iso** file at the current directory path where you run the command.

The UUID number and the image size is displayed alongside.

```
$ <UUID>-simplified-installer.iso: size MB
```

As a result, you downloaded a RHEL for Edge Simplified Installer ISO image. You can use it directly as a boot ISO to install a RHEL for Edge system.

6.5. SETTING UP AN UEFI HTTP BOOT SERVER

This section shows how to set up an **UEFI HTTP Boot** server, so that you can start to provision a RHEL for Edge Virtual Machine over network by connecting to this UEFI HTTP Boot server.

Prerequisites

- You have created the ISO simplified installer image.
- An http server that serves the ISO content.

Procedure

1. Mount the ISO image to the directory your your choice:

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

Replace **/path_directory/installer.iso** with the path to the RHEL for Edge bootable ISO image.

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

```
# cp -R /mnt/rhel8-install/* /var/www/html/
# chmod -R +r /var/www/html/httpboot/*
```

NOTE

Some copying methods can skip the **.treeinfo** file which is required for a valid installation source. Running the **cp** command for whole directories as shown in this procedure will copy **.treeinfo** correctly.

3. Update the **/var/www/html/EFI/BOOT/grub.cfg** file, by replacing:
    a. **coreos.inst.install_dev=/dev/sda** with **coreos.inst.install_dev=/dev/vda**
    b. **linux /images/pxeboot/vmlinuz** with **linuxefi /images/pxeboot/vmlinuz**
    c. **initrd /images/pxeboot/initrd.img** with **initrdefi /images/pxeboot/initrd.img**
    d. **coreos.inst.image_file=/run/media/iso/disk.img.xz** with **coreos.inst.image_url=http://{IP-ADDRESS}/disk.img.xz**
    The *IP-ADDRESS* is the ip address of this machine, which will serve as a http boot server.
4. Start the httpd service:

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

    As a result, after you set up an **UEFI HTTP Boot** server, you can install your RHEL for Edge devices by using **UEFI HTTP** boot.

6.6. DEPLOYING THE SIMPLIFIED ISO IMAGE IN A VIRTUAL MACHINE

Deploy the RHEL for Edge ISO image you generated by creating a RHEL for Edge Simplified image by using any the following installation sources:

- **UEFI HTTP Boot**
- **virt-install**

This example shows how to create a **virt-install** installation source from your ISO image for a **network-based** installation .

Prerequisites

- You have created an ISO image.
- You set up a network configuration to support UEFI HTTP boot.

Procedure

1. Set up a network configuration to support **UEFI HTTP** boot. See Setting up UEFI HTTP boot with libvirt.
2. Use the **virt-install** command to create a RHEL for Edge Virtual Machine from the UEFI HTTP Boot.

    ```
    # virt-install \
      --name edge-install-image \
      --disk path="  “, ,format=qcow2
      --ram 3072 \
      --memory 4096 \
      --vcpus 2 \
      --network network=integration,mac=mac_address \
      --os-type linux
      --os-variant rhel8 \
      --cdrom "/var/lib/libvirt/images/”ISO_FILENAME"
    ```

```
    --boot
uefi,loader_ro=yes,loader_type=pflash,nvram_template=/usr/share/edk2/ovmf/OVMF_VARS.fd,
loader_secure=no
    --virt-type kvm \
    --graphics none \
     --wait=-1
     --noreboot
```

After you run the command, the Virtual Machine installation starts.

Verification

- Log in to the created Virtual Machine.

6.7. DEPLOYING THE SIMPLIFIED ISO IMAGE FROM A USB FLASH DRIVE

Deploy the RHEL for Edge ISO image you generated by creating a RHEL for Edge Simplified image by using an **USB installation**.

This example shows how to create a **USB installation** source from your ISO image.

Prerequisites

- You have created a simplified installer image, which is an ISO image.
- You have a 8 GB USB flash drive.

Procedure

1. Copy the ISO image file to a USB flash drive.
2. Connect the USB flash drive to the port of the computer you want to boot.
3. Boot the ISO image from the USB flash drive.The boot menu shows you the following options:

```
Install Red Hat Enterprise Linux 8
Test this media & install Red Hat Enterprise Linux 8
```

4. Choose Install Red Hat Enterprise Linux 8. This starts the system installation.

Additional resources

- Booting the installation.

CHAPTER 7. AUTOMATICALLY PROVISIONING AND ONBOARDING RHEL FOR EDGE DEVICES WITH FDO

You can build a RHEL for Edge Simplified Installer image, and provision it to a RHEL for Edge image. The FIDO device onboarding (FDO) process automatically provision and onboard your Edge devices, and exchange data with other devices and systems connected on the networks.

IMPORTANT

Red Hat provides the **FDO** process as a Technology Preview feature and should run on secure networks. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process. See Technology Preview Features Support Scope on the Red Hat Customer Portal for information about the support scope for Technology Preview features.

7.1. THE FIDO DEVICE ONBOARDING (FDO) PROCESS

Device onboarding is the process that:

- Provisions and onboards a physical device.
- Automatically configures credentials for this device.
- Enables this device to securely connect and interact on the network.

With FIDO device onboarding (FDO), you can perform a secure device onboarding by adding new devices into your IOT architecture. This includes the specified device configuration that needs to be trusted and integrated with the rest of the running systems and to deploy new systems that are ready to be used. The FDO authentication is an automatic onboarding process that is triggered by the installation of a new device to securely onboard a device. The FDO protocol solves the trust and chain of ownership along with the automation needed to securely onboard a device at scale. It performs device initialization at the manufacturing stage and late device binding for its actual use. This means that actual binding of the device to a management system happens on the first boot of the device without requiring manual configuration on the device. By using the FDO protocol, you have support for automated secure devices onboarding, that is, zero touch installation and onboarding that does not need any specialized person at the edge location. After the device is onboarded, you can connect to it and apply patches, updates, and rollbacks.

With FDO, you can benefit from the following:

- FDO is a secure and simple way to enroll a device to a management platform. Instead of embedding a Kickstart configuration to the image, FDO applies the customization, such as inclusion of sensitive data as credentials, keys or certificates directly to the ISO image.
- FDO solves the issue of late binding to a device, enabling any sensitive data to be shared over a secure FDO channel.
- FDO cryptographically identifies the system identity and ownership before enrolling and passing the configuration and other secrets to the system. That enables non-technical users to power-on the system.

To build a RHEL for Edge Simplified Installer image and automatically onboard it, provide an existing OSTree commit. The resulting simplified image contains a raw image that has the OSTree commit deployed. After you boot the Simplified installer ISO image, it provisions a RHEL for Edge system that you can use on a hard disk or as a boot image in a virtual machine.

The RHEL for Edge Simplified Installer image is optimized for unattended installation to a device and supports both network-base deployment and non-network-based deployments. However, for network-based deployment, it supports only UEFI HTTP boot.

The FDO protocol is based on the following servers:

- Manufacturing server
  This server is at the manufacturer server location. The manufacturing server:

  1. Signs the device.

  2. Creates a voucher that is used to set the ownership of the device, later in the process.

  3. Binds the device to a specific management platform.

- Rendezvous server
  This server is at the owner server location or at the platform where the Device management system will be located, for example a cloud. The Rendezvous server:

  1. Gets the voucher generated by the manufacturing server during the first device boot.

  2. Matches the device UUID with a target platform and provides information to the device about which Owner server endpoint this device must use.

- Owner management server
  This server is at the owner server location or at the platform where the Device will be deployed. The Owner management server:

  1. Creates a secure channel between the device and the Owner server after the device authentication.

  2. Uses the secure channel to send the required information, such as files and scripts for the onboarding automation to the device.

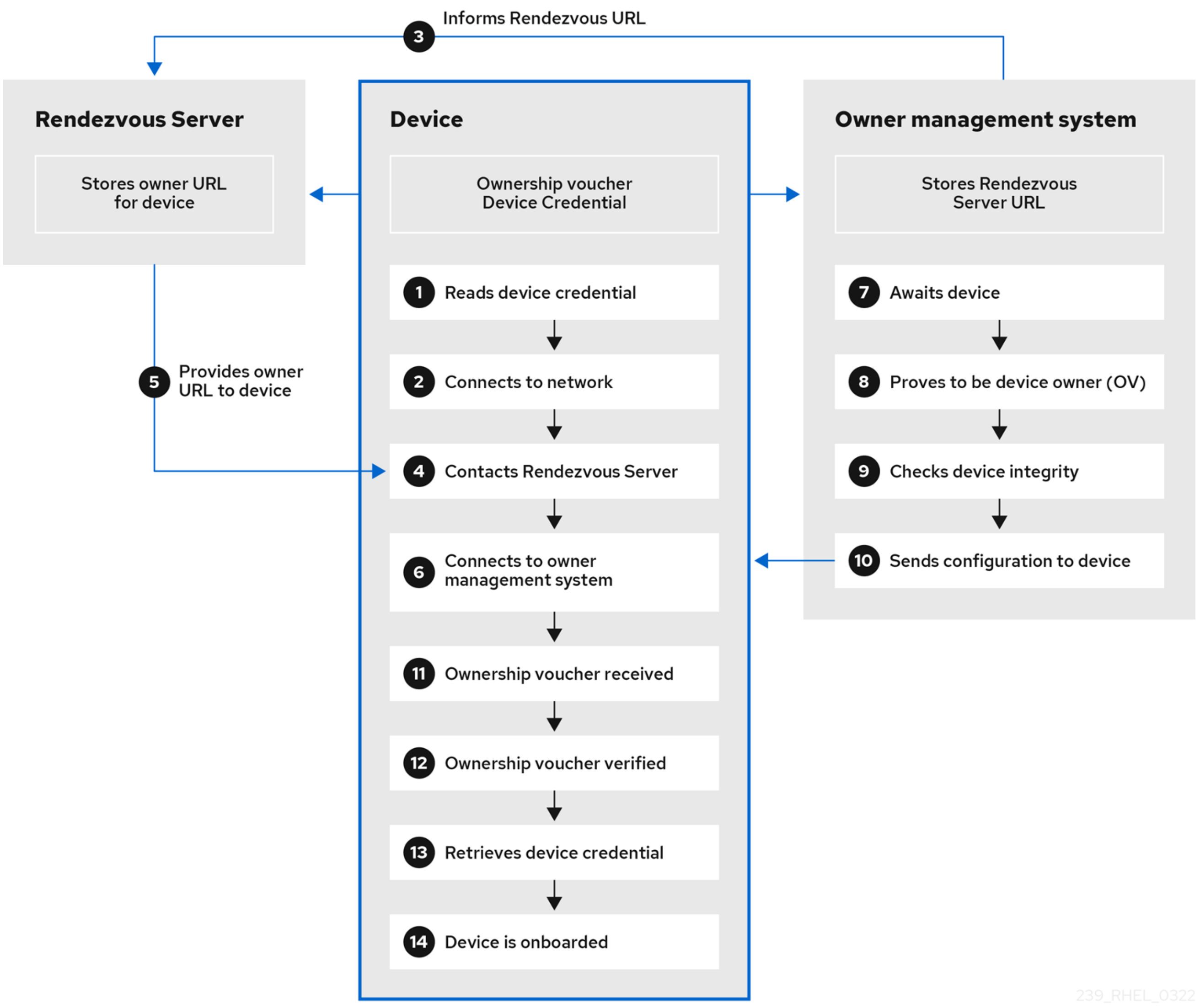
- Device client
  This is the server installed on the device. The Device client

