Red Hat OpenStack Services on OpenShift 18.0

Integrating partner content

Integrating and certifying third-party software and hardware for Red Hat OpenStack Services on OpenShift

Last Updated: 2024-08-19

Red Hat OpenStack Services on OpenShift 18.0 Integrating partner content

Integrating and certifying third-party software and hardware for Red Hat OpenStack Services on OpenShift

OpenStack Team
rhos-docs@redhat.com

Legal Notice

Abstract

This guide contains procedures for integrating and certifying third-party software and hardware in a Red Hat OpenStack Services on OpenShift environment.

Table of Contents

PROVIDING FEEDBACK ON RED HAT DOCUMENTATION

We appreciate your input on our documentation. Tell us how we can make it better.

Providing documentation feedback in Jira

Use the Create Issue form to provide feedback on the documentation for Red Hat OpenStack Services on OpenShift (RHOSO) or earlier releases of Red Hat OpenStack Platform (RHOSP). When you create an issue for RHOSO or RHOSP documents, the issue is recorded in the RHOSO Jira project, where you can track the progress of your feedback.

To complete the Create Issue form, ensure that you are logged in to Jira. If you do not have a Red Hat Jira account, you can create an account at https://issues.redhat.com.

1. Click the following link to open a **Create Issue** page: Create Issue
2. Complete the **Summary** and **Description** fields. In the **Description** field, include the documentation URL, chapter or section number, and a detailed description of the issue. Do not modify any other fields in the form.
3. Click **Create**.

CHAPTER 1. INTEGRATING AND CERTIFYING WITH RHOSO

This guide describes the process for Partners to integrate Partner solutions, such as drivers (Storage, Networking) or application workloads (VNF, NFV), with Red Hat OpenStack Services on OpenShift (RHOSO), before certifying the solutions. Upon the successful integration of the partner solution with RHOSO, partners can then proceed with Red Hat OpenStack Certification. Red Hat OpenStack Certification is offered to Partners who want to offer their solutions for use with RHOSO in a jointly-supported customer environment. For more information about the certification process, see the Product Documentation for Red Hat Software Certification

1.1. GENERAL CERTIFICATION PREREQUISITES

If this is your first time certifying your solution with RHOSO, or you need a refresh on the Red Hat partner process, you can request an orientation session with the Red Hat Technology Partner Success Desk (TPSD) of the Ecosystem Partner Management (EPM) Team. Select the "Certification" for the category and Red Hat OpenStack Platform for the product. The orientation session is to review the requirements for the certification process and answer any questions that you might have.

The following are certification prerequisites and integration points for OpenStack drivers for the Block storage service (cinder), the File storage services (manila), and the Networking service (neutron). Ensure that you review the prerequisites for any OpenStack driver or service before you progress further.

CHAPTER 2. INTEGRATING RHOSO STORAGE SERVICES

Red Hat only certifies Red Hat OpenStack Services on OpenShift (RHOSO) Storage drivers that are distributed by Red Hat. Conversely, Red Hat does not certify drivers that are distributed directly by the Partner. For Red Hat to distribute a Partner's storage driver, ensure that the following requirements are met:

Prerequisites

- The driver must be present in the upstream OpenStack project, such as openstack-cinder or openstack-manila. For information on contributing upstream to OpenStack, see the OpenStack Contributor Guide and the upstream guides for OpenStack Block Storage (Cinder) or OpenStack Shared Filesystems (Manila). The Cinder project also has specific guidelines for contributing drivers that you can view here: All About Cinder Drivers.
- Partners must contribute the driver patches upstream, and the patches must be accepted by the upstream community before they can be included in RHOSO. Red Hat does not accept patches or code modifications to a driver that have not been accepted by the upstream community.
- The driver must be present in the RHOSO release.
    - This requirement is automatically met when the driver is present in the upstream OpenStack release associated with the corresponding version of RHOSO. RHOSO 18 is based on the upstream OpenStack 2023.1 Antelope release. Storage drivers present in the upstream 2023.1 release are automatically included in RHOSO 18.
    - When an upstream driver or updates to that driver is absent in the corresponding upstream release, the Partner can submit requests for Red Hat to back port upstream patches into RHOSO by creating a JIRA issue in the following project: OSPRH.

