Reference Architectures 2021

Red Hat Openshift Container Platform 4.6 and 4.7 on Red Hat Openstack Platform 16.1

Last Updated: 2021-04-16

Reference Architectures 2021 Red Hat Openshift Container Platform 4.6 and 4.7 on Red Hat Openstack Platform 16.1

August Simonelli
asimonel@redhat.com

Legal Notice

Abstract

The purpose of this document is to provide guidelines and considerations for deploying Red Hat OpenShift Container Platform 4.6 and 4.7 on Red Hat OpenStack Platform 16.1.

Table of Contents

CHAPTER 1. PURPOSE OF THIS DOCUMENT

In this document, we offer a validated, private cloud solution for running Red Hat OpenShift Container Platform 4.6 and 4.7 on Red Hat OpenStack Platform 16.1.

Red Hat OpenShift Container Platform, Red Hat OpenStack Platform, and Red Hat Ceph Storage are the key architecture components of our solution.

This document is not an implementation guide, but a supplement to product documentation. You can find complete, supported, and fully tested individual product documentation for all three architecture components in the Additional Resources section.

1.1. RED HAT TESTED SOLUTION

This document presents our recommended practices for an OpenShift on OpenStack solution. We analyse each component and explain the options. We then highlight that selection for implementation.

Unlike previous versions, this document does not demonstrate step-by-step implementation steps for those selections. Instead, all recommendations made are tested together, in a lab, by Red Hat's OpenShift on OpenStack Quality Engineering (QE) team.

This document reflects a truly working and tested solution.

For specific implementation steps and procedures, Red Hat product documentation has the details required for a successful deployment. Combined with the review and testing assurances from this solution document, we are able to provide a truly holistic and practical solution to form the basis of any OpenShift on OpenStack implementation.

IMPORTANT

Testing is limited to lab-scale and example workloads. While all efforts are made to simulate real-world conditions, it is recommended that customers reviewing this solution carry out their own testing using their specific workloads and requirements.

1.2. TARGET USE CASES

Enterprise, Telco IT, Government, and IT Service Provider organizations can use this solution. We show you how to implement a Private Cloud with the following programmable capabilities:

- Infrastructure as a Service (IaaS)
- Containers as a Service (CaaS)
- Platform as a Service (PaaS)

This solution targets key use cases such as software developer cloud, web, mobile, AI/ML, and analytics workloads.

1.3. SOLUTION BENEFITS FOR IT AND BUSINESS

Red Hat's tested solution for implementing OpenShift on OpenStack has the following benefits:

- **Accelerated time to value and innovation.**A fully tested and validated solution saves time that has historically been required to design, deploy, and test a full solution stack composed of several products from Red Hat and our ecosystem partners.

- **Simpler deployments with lower risks.**This solution presents a holistic starting point for a full solution stack. This is a highly prescriptive solution to build upon.

- **Quality engineering and testing.**Red Hat has fully tested this solution and supports these recommendations.

CHAPTER 2. TECHNOLOGY OVERVIEW

We offer a solution for running Red Hat OpenShift Container Platform 4.6 and 4.7 on Red Hat OpenStack Platform 16. Our solution deploys Red Hat OpenShift Container Platform 4.6 and 4.7 to physical servers that run Red Hat OpenStack Platform 16.1. We use Red Hat OpenStack Platform director to perform the initial OpenStack installation and Day 2 operations.

Starting with the 4.x stream, the Red Hat OpenShift Container Platform installer features a streamlined interface and simplified installation process allowing a faster, easier, and more precise installation.

Further details of the new installation process are on the OpenShift Container Platform installation overview page.

2.1. RELATIONSHIP BETWEEN OPENSHIFT AND OPENSTACK

The relationship between OpenStack and OpenShift is complementary.

- OpenStack exposes resources via its application programming interface (API) and OpenShift requests them.
- OpenStack provides OpenShift with compute, storage, and networking infrastructure, plus additional resources, such as self-service load balancers and encryption.
- OpenShift runs its containerized applications on the infrastructure provisioned by OpenStack.