  1. Starts the queries to the multiple servers where the onboarding automation will be executed.

  2. Uses TCP/IP protocols to communicate with the servers.

The following diagram represents the FIDO device onboarding workflow:

Figure 7.1. Deploying RHEL for Edge in non-network environment


Informs Rendezvous URL
3
Rendezvous Server
Stores owner URL for device
5 Provides owner URL to device
Device
Ownership voucher Device Credential
1 Reads device credential
2 Connects to network
4 Contacts Rendezvous Server
6 Connects to owner management system
11 Ownership voucher received
12 Ownership voucher verified
13 Retrieves device credential
14 Device is onboarded
Owner management system
Stores Rendezvous Server URL
7 Awaits device
8 Proves to be device owner (OV)
9 Checks device integrity
10 Sends configuration to device
239_RHEL_0322

At the **Manufacturer server**, the device gets the FDO credentials, a set of certificates and keys to be installed on the operating system, and the Rendezvous server endpoint (URL). It also gets the Ownership Voucher, that is maintained separately in case you need to change the owner assignment.

1. Device client reads device credential
2. Device client connects to network
3. At an early point, the Owner management system informs the Manufacturer Rendezvous server about the location of the Owner management system
4. After connecting to the network, the Device client contacts the Rendezvous Server
5. The Rendezvous Server sends the owner endpoint URL to the Device Client, and registers the device. This action connects and boots the device.
6. The Device client connects to the Owner management system shared by the Rendezvous Server, proves that it is the correct device by signing a statement with a device key
7. The Owner management system prove itself by signing a statement with the last key of the Owner Voucher

8. The Owner management system provides the configuration for the device, which the Device client stores for example, in an SSH key

9. The Device client receives and verify the Ownership voucher

10. Then, the Device client retrieves its device credentials

11. After that, the Owner management system reports the Device client as onboarded
The entire FDO process is done and no longer in use in this device.

7.2. AUTOMATICALLY PROVISIONING AND ONBOARDING RHEL FOR EDGE DEVICES

Automatically provisioning and onboarding a RHEL for Edge device involves the following high-level steps:

1. Install and register a RHEL system

2. Install Image Builder

3. Using Image Builder, create a blueprint with customizations for RHEL for Edge Container image

4. Import the RHEL for Edge blueprint in Image Builder

5. Create a RHEL for Edge image embed in an OCI container with a webserver ready to deploy the commit as an OSTree repository

6. Create a blueprint for **edge-simplified-installer** with customizations for storage device path and FDO customizations

```
name = "fdo"
description = "FDO blueprint"
version = "0.0.1"
packages = []
modules = []
groups = []
distro = ""

[customizations]
installation_device = "/dev/vda"

[customizations.fdo]
manufacturing_server_url = "http://10.0.0.2:8080"
diun_pub_key_insecure = "true"
```

7. Build a simplified installer RHEL for Edge image

8. Download the RHEL for Edge simplified installer image

9. Install the simplified installer ISO image to a device. The FIDO FDO client runs on the Simplified Installer ISO and the UEFI directory structure makes the image bootable.

10. The network configuration enables the device to reach out to the manufacturing server to perform the initial device credential exchange.

11. After the system reaches the endpoint, the device credentials are created for the device.

12. The onboard server uses the device credential to authenticate against the onboarding server. .The onboarding server passes the configuration to the device/system: After it connects to the system, it connects to their onboarding server, receives the configuration.

13. The onboarding server provides the device with an SSH key and installs the system.

14. Then, it reboots the system and encrypts it with a strong key stored at TPM.

15. You can login to the system with the credentials from the blueprint you created and check the configuration that was created into the Simplified Installer ISO image.

Additional resources

- FDO automatic onboarding technologies

7.3. GENERATING KEY AND CERTIFICATES

To run the FIDO device onboarding (FDO) infrastructure, you need to generate keys and certificates. FDO generates these keys and certificates to configure the manufacturing server. FDO automatically generates the certificates and **.yaml** configuration files when you install the services, and re-creating them is optional. After you install and start the services, it runs with the default settings.

IMPORTANT

Red Hat provides the **fdo-admin-tool generate-key-and-cert** tool as a Technology Preview feature and should run on secure networks. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process. See Technology Preview Features Support Scope on the Red Hat Customer Portal for information about the support scope for Technology Preview features.

Prerequisites

- You installed the **fdo-admin-cli** RPM package

Procedure

1. Create a directory for the keys and certificates:

```
$ mkdir /etc/fdo/keys
```

2. Generate the keys and certificates in the directory you created:

```
$ for i in "diun" "manufacturer" "device_ca" "owner"; do fdo-admin-tool generate-key-and-cert
$i; done
$ ls keys
device_ca_cert.pem device_ca_key.der diun_cert.pem diun_key.der manufacturer_cert.pem
manufacturer_key.der owner_cert.pem owner_key.der
```

NOTE

If you used the source code and compiled it, the correct path is **./target/debug/fdo-admin-tool** or **./target/debug/fdo-admin-tool**, depending on your build options.

3. Check the key and certificates that were created:

```
$ tree keys
```

You can see the following output:

```
– device_ca_cert.pem
– device_ca_key.der
– diun_cert.pem
– diun_key.dre
– manufacturer_cert.pem
– manufacturer_key.der
– owner_cert.pem
– owner_key.pem
```

Additional resources

- The **fdo-admin-tool generate-key-and-cert –help**

7.4. INSTALLING THE MANUFACTURING SERVER PACKAGE

The **manufacturing server** RPM package provides the credentials to securely onboard the device. During the device installation, the manufacturing server requests for the Rendezvous server to provide the device credential authentication against the server and install the device credentials to the installed system.

IMPORTANT

Red Hat provides the **fdo-manufacturing-server** tool as a Technology Preview feature and should run on secure networks. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process. See Technology Preview Features Support Scope on the Red Hat Customer Portal for information about the support scope for Technology Preview features.

To install the **manufacturing server** RPM package, complete the following steps:

Procedure

1. Install the **fdo-admin-cli** package:

```
# yum install -y fdo-admin-cli
```

2. Install the **manufacturing server** RPM package:

```
# yum install fdo-manufacturing-server --refresh
```

■

3. Check if the files were correctly installed:

   ```
   $ ls /usr/share/doc/fdo
   ```

   You can see the following output:

   ```
   Output:
   manufacturing server.yml
   Owner-onboarding-server.yml
   rendezvous-server.yml
   ```

4. Optional: Check the content of each file, for example:

   ```
   $ cat /usr/share/doc/fdo/manufacturing-server.yml
   ```

5. Configure the manufacturing server. You must provide the following:

   - The manufacturing server URL
   - The IP address or DNS name for the rendezvous server
   - The path to the keys and certificates you generated. See Generating key and certificates section.

6. After you install the RHEL for Edge network simplified image to your device, ensure that the manufacturer server is running on a Podman container. The manufacturing server takes care of the creating and enabling device credentials on the new device.

   ```
   $ cat /usr/share/doc/fdo/manufacturing-server.yml
   ```

Additional resources

- The manufacturing-server.yml example
- FDO automatic onboarding terminology

7.5. AUTOMATICALLY ONBOARDING AN RHEL FOR EDGE DEVICE BY USING FDO AUTHENTICATION

To prepare your device to automatically onboard a RHEL for Edge device, complete the following steps:

Prerequisites

- You built and served an OStree container.
- Device assembled and provisioned. This example uses a VM machine, but you can use it in a real device.
- You are running a UEFI HTTP Boot server.
- You installed the **fdo-manufacturing-server** RPM package. Run:

```
# yum install -y fdo-admin-cli
```

Procedure

1. Run the installation using the ISO Simplified image. You can install it from a CD-ROM or from a USB flash drive, for example.
   The installation runs the ISO Simplified Installer image, where the FDO client runs and the UEFI directory structure makes the image bootable, to burn the raw image in the ISO.