Additional integration tasks

To integrate a storage driver with RHOSO, you must perform the following actions in addition to the prerequisites:

- Configuring the storage driver.
- Adding software dependencies required by the driver.
- Accessing extra files required by the driver.

2.1. CONFIGURING THE STORAGE DRIVER

Red Hat OpenStack Services on OpenShift (RHOSO) uses OpenShift custom resource definitions (CRDs), which you deploy using an **OpenStackControlPlane** custom resource (CR). The **OpenStackControlPlane** CR includes specification templates that govern the **openstack-cinder** and **openstack-manila** service deployments, which include sections for configuring back end storage drivers. The syntax for configuring storage backends is similar to the **openstack-cinder** and **openstack-manila** syntax.

For more information on how to configure and deploy the **openstack-cinder** and **openstack-manila** storage services, see the Configuring persistent storage guide.

2.2. ADDING SOFTWARE DEPENDENCIES FOR THE STORAGE DRIVER

Red Hat OpenStack Services on OpenShift (RHOSO) **openstack-cinder** and **openstack-manila** services execute in Linux containers that are built by Red Hat. These container images include the necessary software to support a large number of drivers. However, some drivers require additional software components that are not included in the RHOSO container images. These are typically python modules that are supplied by the Partner and not available for inclusion in Red Hat's container images. Drivers that are fully self-contained in RHOSO container images are considered "in-tree," as opposed to drivers that have external software dependencies.

Early in the integration process, Partners must determine whether their driver has an external software dependency. The following information only applies when the driver has external dependencies:

When a Partner's driver has an external software dependency, Partners must provide a container image that adds an additional layer on top of Red Hat's RHOSO container image.

- Partner container images for RHOSO are similar to a partner's container images for director-based RHOSP.
- The purpose of providing a partner container image is to satisfy the external software dependencies required by the Partner's driver. You cannot use container images to deploy a modified version of the Partner's driver or a version of the driver that differs from the one provided by Red Hat in the underlying RHOSO container image.
- Partners are responsible for generating their container images, and the image has to go through container image certification procedure before the Red Hat OpenStack certification. Details on how to certify a container image are provided later in this guide. NOTE: Container image certification is not the same as Red Hat OpenStack certification. Container images must undergo a separate certification procedure in order to be delivered in the Red Hat Ecosystem Catalog.
- After a Partner's storage driver has passed Red Hat OpenStack Certification, the Partner is responsible for generating a certified container image for every subsequent minor update to the RHOSO release.
    - For each minor RHOSO 18 update, Partners must generate an updated container image for the updated release, and publish the updated container image in the Red Hat Ecosystem Catalog.
    - Container images for older RHOSO 18 minor updates must remain in the Red Hat Ecosystem Catalog. This ensures that customers that are not using the latest RHOSO release can still access the Partner's container image that was built for their RHOSO version.

2.2.1. Building partner container images

A Partner must provide a Red Hat certified container image if their storage driver has external software dependencies that are not supplied by Red Hat's corresponding container image. An example is the **cinder-volume** service, which includes Partner drivers for many block storage back ends. When a Partner's driver has external software dependencies, they must provide a **cinder-volume** container image to layer that software on top of Red Hat's RHOSO **cinder-volume** container image.