The products are tightly integrated. OpenShift can consume OpenStack resources on demand and without user intervention.

2.1.1. Red Hat Enterprise Linux CoreOS (RHCOS)

Beginning with OpenShift 4, OpenShift nodes now run on Red Hat Enterprise Linux (RHEL) CoreOS (RHCOS). RHEL CoreOS combines the ease of over-the-air updates from Container Linux (formerly known as CoreOS) with the Red Hat Enterprise Linux kernel to deliver a more secure, easily managed container host.

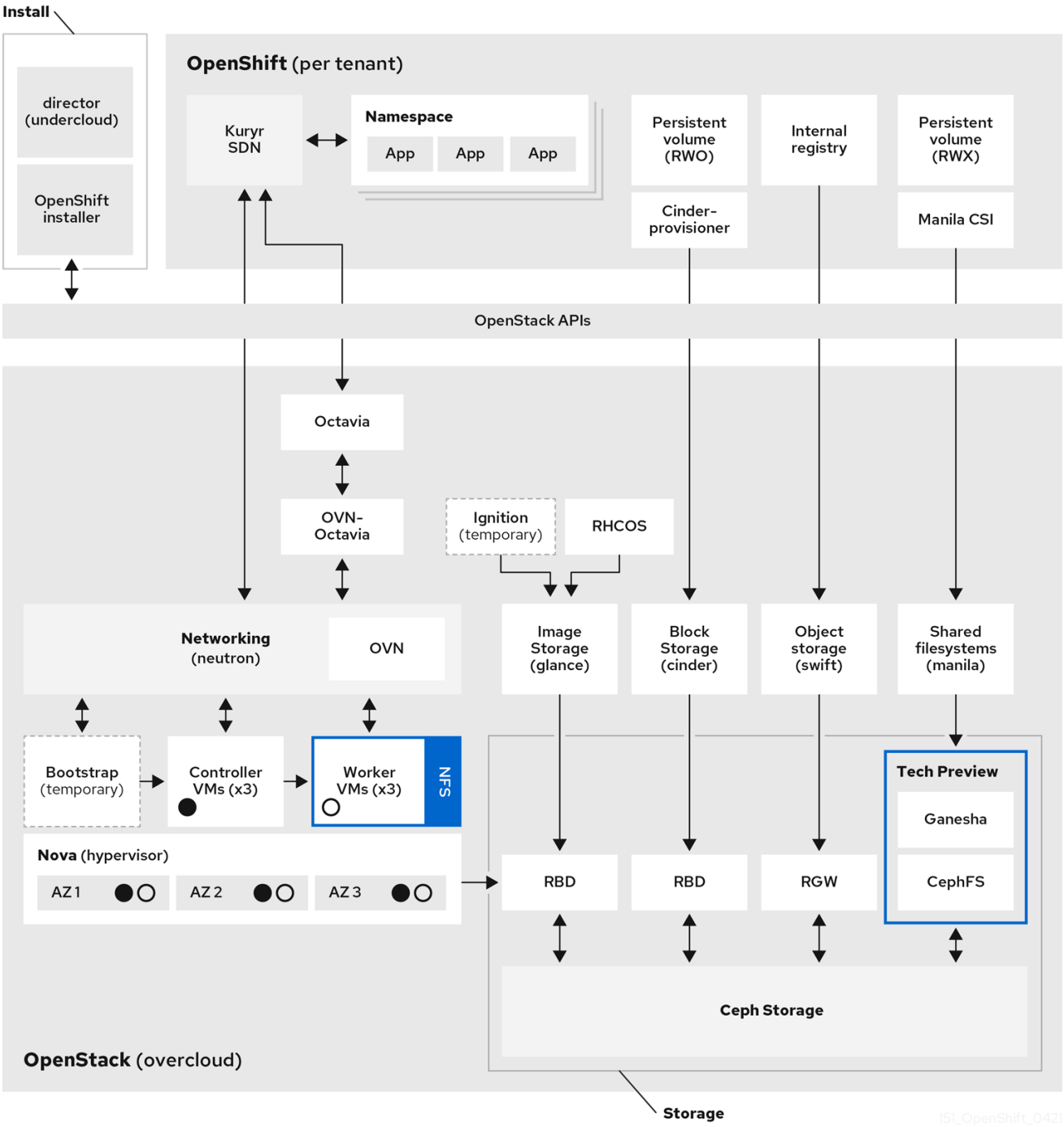
In an installer-provisioned infrastructure based deployment, RHCOS is the supported operating system for all the OpenShift Container Platform nodes and is used by default for workers and controllers. It is also an OpenShift requirement that the controller nodes run RHCOS. Currently, RHCOS is only used in conjunction with OpenShift and not for use as an independent operating system.