2. Verify through the terminal that the device has reached the manufacturing service to perform the initial device credential exchange and produced an ownership voucher:

   ```
   $ ls directory-path/ownership_voucher/
   ```

   The output should show the **ownership_voucher** ID to indicate that the correct device credentials were added to the device.

   The onboarding server uses the device credential to authenticate against the onboarding server. It then passes the configuration to the device. After the device receives the configuration from the onboarding server, it receives an SSH key and installs the operating system on the device. Finally, the system automatically reboots, encrypts it with a strong key stored at TPM.

   After the device automatically reboots, the device contacts the onboarding server to be onboarded and the user credentials are automatically provisioned by FDO.

Verification

After the device automatically reboots, you can log in to the device with the credentials you created for the blueprint.

1. Log in to the device by providing the username and password you created for the blueprint.

2. Optional: verify that the configuration that was created into the raw image.

Additional resources

- Setting up an UEFI HTTP Boot server
- Deploying the Simplified ISO image from a USB flash drive

CHAPTER 8. DEPLOYING A RHEL FOR EDGE IMAGE IN A NETWORK-BASE ENVIRONMENT

You can deploy a RHEL for Edge image using the RHEL installer graphical user interface or a Kickstart file. The overall process for deploying a RHEL for Edge image depends on whether your deployment environment is network-based or non-network-based.

NOTE

To deploy the images on bare metal, use a Kickstart file.

Network-based deployments

Deploying a RHEL for Edge image in a network-based environment involves the following high-level steps:

a. Extract the image file contents.

b. Set up a web server

c. Install the image

8.1. EXTRACTING RHEL FOR EDGE IMAGE COMMIT

After you download the commit, extract the **.tar** file and note the ref name and the commit ID.

The downloaded commit file consists of a **.tar** file with an **OSTree** repository. The **OSTree** repository has a commit and a **compose.json** file.

The **compose.json** file has information metadata about the commit with information such as the "Ref", the reference ID and the commit ID. The commit ID has the RPM packages.

To extract the package contents, perform the following the steps:

Prerequisites

- Create a Kickstart file or use an existing one.

Procedure

1. Extract the downloaded image **.tar** file:

```
# tar xvf <UUID>-commit.tar
```

2. Go to the directory where you have extracted the **.tar** file.
   It has a **compose.json** file and an OSTree directory. The **compose.json** file has the commit number and the **OSTree** directory has the RPM packages.

3. Open the **compose.json** file and note the commit ID number. You need this number handy when you proceed to set up a web server.
   If you have the **jq** JSON processor installed, you can also retrieve the commit ID by using the **jq** tool:

```
# jq '.["ostree-commit"]' < compose.json
```

4. List the RPM packages in the commit.

```
# rpm-ostree db list rhel/8/x86_64/edge --repo=repo
```

5. Use a Kickstart file to run the RHEL installer. Optionally, you can use any existing file or can create one by using the Kickstart Generator tool.
   In the Kickstart file, ensure that you include the details about how to provision the file system, create a user, and how to fetch and deployment the RHEL for Edge image. The RHEL installer uses this information during the installation process.

   The following is a Kickstart file example:

```
lang en_US.UTF-8
keyboard us
timezone Etc/UTC --isUtc
text
zerombr
clearpart --all --initlabel
autopart
reboot
user --name=core --group=wheel
sshkey --username=core "ssh-rsa AAAA3Nza…."
rootpw --lock
network --bootproto=dhcp

ostreesetup --nogpg --osname=rhel --remote=edge --url=https://mirror.example.com/repo/ --
ref=rhel/8/x86_64/edge
```

   The OStree-based installation uses the **ostreesetup** command to set up the configuration. It fetches the OSTree commit, by using the following flags:

   - **--nogpg** - Disable GNU Privacy Guard (GPG) key verification.
   - **--osname** - Management root for the operational system installation.
   - **--remote** - Management root for the operational system installation
   - **--url** - URL of the repository to install from.
   - **--ref** - Name of the branch from the repository that the installation uses.
   - **--url=https://mirror.example.com/repo/** - is the address of the host system where you extracted the edge commit and served over **nginx**. You can use the address to reach the host system from the guest computer.
     For example, if you extract the commit image in the **/var/www/html** directory and serve the commit over **nginx** on a computer whose hostname is **www.example.com**, the value of the **--url** parameter is **http://www.example.com/repo**.

     NOTE

     Use the http protocol to start a service to serve the commit, because https is not enabled on the Apache HTTP Server.

Additional resources

- Downloading a RHEL for Edge image
- Creating Kickstart files

8.2. SETTING UP A WEB SERVER TO INSTALL RHEL FOR EDGE IMAGES

After you have extracted the RHEL for Edge image contents, set up a web server to provide the image commit details to the RHEL installer by using HTTP.

The following example provides the steps to set up a web server by using a container.

Prerequisites

- You have installed Podman on your system. See How do I install Podman in RHEL

Procedure

1. Create the **nginx** configuration file with the following instructions:

```
events {

}

http {
    server{
        listen 8080;
        root /usr/share/nginx/html;
                }
         }

pid /run/nginx.pid;
daemon off;
```

2. Create a Dockerfile with the following instructions:

```
FROM registry.access.redhat.com/ubi8/ubi
RUN yum -y install nginx && yum clean all
COPY kickstart.ks /usr/share/nginx/html/
COPY repo /usr/share/nginx/html/
COPY nginx /etc/nginx.conf
EXPOSE 8080
CMD ["/usr/sbin/nginx", "-c", "/etc/nginx.conf"]
ARG commit
ADD ${commit} /usr/share/nginx/html/
```

Where,

- ***kickstart.ks*** is the name of the Kickstart file from the RHEL for Edge image. The Kickstart file includes directive information. To help you manage the images later, it is advisable to include the checks and settings for Greenboot checks. For that, you can update the Kickstart file to include the following settings:

```
lang en_US.UTF-8
keyboard us
timezone Etc/UTC --isUtc
text
zerombr
clearpart --all --initlabel
autopart
reboot
user --name=core --group=wheel
sshkey --username=core "ssh-rsa AAAA3Nza...."

ostreesetup --nogpg --osname=rhel --remote=edge
--url=https://mirror.example.com/repo/
--ref=rhel/8/x86_64/edge

%post
cat << EOF > /etc/greenboot/check/required.d/check-dns.sh
#!/bin/bash

DNS_SERVER=$(grep nameserver /etc/resolv.conf | cut -f2 -d" ")
COUNT=0

# check DNS server is available
ping -c1 $DNS_SERVER
while [ $? != '0' ] && [ $COUNT -lt 10 ]; do




    COUNT++
echo "Checking for DNS: Attempt $COUNT ."
sleep 10
ping -c 1 $DNS_SERVER
done
EOF
%end
```

Any HTTP service can host the OSTree repository, and the example, which uses a container, is just an option for how to do this. The Dockerfile performs the following tasks:

a. Uses the latest Universal Base Image (UBI)

b. Installs the web server (nginx)

c. Adds the Kickstart file to the server

d. Adds the RHEL for Edge image commit to the server

3. Build a Docker container

```
# podman build -t name-of-container-image --build-arg commit=uuid-commit.tar .
```

4. Run the container

```
# podman run --rm -d -p port:8080 localhost/name-of-container-image
```

As a result, the server is set up and ready to start the RHEL Installer by using the **commit.tar** repository and the Kickstart file.

8.3. DOWNLOADING RHEL BOOT.ISO IMAGE

You can download a Red Hat Boot ISO image from the Red Hat Customer Portal. The Red Hat Boot ISO image is used to launch the RHEL installer. The installer fetches the Kickstart file that you provide for installing RHEL for Edge images.

Prerequisites

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

Procedure

1. Open a browser and access https://access.redhat.com/downloads.
2. Under Infrastructure Management, click the **Red Hat Enterprise Linux 8** Product.
3. Click the **Download Now** button for the option "Red Hat Enterprise Linux 8 Boot ISO"

Additional resources

- Downloading a RHEL installation ISO image .

8.4. INSTALLING THE RHEL FOR EDGE IMAGE USING A KICKSTART FILE

To install the RHEL for Edge image that uses a Kickstart file, use the web server. The web server uses the RHEL for Edge image **commit.tar** repository and the Kickstart file to start the RHEL Installer.

Prerequisites

- The server to fetch the commit in the RHEL Installer is available and running.
- A **.qcow2** disk image to install the commit you created. See Creating a system image with Image Builder in the CLI.