1. Create a **Containerfile** for generating the container image:
   The following example shows a sample **Containerfile** or **Dockerfile** for generating a **cinder-volume** container image that includes external software dependencies required by a Partner's **openstack-cinder** driver. The example can be adapted to generate a **manila-share** container image that includes external software dependencies required by a Partner's **openstack-manila** driver.

```
FROM registry.redhat.io/rhoso/openstack-cinder-volume-rhel9:18.0.0 1

LABEL name="rhoso18/openstack-cinder-volume-partnerX-plugin" \
      maintainer="maintainer@partnerX.com" \
      vendor="PartnerX" \
      summary="RHOSO 18.0 cinder-volume PartnerX PluginY" \
      description="RHOSO 18.0 cinder-volume PartnerX PluginY" 2

# Switch to root to install software dependencies
USER root

# Enable a repo to install a package 3
COPY vendorX.repo /etc/yum.repos.d/vendorX.repo
RUN dnf clean all && dnf install -y vendorX-plugin

# Install a package over the network 4
RUN dnf install -y http://vendorX.com/partnerX-plugin.rpm

# Install a local package 5
COPY partnerX-plugin.rpm /tmp
RUN dnf install -y /tmp/partnerX-plugin.rpm && \
  rm -f /tmp/partnerX-plugin.rpm

# Install a python package from PyPI 6
RUN curl -OL https://bootstrap.pypa.io/get-pip.py && \
  python3 get-pip.py --no-setuptools --no-wheel && \
  pip3 install partnerX-plugin && \
  rm -f get-pip.py

# Add required license as text file(s) in /licenses directory
# (GPL, MIT, APACHE, Partner End User Agreement, etc)
RUN mkdir /licenses
COPY licensing.txt /licenses

# Switch to cinder user
USER cinder
```

1. Use the FROM clause to specify the Red Hat's RHOSO base image, which in this example is the cinder-volume service. The 18.0 tag specifies the current latest release. To generate an image based on a specific minor release, modify the tag to specify that release, for example 18.0.1, or openstack-cinder-volume-rhel9:*18.0.1*. For RHOSO 18 GA, use the URL: **registry.redhat.io/rhoso/openstack-cinder-volume-rhel9:18.0**.
2. The labels in the sample **Containerfile** override the corresponding labels in the base image to uniquely identify the Partner's image.
3. You can install the software dependencies by this method, or the method at 4, 5, or 6.
4. You can install the software dependencies by this method, or the method at 3, 5, or 6.
5. You can install the software dependencies by this method, or the method at 3, 4, or 6.
6. You can install the software dependencies by this method, or the method at 3, 4, or 5.

2. Build, tag, and upload the container image:
   You can use the **podman build** or **buildah build** commands to build the container image. For more information on how Partners chose a registry and provide an access token to the registry for the certification, see the Red Hat Software Certification Workflow Guide .

   Tag the image to match the corresponding RHOSO 18 base image. For example, when the base image is version 18.0.0, the Partner's image is also tagged as version 18.0.0.

   You can also use the above example procedure with the file storage service, **openstack-manila**. Ensure that you use the appropriate RHOSO **openstack-manila-share** base image in place of the **openstack-cinder-volume** base image.

3. Certify and publish the container image:
   For information on how to certify the container image, see Red Hat Enterprise Linux Software Certification Policy Guide and Red Hat Software Certification Workflow Guide . You can publish container images in the Red Hat Ecosystem Catalog .

2.2.2. Maintaining partner container images and image tags

When a Partner certifies their storage solution, and if the solution includes a container image, then the Partner is responsible for rebuilding that image every time the underlying RHOSO container image changes. This happens with each RHOSO maintenance release, but it can also happen when RHOSO container images are updated to address a CVE.

For example, if a Partner certified their solution against RHOSO 18.0.1, the Partner's container image needs two tags:

- 18.0.1 to indicate the specific release.
- 18.0 to indicate this is the latest version associated with RHOSO 18.

Later, when Red Hat releases version 18.0.2, the Partner must rebuild their image and update the images and tags:

- The tag for the new image is 18.0.2.
- The older 18.0.1 image must remain in the Red Hat Ecosystem Catalog . Partners must not remove old images.
- Remove the 18.0 tag from the older 18.0.1 image, and add it to the new 18.0.2 image.