Find more information about Red Hat Enterprise Linux (RHEL) CoreOS .

2.2. SOLUTION OVERVIEW

Although there are many available options for placing OpenShift on OpenStack, we provide one validated solution to ensure clarity, simplicity, and supportability. The Red Hat Tested Solution represents the components and integrations of this solution, which has been tested by QE and is a starting point for all enterprise deployments.

Figure 2.1. Diagram of the Red Hat solution



We made these key choices to complete the installation and setup shown in Figure 1:

Installation

- OpenStack is installed using director.
- OpenStack is installed using external TLS encryption.
- OpenShift is installed using the full-stack automation (IPI) method.
- OpenShift is installed from the director host using a non-privileged OpenStack tenant.

Storage

- OpenStack deploys Fileshare-as-a-Service (manila) usable with RWX container workloads.
- OpenStack deploys the Block Storage service (cinder) usable with RWO container workloads.
- OpenStack utilises RHCS for Compute (nova), Image (glance), Block Storage (cinder), and Object (swift).
- OpenStack uses RHCS with Ganesha for Fileshare-as-a-Service (manila).
- OpenShift uses a Container Storage Interface (CSI) driver to provide access to manila.
- OpenShift 4.6 uses the in-tree Cinder provisioner (kubernetes.io/cinder) to provide access to cinder. Beginning with OpenShift 4.7, we also support a Container Storage Interface (CSI) driver for cinder.
- OpenShift uses Object storage for the internal registry.

Compute

- OpenShift control-plane and worker VMs are deployed using nova availability zones to provide high availability.

Networking

- OpenStack uses Open Virtual Network (OVN) for its SDN.
- OpenShift networking is managed by Kuryr-Kubernetes and Kuryr-CNI, which allows Neutron-based networking in Kubernetes
- OpenStack deploys Load-Balancing-as-a-Service (Octavia) for OpenShift load balancing
- OpenShift uses the OVN Provider driver for Octavia to provide load balancing

CHAPTER 3. DESIGN CONSIDERATIONS

This section describes how our solution addresses key design considerations for necessary and key integrations. You must consider each section in your architectural planning for an OpenShift on OpenStack implementation. In each section, we discuss key integrations, their various options, advantages, concerns, and overall requirements. We then conclude each section with an explanation of the design consideration we chose in our solution.

The recommendations in this solution were developed in conjunction with Red Hat Quality Engineering, Field Consultants, Engineers, and Product teams.

3.1. SUPPORT LIFECYCLES

The release of Red Hat OpenShift Container Platform 4.6 is an Extended Update Support (EUS) release for OpenShift. Customers can now combine long-term support options from OpenStack with EUS for OpenShift. This integrated solution provides enterprise-essential support timeframes across the stack.

Please note, Red Hat OpenShift Container Platform 4.7 is not an EUS release and follows the standard release cadence for OpenShift releases.

3.1.1. Red Hat OpenShift Container Platform

The 4.6 release of Red Hat OpenShift Container Platform Red Hat is designated as an Extended Update Support (EUS) release. EUS provides maintenance support coverage for a release, replacing the release delimited maintenance support phase with a fixed 14 months maintenance support phase. Inclusive of the full support phase, this provides customers with an approximate 18 months total support coverage for a minor version.

For full details on EUS for Red Hat OpenShift Container Platform, including life cycle updates and a list of covered components, please visit the Red Hat OpenShift Container Platform Life Cycle Policy page.