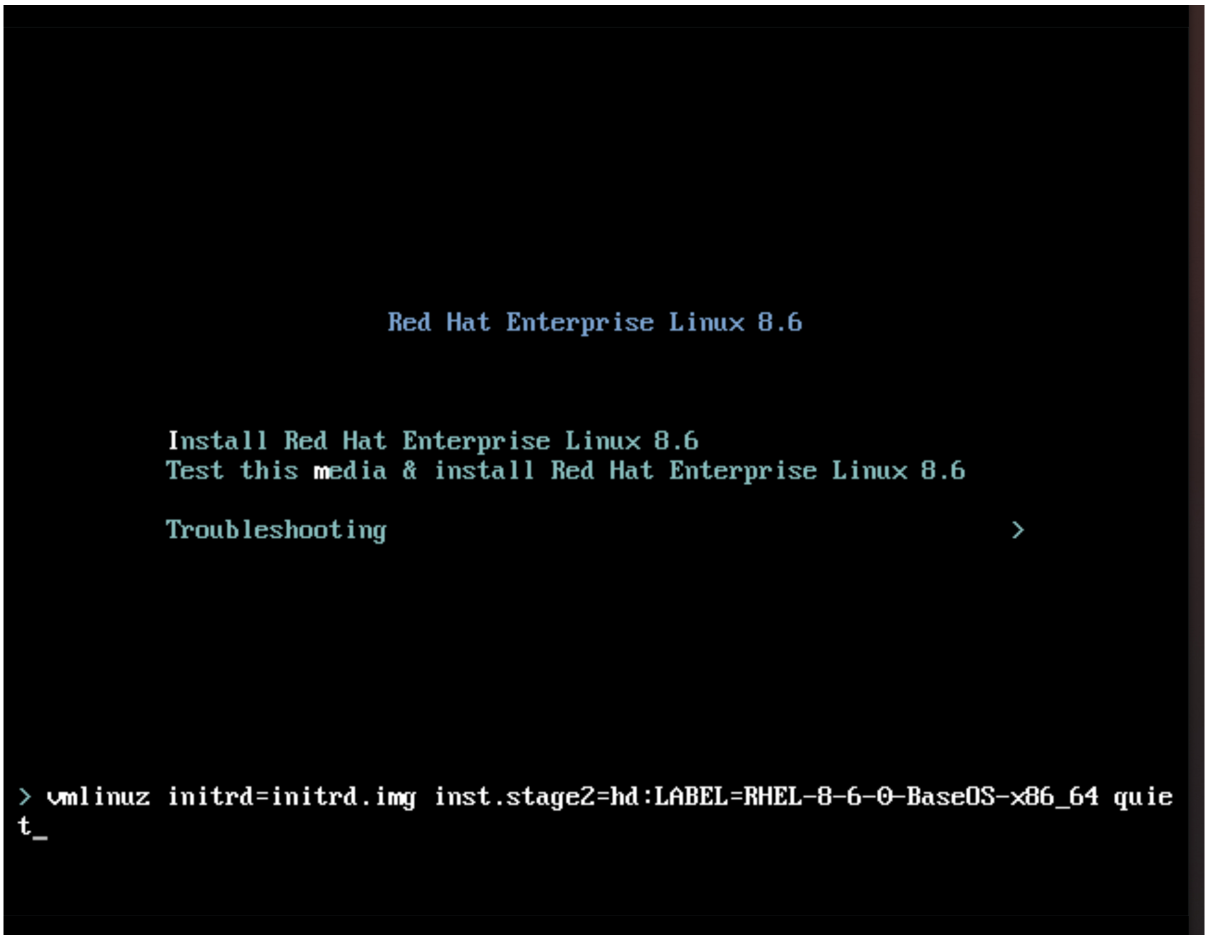
Procedure

1. Run the RHEL Anaconda Installer by using **libvirt virt-install**:

```
virt-install \
--name rhel-edge-test-1 \
--memory 2048 \
--vcpus 2 \
--disk path=prepared_disk_image.qcow2,format=qcow2,size=8 \
--os-variant rhel8 \
--cdrom /home/username/Downloads/rhel-8-x86_64-boot.iso
```

2. On the installation screen:

Figure 8.1. Red Hat Enterprise Linux boot menu



```
                    Red Hat Enterprise Linux 8.6


        Install Red Hat Enterprise Linux 8.6
        Test this media & install Red Hat Enterprise Linux 8.6

        Troubleshooting                                      >




> vmlinuz initrd=initrd.img inst.stage2=hd:LABEL=RHEL-8-6-0-BaseOS-x86_64 quie
t_
```

   a. Press the **e** key to add an additional kernel parameter:

   ```
   inst.ks=http://edge_device_ip:port/kickstart.ks
   ```

   The kernel parameter specifies that you want to install RHEL by using the Kickstart file and not the RHEL image contained in the RHEL Installer.

   b. After adding the kernel parameters, press **Ctrl**+**X** to boot the RHEL installation by using the Kickstart file.
   The RHEL Installer starts, fetches the Kickstart file from the server (HTTP) endpoint and executes the commands, including the command to install the RHEL for Edge image commit from the HTTP endpoint. After the installation completes, the RHEL Installer prompts you for login details.

Verification

1. On the Login screen, enter your user account credentials and click **Enter**.

2. Verify whether the RHEL for Edge image is successfully installed.

```
$ rpm-ostree status
```

The command output provides the image commit ID and shows that the installation is successful.

Following is a sample output:

```
State: idle
Deployments:
* ostree://edge:rhel/8/x86_64/edge
    Timestamp: 2020-09-18T20:06:54Z
   Commit: 836e637095554e0b634a0a48ea05c75280519dd6576a392635e6fa7d4d5e96
```

Additional resources

- How to embed a Kickstart file into an ISO image .
- Booting the installation.

CHAPTER 9. DEPLOYING A RHEL FOR EDGE IMAGE IN A NON-NETWORK-BASE ENVIRONMENT

The RHEL for Edge Container (**.tar**) in combination with the RHEL for Edge Installer (**.iso**) image type result in a ISO image. The ISO image can be used in disconnected environments during the image deployment to a device. However, network access might require network access to build the different artifacts.

Deploying a RHEL for Edge image in a non-network-based environment involves the following high-level steps:

1. Download the RHEL for Edge Container. See Downloading a RHEL for Edge image for information about how to download the RHEL for Edge image.
2. Load the RHEL for Edge Container image into Podman
3. Run the RHEL for Edge Container image into Podman
4. Load the RHEL for Edge Installer blueprint
5. Build the RHEL for Edge Installer image
6. Prepare a **.qcow2** disk
7. Boot the Virtual Machine (VM)
8. Install the image

9.1. CREATING A RHEL FOR EDGE CONTAINER IMAGE FOR NON-NETWORK-BASED DEPLOYMENTS

You can build a running container by loading the downloaded RHEL for Edge Container OSTree commit into Podman. For that, follow the steps:

Prerequisites

- You created and downloaded a RHEL for Edge Container OSTree commit.
- You have installed **Podman** on your system. See How do I install Podman in RHEL .

Procedure

1. Navigate to the directory where you have downloaded the RHEL for Edge Container OSTree commit.
2. Load the RHEL for Edge Container OSTree commit into **Podman**.

   ```
   $ sudo podman load -i UUID-container.tar
   ```

   The command output provides the image ID, for example: **@8e0d51f061ff1a51d157804362bc875b649b27f2ae1e66566a15e7e6530cec63**
3. Tag the new RHEL for Edge Container image, using the image ID generated by the previous step.

```
$ sudo podman tag image-ID localhost/edge-container
```

The **podman tag** command assigns an additional name to the local image.

4. Run the container named **edge-container**.

```
$ sudo podman run -d --name=edge-container -p 8080:8080 localhost/edge-container
```

The **podman run -d --name=edge-container** command assigns a name to your container-based on the **localhost/edge-container** image.

5. List containers:

```
$ sudo podman ps -a
CONTAINER ID  IMAGE                          COMMAND CREATED     STATUS
PORTS   NAMES
2988198c4c4b  …./localhost/edge-container   /bin/bash  3 seconds ago  Up 2 seconds ago
edge-container
```

As a result, **Podman** runs a container that serves an OSTree repository with the RHEL for Edge Container commit.

9.2. CREATING A RHEL FOR EDGE INSTALLER IMAGE FOR NON-NETWORK-BASED DEPLOYMENTS

After you have built a running container to serve a repository with the **RHEL for Edge Container** commit, create an **RHEL for Edge Installer (.iso)** image. The **RHEL for Edge Installer (.iso)** pulls the commit served by **RHEL for Edge Container (.tar)**. After the **RHEL for Edge Container** commit is loaded in Podman, it exposes the **OSTree** in the URL format.

To create the RHEL for Edge Installer image in web console, follow the steps:

Prerequisites

- You created a blueprint for RHEL for Edge image. See Creating a blueprint for the RHEL for Edge Installer image using Image Builder in RHEL web console
- On a RHEL system, you have accessed the Image Builder dashboard.

Procedure

1. On the Image Builder dashboard, for the RHEL for Edge Installer blueprint that you have created for RHEL for Edge image, click **Create Image**.
2. On the **Create Image** window, perform the following steps:
   a. In the **Repository** textbox, specify the RHEL for Edge container OSTree URL to embed in the image. For example, http://localhost:8080/repo
   b. In the **Ref** textbox, specify the same reference as you provided during the creation of the RHEL for Edge Container commit to embed in the image. For example, **rhel/edge/test**.
   c. Click **Create**.

Image Builder pulls the commit that is being served by the running container during the image build.

After the image build is complete, you can download the resulting ISO image.

3. Download the image. See Downloading a RHEL for Edge image.

Additional resources

- Creating a RHEL for Edge Installer image using command-line interface for non-network-based deployments

9.3. INSTALLING THE RHEL FOR EDGE IMAGE FOR NON-NETWORK-BASED DEPLOYMENTS

To install the RHEL for Edge image, follow the steps:

Prerequisites

- You created a RHEL for Edge Installer ISO image.
- You stopped the running container.
- A disk image to install the commit you created.
- You installed the **edk2-ovmf** package.
- You installed the **virt-viewer** package.
- You customized your blueprint with a user account. See Creating an administrative user account for a RHEL for Edge image blueprint.

> **WARNING**
>
> If you do not define a user account customization in your blueprint, you will not be able to login to the ISO image.

Procedure

1. Create a **qcow** VM disk file to install the (.iso) image. That is an image of a hard drive for the virtual machine (VM). For example:

```
$ qemu-img create -f qcow2 diskfile.qcow2 20G
```

2. Use the **virt-install** command to boot the VM using the disk as a drive and the installer ISO as a CD-ROM. For example:

```
$ virt-install \
--boot uefi \
--name VM_NAME
```

```
--memory 2048 \
--vcpus 2 \
--disk path=diskfile.qcow2
--cdrom /var/lib/libvirt/images/UUID-installer.iso \
--os-variant rhel9.0
```

This command instructs **virt-install** to:

- Instructs the VM to use UEFI to boot, instead of the BIOS.
- Mount the installation ISO.
- Use the hard drive image created in the first step.
  It gives you an Anaconda Installer. The RHEL Installer starts, loads the Kickstart file from the ISO and executes the commands, including the command to install the RHEL for Edge image commit. Once the installation is complete, the installer prompts for login details.