2.2.3. Deploying partner container images

With director-based RHOSP, you can only specify a single cinder-volume container image despite the number of back ends. However, with RHOSO, you can customize the container image per back end and configure a multi-storage backend within a single RHOSO deployment. You can use the OpenStackVersion CRD to override the container image for any service. In the following example, the CR configures two **cinder-volume** back ends named **backend-X1** and **backend-X2** to use Partner X's container image, and **backend-Y** to use Partner Y's container image. It also configures a manila-share back end named **backend-Z** to use Partner Z's manila-share container image.

```
apiVersion: core.openstack.org/v1beta1
kind: OpenStackVersion
metadata:
  name: openstack
spec:
```

```
customContainerImages:
 cinderVolumeImages:
   backend-X1: registry.connect.redhat.com/partnerX/openstack-cinder-volume-partnerX-plugin
   backend-X2: registry.connect.redhat.com/partnerX/openstack-cinder-volume-partnerX-plugin (1)
   backend-Y: registry.connect.redhat.com/partnerY/openstack-cinder-volume-partnerY-plugin
 manilaShareImages: (2)
   backend-Z: registry.connect.redhat.com/partnerZ/openstack-manila-share-partnerZ-plugin
```

1. The **cinder-volume** back ends named **backend-X1** and **backend-X2** are both associated with Partner X's **cinder-volume driver**, which requires a plugin. Both back ends must specify Partner X's custom container image.

2. Use the same procedure when you deploy a Partner's custom manila-share container image.

2.3. ACCESSING EXTRA FILES FOR THE STORAGE DRIVER

You can use the OpenStackControlPlane CRD **extraMounts** feature to provide files to the **openstack-cinder** and **openstack-manila** storage services, for example extra files that might be required by a Partner's storage driver. Consider a situation where a Partner's **openstack-cinder** driver requires a **config.xml** file that contains authentication credentials in order to access the Partner's back end storage array. You can store the contents of the XML file in a kubernetes secret, which can be created from a YAML file:

```
apiVersion: v1
kind: Secret
metadata:
  name: cinder-volume-example-config (1)
type: Opaque
stringData:
  config.xml: | (2)
 <example-credentials>example</example-credentials> (3)
```

1. The secret name is arbitrary, but the example includes the storage service and the Partner's name, "Example", for clarity.

2. The name of the file required by the Example storage driver is config.xml.

3. Sample XML data.

An **extraMounts** entry in the cinder section of the OpenStackControlPlane CR mounts the **config.xml** into the **cinder-volume pod** associated with the cinder back end.

```
apiVersion: core.openstack.org/v1beta1
kind: OpenStackControlPlane
metadata:
  name: openstack
spec:
  cinder:
    template:
      cinderApi:
        ...
      cinderScheduler:
        ...
```

```
cinderVolumes:
  example: (1)
    customServiceConfig: |
      [example]
      volume_backend_name=example
      volume_driver=cinder....ExampleDriver
    networkAttachments:
      - storage
    replicas: 1
extraMounts: (2)
- extraVol:
  - mounts:
    - name: example-config
      mountPath: /etc/cinder/config.xml (3)
      subPath: config.xml (4)
      readOnly: true
    propagation:
    - example (5)
    volumes:
    - name: example-config
      secret:
        secretName: cinder-volume-example-config (6)
```

1. This section is for the cinder-volume back end configuration for the Example driver.
2. This section is for the extraMounts configuration for the cinder section of the OpenStackControlPlane. Note that the extraMounts are not nested under the cinderVolumes section.
3. This section is for the mount point where the config.xml file appears in the cinder-volume pod.
4. This section is for the subPath to specify the config.xml filename. This is necessary to mount a single file in the /etc/cinder directory.
5. This section is for the propagation to specify that it applies to only the *Example* cinder-volume back end. The value matches the back end name in <1>.
6. The secretName matches the name of the secret created previously.