> **NOTE**
>
> Anaconda is preconfigured to use the Container commit during the installation. However, you need to set up system configurations, such as disk partition, timezone, between others.

3. Connect to Anaconda GUI with **virt-viewer** to setup the system configuration:

```
$ virt-viewer --connect qemu:///system --wait VM_NAME
```

4. Reboot the system to finish the installation.
5. On the login screen, specify your user account credentials and click **Enter**.

Verification steps

1. Verify whether the RHEL for Edge image is successfully installed.

```
$  rpm-ostree status
```

The command output provides the image commit ID and shows that the installation is successful.

CHAPTER 10. MANAGING RHEL FOR EDGE IMAGES

To manage the RHEL for Edge images, you can perform any of the following administrative tasks:

- Edit the RHEL for Edge image blueprint using Image Builder in RHEL web console
- Edit the RHEL for Edge image blueprint using Image Builder command-line
- Update the RHEL for Edge images
- Configure rpm-ostree remotes on nodes, to update node policy
- Restore RHEL for Edge images manually or automatically using a Greenboot

10.1. EDITING A RHEL FOR EDGE IMAGE BLUEPRINT USING IMAGE BUILDER IN RHEL WEB CONSOLE

You can edit the RHEL for Edge image blueprint to:

- add additional components that you may require
- modify the version of any existing component
- remove any existing component

10.1.1. Adding a component to RHEL for Edge image blueprint using Image Builder in RHEL web console

To add a component to a RHEL for Edge image blueprint, ensure that you have met the following prerequisites and then follow the procedure to edit the corresponding blueprint.

Prerequisites

- On a RHEL system, you have accessed the Image Builder dashboard.
- You have created a blueprint for RHEL for Edge image.

Procedure

1. On the Image Builder dashboard, click the RHEL for Edge image blueprint that you want to edit. To search for a specific blueprint, enter the blueprint name in the Filter By Name text box, and then press **Enter**.
2. On the upper right side of the blueprint, click **Edit Packages**. The view changes to the Edit Packages mode.
3. Enter the component name that you want to add in the Filter By Name text box, and then press Enter.
   A list with the component name appears.
4. Click the **+** sign adjacent to the component.
   The component is added to the Blueprint.
5. Click **Commit**.

The blueprint updates are saved, and a message with the pending commit appears.

6. On the summary dialogue box, review the changes and then click **Commit**.
   A message confirming the successful commit appears.

   As a result, a new version of the blueprint is created and the right pane lists the latest components.

10.1.2. Changing the version of an existing component in a RHEL for Edge image blueprint using the RHEL web console

You had selected a default (latest) version or had chosen a version for the components that you included in the blueprint. If required, you can now change the version for any component that you might want to.

To do so, ensure that you have met the following prerequisites and then follow the procedure to change the version of the component in the corresponding blueprint.

Prerequisites

- On a RHEL system, you have accessed the Image Builder dashboard.
- You have created a blueprint for RHEL for Edge image.
- You have added at least one component to the RHEL for Edge blueprint.

Procedure

1. On the Image Builder dashboard, click the blueprint that you want to edit.
   To search for a specific blueprint, enter the blueprint name in the Filter By Name text box, and then press **Enter**.
2. On the upper right side of the blueprint, click **Edit Packages**.
   The view changes to the Edit Packages mode, and the right panel lists the component names that are currently committed to the blueprint.
3. Click the component name.
4. Select the desired version from the Component Options Version dropdown list.
5. Click **Apply Changes**.
   The change is saved and the right pane lists the latest changes.
6. Click **Commit**.
   The new version is saved in the blueprint. A message with pending commits appears.
7. On the summary dialogue box, review the changes and then click **Commit**.
   A message confirming the successful commit appears.

   As a result a new version of the blueprint is created and the right pane lists the latest components.

10.1.3. Removing a component from RHEL for Edge image blueprint using Image Builder in RHEL web console

To remove one or more unwanted components from a RHEL for Edge image blueprint that you created, ensure that you have met the following prerequisites and then follow the procedure.

Prerequisites

- On a RHEL system, you have accessed the Image Builder dashboard.
- You have created a blueprint for RHEL for Edge image.
- You have added at least one component to the RHEL for Edge blueprint.

Procedure

1. On the Image Builder dashboard, click the blueprint that you want to edit.
   To search for a specific blueprint, enter the blueprint name in the Filter By Name text box, and then press Enter.
2. On the upper right side of the blueprint, click **Edit Packages**.
   The view changes to the Edit Packages mode. The right panel lists the component names that are currently committed to the blueprint.
3. From the More Options menu, click **Remove**.
   Optionally, click the component name and then click **Remove**.
4. Click **Commit**.
   A message with pending commits appears.
5. Review your changes and then click **Commit**.
   A message confirming the successful commit appears.

   As a result, a new version of the blueprint is created and the right pane lists the latest components.

10.1.4. Editing a RHEL for Edge image blueprint using command-line interface

You can change the specifications for your RHEL for Edge image blueprint using Image Builder command-line. To do so, ensure that you have met the following prerequisites and then follow the procedure to edit the corresponding blueprint.

Prerequisites

- You have access to the Image Builder command-line.
- You have created a RHEL for Edge image blueprint.

Procedure

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

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

2. Edit the **BLUEPRINT-NAME.toml** file with a text editor of your choice and make your changes.
   Before finishing with the edits, make sure the file is a valid blueprint:
3. Increase the version number.

Ensure that you use a Semantic Versioning scheme.

NOTE

if you do not change the version, the patch component of the version is increased automatically.

4. Check if the contents are valid TOML specifications. See the TOML documentation for more information.

NOTE

TOML documentation is a community product and is not supported by Red Hat. You can report any issues with the tool at https://github.com/toml-lang/toml/issues.

5. Save the file and close the editor.

6. Push (import) the blueprint back into Image Builder command-line:

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

NOTE

When pushing the blueprint back into the Image Builder command-line, provide the file name including the **.toml** extension.

7. Verify that the contents uploaded to Image Builder match your edits:

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

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

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

10.2. UPDATING RHEL FOR EDGE IMAGES

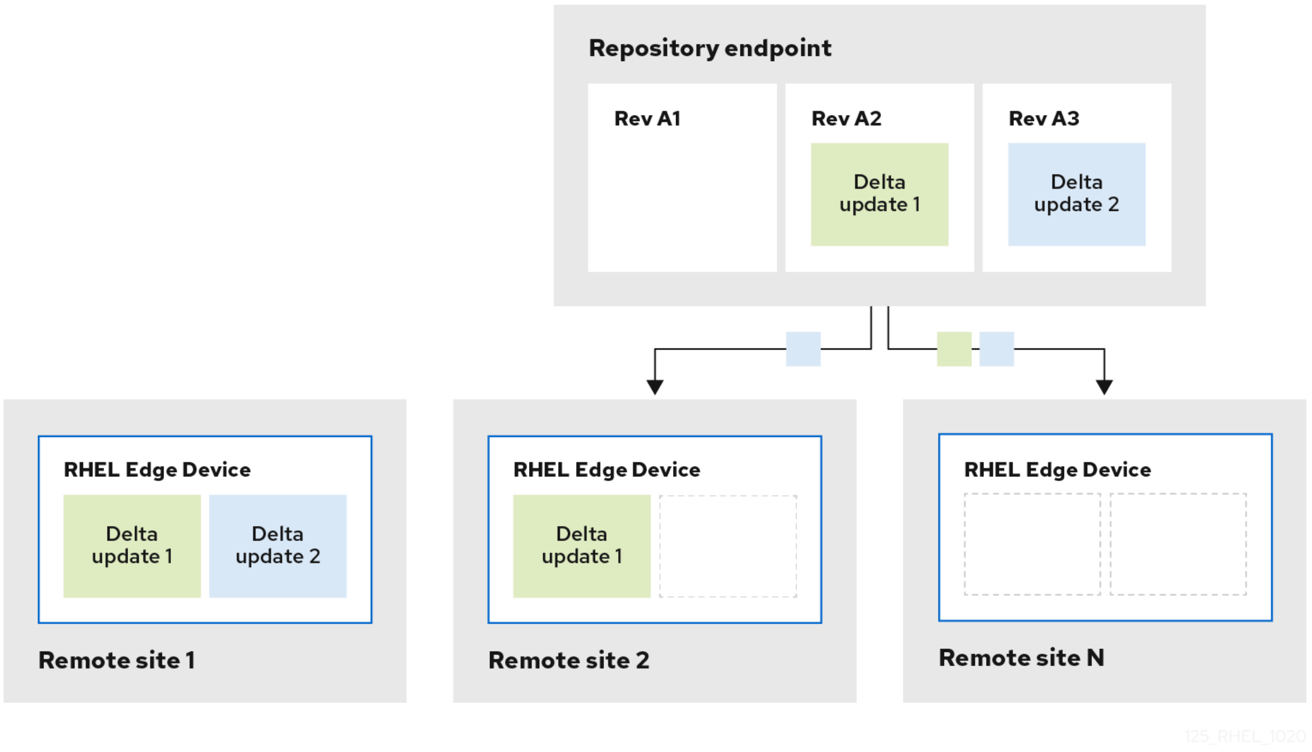
10.2.1. How are RHEL for Edge image updates deployed

With RHEL for Edge images, you can either deploy the updates manually or can automate the deployment process. The updates are applied in an atomic manner, where the state of each update is known, and the updates are staged and applied only upon reboot. Because no changes are seen until you reboot the device, you can schedule a reboot to ensure the highest possible uptime.

During the image update, only the updated operational system content is transferred over the network. This makes the deployment process more efficient compared to transferring the entire image. The operational system binaries and libraries in /**usr** are **read-only**, and the **read and write** state is maintained in /**var** and /**etc** directories.

When moving to a new deployment, the /**etc** and the /**var** directories are copied to the new deployment with **read and write** permissions. The /**usr** directory is copied as a soft link to the new deployment directory, with **read-only** permissions.

The following diagram illustrates the RHEL for Edge image update deployment process:



By default, the new system is booted using a procedure similar to a **chroot** operation. The new /**sysroot** directory mainly has the following parts:

- Repository database at the /**sysroot/ostree/repo** directory.
- File system revisions at the /**sysroot/ostree/deploy/rhel/deploy** directory, which are created by each operation in the system update.
- The /**sysroot/ostree/boot** directory, which links to deployments on the previous point. Note that /**ostree** is a soft link to /**sysroot/ostree**. The files from the /**sysroot/ostree/boot** directory are not duplicated. The same file is used if it is not changed during the deployment. The files are hard-links to another file stored in the /**sysroot/ostree/repo/objects** directory.

The operational system selects the deployment in the following way:

1. The **dracut** tool parses the **ostree** kernel argument in the **initramfs root** file system and sets up the /**usr** directory as a **read-only** bind mount.
2. Bind the deployment directory in /**sysroot** to / directory.
3. Re-mount the operation system already mounted **dirs** using the **MS_MOVE** mount flag

If anything goes wrong, you can perform a deployment rollback by removing the old deployments with the **rpm-ostree** cleanup command. Each client machine contains an **OSTree** repository stored in /**ostree/repo**, and a set of deployments stored in /**ostree/deploy/$STATEROOT/$CHECKSUM**.

With the deployment updates in RHEL for Edge image, you can benefit from a better system consistency across multiple devices, easier reproducibility, and better isolation between the pre and post system states changes .

10.2.2. Deploying RHEL for Edge image updates manually

After you have edited a RHEL for Edge blueprint, you can update the image commit. Image Builder generates a new commit for the updated RHEL for Edge image. Use this new commit to deploy the image with latest package versions or with additional packages.

To deploy RHEL for Edge images updates, ensure that you meet the prerequisites and then follow the procedure.

Prerequisites

- On a RHEL system, you have accessed the Image Builder dashboard.
- You have created a RHEL for Edge image blueprint.
- You have edited the RHEL for Edge image blueprint. See Editing a RHEL for Edge image blueprint using Image Builder in RHEL web console.

Procedure

1. On the Image Builder dashboard, for the blueprint that you have edited, click **Create Image**.
2. On the **Create Image** window, perform the following steps:
   a. From the **Image output type** dropdown list, select **RHEL for Edge Commit (.tar)**. Click **Next**.
   b. In the **OSTree settings** page, enter:
      i. In the **Repository URL**, enter the URL to the OSTree repository of the commit to embed in the image. For example, http://10.0.2.2:8080/repository/. See Setting up a web server to install RHEL for Edge image.
      ii. In the **Parent commit** textbox, specify the parent commit ID that was previously generated. See Extracting RHEL for Edge image commit .
      iii. In the **Ref** textbox, you can either specify a name for your commit or leave it empty. By default, the web console specifies the **Ref** as **rhel/8/arch_name/edge**. Click **Next**.
   c. In the **Customizations** page:
      i. Optional: In the **System**, enter a Hostname. If you do not add it, the operating system determines the hostname. Click **Next**.
      ii. Optional: Click **Add User**. Enter a Username, password and an SSH key. You can mark the user as the Server administrator.
   d. Optional: In the packages page:
      i. In the Available packages, enter the package name to customize your image. Click **Next**.
   e. In the Review page, check the customizations. Click **Save blueprint**. It activates the **Create image** button.

f. Click **Create image**. Image Builder creates a RHEL for Edge image for the updated blueprint.
   To view the RHEL for Edge image creation progress, click the blueprint name from the breadcrumbs, and then click the **Images** tab.

   NOTE

   The image creation process takes a few minutes to complete.

   The resulting image includes the latest packages that you have added, if any, and have the original **commit ID** as a parent.

3. Download the resulting RHEL for Edge image. For more information about downloading a RHEL for Edge image, see Downloading a RHEL for Edge image .

4. Extract the OSTree commit. For more information about extracting an OSTree commit, see Extracting RHEL for Edge image commit .

5. Build a docker container, serving the child commit ID this time.

```
# podman build -t name-of-server --build-arg commit=uuid-child_commit.tar .
```

6. Run the container.

```
# podman run --rm -p 8000:80 name-of-server
```

7. On the RHEL system provisioned, from the original edge image, verify the current status.

```
$ rpm-ostree status
```

   If there is no new commit ID, run the following command to verify if there is any upgrade available:

```
$ rpm-ostree upgrade --check
```

   The command output provides the current active OSTree commit ID.

8. Update OSTree to make the new OSTree commit ID available.

```
$ rpm-ostree upgrade
```

   OSTree verifies if there is an update on the repository. If yes, it fetches the update and requests you to reboot your system so that you can activate the deployment of this new commit update.

9. Check the current status again:

```
$ rpm-ostree status
```

   You can now see that there are 2 commits available:

   - The active parent commit.
   - A new commit that is not active and contains 1 added difference.

10. To activate the new deployment and to make the new commit active, reboot your system.

    ```
    # systemctl reboot
    ```

    The Anaconda Installer reboots into the new deployment. On the login screen, you can see a new deployment available for you to boot.

11. If you want to boot into the newest deployment (commit), the **rpm-ostree** upgrade command automatically orders the boot entries so that the new deployment is first in the list. Optionally, you can use the arrow key on your keyboard to select the GRUB menu entry and press **Enter**.

12. Provide your login user account credentials.

13. Verify the OSTree status:

    ```
    $ rpm-ostree status
    ```

    The command output provides the active commit ID.

14. To view the changed packages, if any, run a diff between the parent commit and the new commit:

    ```
    $ rpm-ostree db diff parent_commit new_commit
    ```

    The update shows that the package you have installed is available and ready for use.

10.2.3. Deploying RHEL for Edge image updates manually using the command-line

After you have edited a RHEL for Edge blueprint, you can update the image commit. Image Builder generates a new commit for the updated RHEL for Edge image. Use the new commit to deploy the image with latest package versions or with additional packages using the CLI.

To deploy RHEL for Edge image updates using the CLI, ensure that you meet the prerequisites, and then follow the procedure.

Prerequisites

- You created the RHEL for Edge image blueprint.
- You edited the RHEL for Edge image blueprint. See Editing a RHEL for Edge image blueprint using command-line interface.

Procedure

1. Create the RHEL for Edge Commit (**.tar**) image with the following arguments:

    ```
    # composer-cli compose start-ostree --ref ostree_ref --url URL-OSTree-repository -
    blueprint_name_ image-type
    ```

    where

    - ***ref*** is the reference you provided during the creation of the RHEL for Edge Container commit. For example, **rhel/8/x86_64/edge**.

    - ***URL-OSTree-repository*** is the URL to the OSTree repository of the commit to embed. I

- ***URL-OSTree-repository*** is the URL to the OSTree repository of the commit to embed in the image. For example, http://10.0.2.2:8080/repository/. See Setting up a web server to install RHEL for Edge image.
- ***image-type*** is **edge-commit**.
  Image Builder creates a RHEL for Edge image for the updated blueprint.

2. Check the RHEL for Edge image creation progress:

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

NOTE

The image creation processes can take up to ten to thirty minutes to complete.

The resulting image includes the latest packages that you have added, if any, and has the original **commit ID** as a parent.

3. Download the resulting RHEL for Edge image. For more information, see Downloading a RHEL for Edge image using the Image Builder command-line interface.
4. Extract the OSTree commit. For more information, see Extracting RHEL for Edge image commit.
5. Serve the OSTree commit by using httpd. See Setting up a web server to install RHEL for Edge image.
6. On the RHEL system provisioned from the original edge image, verify the current status:

```
$ rpm-ostree status
```

If there is no new commit ID, run the following command to verify if there is any upgrade available:

```
$ rpm-ostree upgrade --check
```

The command output provides the current active OSTree commit ID.

7. Update OSTree to make the new OSTree commit ID available:

```
$ rpm-ostree upgrade
```

OSTree verifies if there is an update on the repository. If yes, it fetches the update and requests you to reboot your system so that you can activate the deployment of the new commit update.

8. Check the current status again:

```
$ rpm-ostree status
```

You should now see that there are 2 commits available:

- The active parent commit
- A new commit that is not active and contains 1 added difference

9. To activate the new deployment and make the new commit active, reboot your system:

   ```
   # systemctl reboot
   ```

   The Anaconda Installer reboots into the new deployment. On the login screen, you can see a new deployment available for you to boot.

10. If you want to boot into the newest deployment, the **rpm-ostree upgrade** command automatically orders the boot entries so that the new deployment is first in the list. Optionally, you can use the arrow key on your keyboard to select the GRUB menu entry and press **Enter**.

11. Log in using your account credentials.

12. Verify the OSTree status:

    ```
    $ rpm-ostree status
    ```

    The command output provides the active commit ID.

13. To view the changed packages, if any, run a diff between the parent commit and the new commit:

    ```
    $ rpm-ostree db diff parent_commit new_commit
    ```

    The update shows that the package you have installed is available and ready for use.

10.2.4. Deploying RHEL for Edge image updates manually for non-network-base deployments

After you have edited a RHEL for Edge blueprint, you can update the image commit. Image Builder generates a new commit for the updated RHEL for Edge image. Use this new commit to deploy the image with the latest package versions or with additional packages.

To deploy RHEL for Edge images updates, ensure that you meet the prerequisites and then follow the procedure.

Prerequisites

- An RHEL for Edge system is up and running.
- An OSTree repository is being served over HTTP.
- You have created a RHEL for Edge image blueprint.
- You have edited the RHEL for Edge image blueprint. See Editing a RHEL for Edge image blueprint using Image Builder in RHEL web console.

Procedure

1. On the Image Builder dashboard, for the blueprint that you have edited, click **Create Image**.
2. On the **Create Image** window, perform the following steps:
   a. From the **Type** dropdown list, select **RHEL for Edge Container (.tar)**.

b. In the **Parent commit** textbox, specify the previously generated parent commit ID. See Extracting RHEL for Edge image commit .

c. In the **Repository** textbox, specify the URL to the OSTree repository of the commit to embed in the image. For example, http://10.0.2.2:8080/repository/

d. In the **Ref** textbox, specify the same reference as you provided during the creation of the RHEL for Edge Container commit to embed in the image. For example, **rhel/edge/test**.

e. Click **Create**. Image Builder creates a RHEL for Edge image for the updated blueprint. To view the progress of RHEL for Edge image creation, click the blueprint name from the breadcrumbs, and then click the **Images** tab.

NOTE

The image creation process takes a few minutes to complete.

The resulting image includes the latest packages that you have added, if any, and has the original **commit ID** as a parent.

3. Download the resulting RHEL for Edge image. For more information about downloading a RHEL for Edge image, see Downloading a RHEL for Edge image .

4. Load the RHEL for Edge Container image into Podman, serving the child commit ID this time.

```
$ cat ./child-commit_ID-container.tar | sudo podman load
```

5. Run **Podman**.

```
#  sudo podman run -p 8080:8080 localhost/edge-test
```

6. On the RHEL system provisioned, from the original edge image, verify the current status.

```
$ rpm-ostree status
```

If there is no new commit ID, run the following command to verify if there is any upgrade available:

```
$ rpm-ostree upgrade --check
```

If there are updates available, the command output provides information about the available updates in the OSTree repository, such as the current active OSTree commit ID. Else, it prompts a message informing that there are no updates available.

7. Update OSTree to make the new OSTree commit ID available.

```
$ rpm-ostree upgrade
```

OSTree verifies if there is an update on the repository. If yes, it fetches the update and requests you to reboot your system so that you can activate the deployment of this new commit update.

8. Check the current status:

```
$ rpm-ostree status
```

You can now see that there are 2 commits available:

- The active parent commit.
- A new commit that is not active and contains 1 added difference.

9. To activate the new deployment and to make the new commit active, reboot your system.

   ```
   # systemctl reboot
   ```

   The Anaconda Installer reboots into the new deployment. On the login screen, you can see a new deployment available for you to boot.

10. If you want to boot into the newest commit/deployment, the **rpm-ostree upgrade** command automatically orders the boot entries so that the new deployment is first in the list. Optionally, you can use the arrow key on your keyboard to select the GRUB menu entry and press **Enter**.

11. Provide your login user account credentials.

12. Verify the OSTree status:

    ```
    $ rpm-ostree status
    ```

    The command output provides the active commit ID.

13. To view the changed packages, if any, run a diff between the parent commit and the new commit:

    ```
    $ rpm-ostree db diff parent_commit new_commit
    ```

    The update shows that the package you have installed is available and ready for use.

10.3. DEPLOYING RHEL FOR EDGE AUTOMATIC IMAGE UPDATES

After you install a RHEL for Edge image on an Edge device, you can check for image updates available, if any, and can auto-apply them.

The rpm-ostreed-automatic.service (systemd service) and rpm-ostreed-automatic.timer (systemd timer) control the frequency of checks and upgrades. The available updates, if any, appear as staged deployments.

Deploying automatic image updates involves the following high-level steps:

- Update the image update policy
- Enable automatic download and staging of updates

10.3.1. Updating the RHEL for Edge image update policy

To update the image update policy, use the **AutomaticUpdatePolicy** and an **IdleExitTimeout** setting from the **rpm-ostreed.conf** file at **/etc/rpm-ostreed.conf** location on an Edge device.

The **AutomaticUpdatePolicy** settings controls the automatic update policy and has the following update checks options:

- **none**: Disables automatic updates. By default, the **AutomaticUpdatePolicy** setting is set to **none**.
- **check**: Downloads enough metadata to display available updates with **rpm-ostree** status.
- **stage**: Downloads and unpacks the updates that are applied on a reboot.

The **IdleExitTimeout** setting controls the time in seconds of inactivity before the daemon exit and has the following options:

- 0: Disables auto-exit.
- 60: By default, the **IdleExitTimeout** setting is set to **60**.

To enable automatic updates, perform the following steps:

Procedure

1. In the **/etc/rpm-ostreed.conf** file, update the following:
   - Change the value of **AutomaticUpdatePolicy** to **check**.
   - To run the update checks, specify a value in seconds for **IdleExitTimeout**.
2. Reload the **rpm-ostreed** service and enable the **systemd** timer.

   ```
   # systemctl reload rpm-ostreed
   # systemctl enable rpm-ostreed-automatic.timer --now
   ```

3. Verify the **rpm-ostree** status to ensure the automatic update policy is configured and time is active.

   ```
   # rpm-ostree status
   ```

   The command output shows the following:

   ```
   State: idle; auto updates enabled (check; last run <minutes> ago)
   ```

   Additionally, the output also displays information about the available updates.

10.3.2. Enabling RHEL for Edge automatic download and staging of updates

After you update the image update policy to check for image updates, the updates if any are displayed along with the update details. If you decide to apply the updates, enable the policy to automatically download and stage the updates. The available image updates are then downloaded and staged for deployment. The updates are applied and take effect when you reboot the Edge device.

To enable the policy for automatic download and staging of updates, perform the following updates:

Procedure

1. In the **/etc/rpm-ostreed.conf** file, update ‘AutomaticUpdatePolicy’ to **stage**.
2. Reload the rpm-ostreed service.

```
# systemctl enable rpm-ostreed-automatic.timer --now
```

3. Verify the rpm-ostree status

   ```
   # rpm-ostree status
   ```

   The command output shows the following:

   ```
   State: idle
   AutomaticUpdates: stage; rpm-ostreed-automatic.timer: last run <time> ago
   ```

4. To initiate the updates, you can either wait for the timer to initiate the updates, or can manually start the service.

   ```
   # systemctl start rpm-ostreed-automatic.service
   ```

   After the updates are initiated, the rpm-ostree status shows the following:

   ```
   # rpm-ostree status
   State: busy
   AutomaticUpdates: stage; rpm-ostreed-automatic.service: running
   Transaction: automatic (stage)
   ```

   When the update is complete, a new deployment is staged in the list of deployments, and the original booted deployment is left untouched. You can decide if you want to boot the system using the new deployment or can wait for the next update.

   To view the list of deployments, run the **rpm-ostree status** command.

   Following is a sample output.

   ```
   # rpm-ostree status
   State: idle
   AutomaticUpdates: stage; rpm-ostreed-automatic.timer: last run <time> ago
   Deployments:
   ```

   To view the list of deployments with the updated package details, run the **rpm-ostree status -v** command.

10.4. ROLLING BACK RHEL FOR EDGE IMAGES

You can verify if the updated image is successfully deployed or not. If the deployment is unsuccessful, you can roll back to a previous version (commit). To roll back to a previous functional state, you can either perform the steps manually or can use an automated process.

10.4.1. How are RHEL for Edge images rolled back

With RHEL for Edge images, only transactional updates are applied to the operating system. With the transactional updates, you can easily rollback the unsuccessful updates to the last known good state, preventing system failure during updates.

You can use intelligent rollbacks with Greenboot to eliminate the issues of choosing between application stability and application of security updates.

Greenboot leverages **rpm-ostree** and runs custom health checks that run on system startup. In case of an issue, the system rolls back the changes and preserves the last working state. When you deploy an **rpm-ostree** update, it runs scripts to check that critical services can still work after the update. If the system does not work, the update rolls back to the last known working version of the system. This process ensures that your RHEL for Edge device is in an operational state.

The following is the Greenboot directory structure:

Example 10.1. Greenboot directory structure

```
etc
└─ greenboot
   ├─ check
   |   └─ required.d
   |   └─ init.py
   └─ green.d
   └─ red.d
```

/etc/greenboot/check/required.d

Contains the health checks that must not fail.

/etc/greenboot/check/wanted.d

Contains the health checks that might fail.

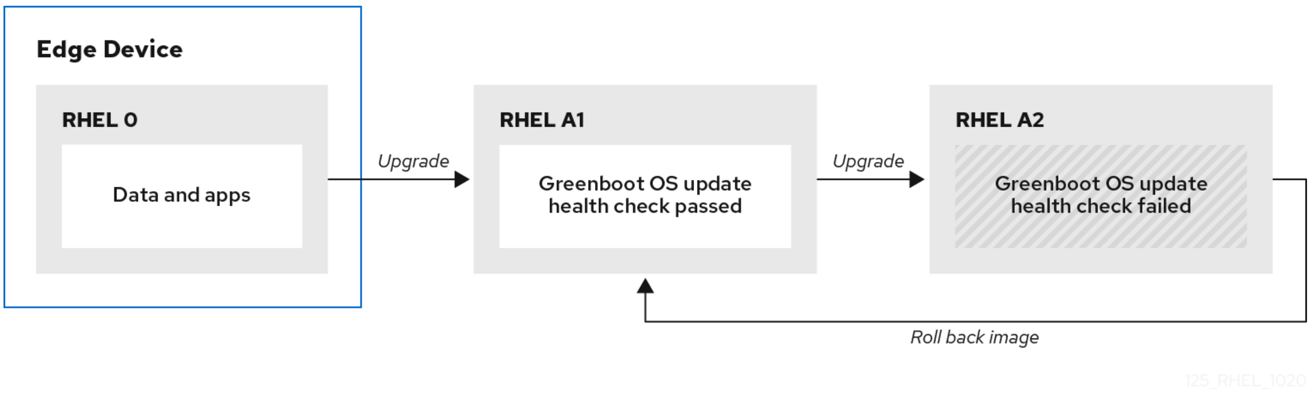
/etc/greenboot/green.d

Contains the scripts to run after a successful boot.

/etc/greenboot/red.d

Contains the scripts to run after a failed boot. The system attempts to boot three times and in case of failure, it executes the scripts.

The following diagram illustrates the RHEL for Edge image roll back process.



10.4.2. Rolling back RHEL for Edge images manually

If the deployment for RHEL for Edge image update fails or if the update fails to work successfully, then you can manually roll back to a previous deployment version.

To roll back to a previous version, perform the following steps:

Procedure

1. Run the **rollback** command:

   ```
   # rpm-ostree rollback
   ```

   The command output provides details about the commit ID that is being moved and indicates a completed transaction with the details of the package being removed.

2. Reboot the system.

   ```
   # systemctl reboot
   ```

   The command activates the previous commit with the stable content. The changes are applied and the previous version is restored.

10.4.3. Rolling back RHEL for Edge images using an automated process

Greenboot checks provides a framework that is integrated into the boot process and can trigger **rpm-ostree** rollbacks when a health check fails. For the health checks, you can create a custom script that indicates whether a health check passed or failed. Based on the result, you can decide when a rollback should be triggered.

To create a health check script, perform the following steps:

Procedure

1. Create a script that returns a standard exit code **0**.
   For example, the following script ensures that the configured DNS server is available:

   ```
   #!/bin/bash

   DNS_SERVER=$(grep ^nameserver /etc/resolv.conf | head -n 1 | cut -f2 -d" ")
   COUNT=0
   # check DNS server is available
   ping -c1 $DNS_SERVER
   while [ $? != '0' ] && [ $COUNT -lt 10 ]; do
   ((COUNT++))
   echo "Checking for DNS: Attempt $COUNT ."
   sleep 10
   ping -c 1 $DNS_SERVER
   done
   ```

2. Include an executable file for the health checks at **/etc/greenboot/check/required.d/**.

   ```
   chmod +x check-dns.sh
   ```

   During the next reboot, the script is executed as part of the boot process, before the system enters the boot-complete.target. If the health checks are successful, no action is taken. If the health checks fail, the system is rebooted several times, before marking the update as failed and rolling back to the previous update.

Verification steps

To check if the default gateway is reachable, run the following health check script:

1. Create a script that returns a standard exit code **0**.

```
#!/bin/bash

DEF_GW=$(ip r | awk '/^default/ {print $3}')
SCRIPT=$(basename $0)

count=10
connected=0
ping_timeout=5
interval=5

while [ $count -gt 0 -a $connected -eq 0 ]; do
  echo "$SCRIPT: Pinging default gateway $DEF_GW"
  ping -c 1 -q -W $ping_timeout $DEF_GW > /dev/null 2>&1 && connected=1 || sleep
$interval
  ((--count))
done

if [ $connected -eq 1 ]; then
  echo "$SCRIPT: Default gateway $DEF_GW is reachable."
  exit 0
else
  echo "$SCRIPT: Failed to ping default gateway $DEF_GW!" 1>&2
  exit 1
fi
```

2. Include an executable file for the health checks at **/etc/greenboot/check/required.d/** directory.

```
chmod +x check-gw.sh
```

APPENDIX A. TERMINOLOGY AND COMMANDS

This section provides the **rpm ostree** terminology and commands.

A.1. OSTREE AND RPM-OSTREE TERMINOLOGY

Following are some helpful terms that are used in context to OSTree and **rpm-ostree** images.

Table A.1. OSTree and rpm-ostree terminology

Term	Definition
OSTree	A tool used for managing Linux-based operating system versions. The OSTree tree view is similar to Git and is based on similar concepts.
rpm-ostree	A hybrid image or system package that hosts operating system updates.
Commit	A release or image version of the operating system. Image Builder generates an ostree commit for RHEL for Edge images. You can use these images to install or update RHEL on Edge servers.
Refs	Represents a branch in ostree. Refs always resolve to the latest commit. For example, **rhel/8/x86_64/edge**.
Revision (Rev)	SHA-256 for a specific commit.
Remote	The http or https endpoint that hosts the ostree content. This is analogous to the baseurl for a yum repository.
static-delta	Updates to ostree images are always delta updates. In case of RHEL for Edge images, the TCP overhead can be higher than expected due to the updates to number of files. To avoid TCP overhead, you can generate static-delta between specific commits, and send the update in a single connection. This optimization helps large deployments with constrained connectivity.

A.2. OSTREE COMMANDS

This section provides a few ostree commands that you can use when installing or managing ostree images.

Table A.2. ostree commands

ostree pull	**ostree pull-local --repo [path] src** **ostree pull-local <path> <rev> --repo=<repo-path>** **ostree pull <URL> <rev> --repo=<repo-path>**
ostree summary	**ostree summary -u --repo=<repo-path>**
View refs	**ostree refs --repo ~/Code/src/osbuild-iot/build/repo/ --list**
View commits in repo	**ostree log --repo=/home/gicmo/Code/src/osbuild-iot/build/repo/ <REV>**
Inspect a commit	**ostree show --repo build/repo <REV>**
List remotes of a repo	**ostree remote list --repo <repo-path>**
Resolve a REV	**ostree rev-parse --repo ~/Code/src/osbuild-iot/build/repo fedora/x86_64/osbuild-demo** **ostree rev-parse --repo ~/Code/src/osbuild-iot/build/repo b3a008eceeddd0cfd**
Create static-delta	**ostree static-delta generate --repo=[path] --from=REV --to=REV**
Sign an **existing** ostree commit with a GPG key	**ostree gpg-sign --repo=<repo-path> --gpg-homedir <gpg_home> COMMIT KEY-ID…**

A.3. RPM-OSTREE COMMANDS

This section provides a few **rpm-ostree** commands that you can use when installing or managing ostree images.

Table A.3. rpm-ostree commands

Commands	Description
rpm-ostree --repo=/home/gicmo/Code/src/osbuild-iot/build/repo/ db list <REV>	This command lists the packages existing in the <REV> commit into the repository.

Commands	Description
rpm-ostree rollback	OSTree manages an ordered list of bootloader entries, called **deployments**. The entry at index 0 is the default bootloader entry. Each entry has a separate **/etc** directory, but all the entries share a single **/var** directory. You can use the bootloader to choose between entries by pressing Tab to interrupt startup. This rolls back to the previous state, that is, the default deployment changes places with the non-default one.
rpm-ostree status	This command gives information about the current deployment in use. Lists the names and refspecs of all possible deployments in order, such that the first deployment in the list is the default upon boot. The deployment marked with * is the current booted deployment, and marking with 'r' indicates the most recent upgrade.
rpm-ostree db list	Use this command to see which packages are within the commit or commits. You must specify at least one commit, but more than one or a range of commits also work.
rpm-ostree db diff	Use this command to show how the packages are different between the trees in two revs (revisions). If no revs are provided, the booted commit is compared to the pending commit. If only a single rev is provided, the booted commit is compared to that rev.
rpm-ostree upgrade	This command downloads the latest version of the current tree, and deploys it, setting up the current tree as the default for the next boot. This has no effect on your running filesystem tree. You must reboot for any changes to take effect.

Additional resources

- The **rpm-ostree** man page.

A.4. FDO AUTOMATIC ONBOARDING TERMINOLOGY

This section provides information about the FDO terminology.

Table A.4. fdo terminology

Commands	Description
FDO	FIDO Device Onboarding.
Device	Any hardware, device, or computer.
Owner	The final owner of the device - a company or an IT department.

Commands	Description
Manufacturer	The device manufacturer.
Manufacturer server	Creates the device credentials for the device.
Manufacturer client	Informs the location of the manufacturing server.
Ownership Voucher (OV)	Record of ownership of an individual device. Contains the following information: * Owner (**fdo-owner-onboarding-service**) * Rendezvous Server - FIDO server (**fdo-rendezvous-server**) * Device (at least one combination) (**fdo-manufacturing-service**)
Device Credential (DC)	Key credential and rendezvous stored in the device at manufacture.
Keys	Keys to configure the manufacturing server * key_path * cert_path * key_type * mfg_string_type: device serial number * allowed_key_storage_types: Filesystem and Trusted Platform Module (TPM) that protects the data used to authenticate the device you are using.
Rendezvous server	Link to a server used by the device and later on, used on the process to find out who is the owner of the device

Additional resources

- FIDO IoT spec

A.5. FDO AUTOMATIC ONBOARDING TECHNOLOGIES

Following are the technologies used in context to FDO automatic onboarding.

Table A.5. OSTree and rpm-ostree terminology

Technology	Definition
UEFI	Unified Extensible Firmware Interface.
RHEL	Red Hat® Enterprise Linux® operating system (OS)
rpm-ostree	Background image-based upgrades.
Greenboot	Healthcheck framework for systemd on rpm-ostree.
Osbuild	Pipeline-based build system for operating system artifacts.
Container	A Linux® container is a set of 1 or more processes that are isolated from the rest of the system.
Coreos-installer	Assists installation of RHEL images, boots systems with UEFI.
FIDO FDO	Specification protocol to provision configuration and onboarding devices.