3.1.2. Red Hat OpenStack Platform

The 16.1 release of Red Hat OpenStack Platform is offered with four years of production support and the option to purchase an extended lifecycle support (ELS) for the final year. Support is offered in three distinct phases of:

- Full Support (the first 18 months)
- Maintenance Support (the next 18-36 months)
- Extended Life Support (an additional two years for an extra subscription cost).

For information on the Red Hat OpenStack Platform life cycle with full details on the different support types, see Red Hat OpenStack Platform Life Cycle .

3.1.3. Red Hat tested solution: support lifecycle

Our solution uses Red Hat OpenShift Platform 4.6 on Red Hat OpenStack Platform 16.1. Both platforms have long-term support solutions.

3.2. CONNECTIVITY

You can deploy OpenShift to many different on-premises architectures. OpenShift can be run with a connected installation, via a proxied connection, and in a restricted, or air-gapped, network environment.

3.2.1. Connected installation

Both the OpenStack and OpenShift installer can take advantage of direct Internet access and an active subscription to Red Hat products. The installation processes are not self-contained and require access to external sources and DNS resolution to retrieve the following core assets:

- OpenStack's control plane containers
- OpenShift control and worker node containers
- OpenShift's RHCOS image

3.2.2. Proxied installation

An OpenShift deployment supports the implementation of an HTTP or HTTPS cluster-wide proxy. Details for setting up a cluster-wider proxy are in the Configuring the cluster-wide proxy section of the documentation.

3.2.3. Restricted network deployment (air-gapped)

Red Hat OpenShift Platform 4 on Red Hat OpenStack Platform 16.1 fully supports a restricted network deployment. Before performing a restricted network deployment, you must prepare the basic components required to bootstrap the installation. Please review the supported installation methods for different platforms for further details.

3.2.4. Red Hat tested solution: connected

In our solution we use the Connected Installation method for installation. We have used a direct connection to the Internet, including full DNS resolution for director and all OpenShift nodes.

Proxied and restricted (air-gapped) deployments are fully tested by Red Hat QE and fully supported for use. We've chosen not to use them for this solution to ensure the simplest path to deployment and to meet lab requirements.

3.3. INSTALLATION METHOD AND TOOLING

For best results, install Red Hat products and product integrations with the tools recommended and supported by Red Hat.

3.3.1. Red Hat OpenStack Platform director

Red Hat OpenStack Platform director is a management and installation tool based on the OpenStack TripleO project. TripleO stands for "OpenStack On OpenStack." This solution uses director to install and manage the life cycle of Red Hat OpenStack Platform.

The fundamental concept behind Red Hat OpenStack Platform director is that there are two clouds: the *undercloud* and the *overcloud*. The undercloud is a standalone OpenStack deployment whose sole purpose is to manage another cloud. It can be deployed on a single physical server or a virtual machine.

The administrator uses the undercloud's OpenStack services to define and deploy the production OpenStack cloud. Director is also used for Day 2 management operations, such as applying software updates and upgrading between OpenStack versions.

The second cloud – called the *overcloud* – is the full-featured production environment deployed by the undercloud. The overcloud is comprised of physical servers with various roles:

1. *Controller nodes* run the OpenStack API endpoints. They also store OpenStack's stateful configuration database and messaging queues.

2. *Compute nodes* run virtual machine hypervisors. They host the computing resources allocated for user workloads.

3. *Storage nodes* provide block, object, or software defined storage for the user workloads.

OpenShift Container Platform runs within a project, or tenant, on the overcloud. Each tenant's OpenShift cluster is isolated from each other.

Director is required for all production OpenStack Platform deployments. Red Hat engineering's tested and validated configuration settings are embedded into director's deployment tooling.

3.3.2. Red Hat OpenShift Container Platform 4 installation patterns

The OpenShift 4 installer offers flexibility through three basic types of Red Hat OpenShift Container Platform installations:

- **Full stack automation** or Installer-Provisioned Infrastructure (IPI)

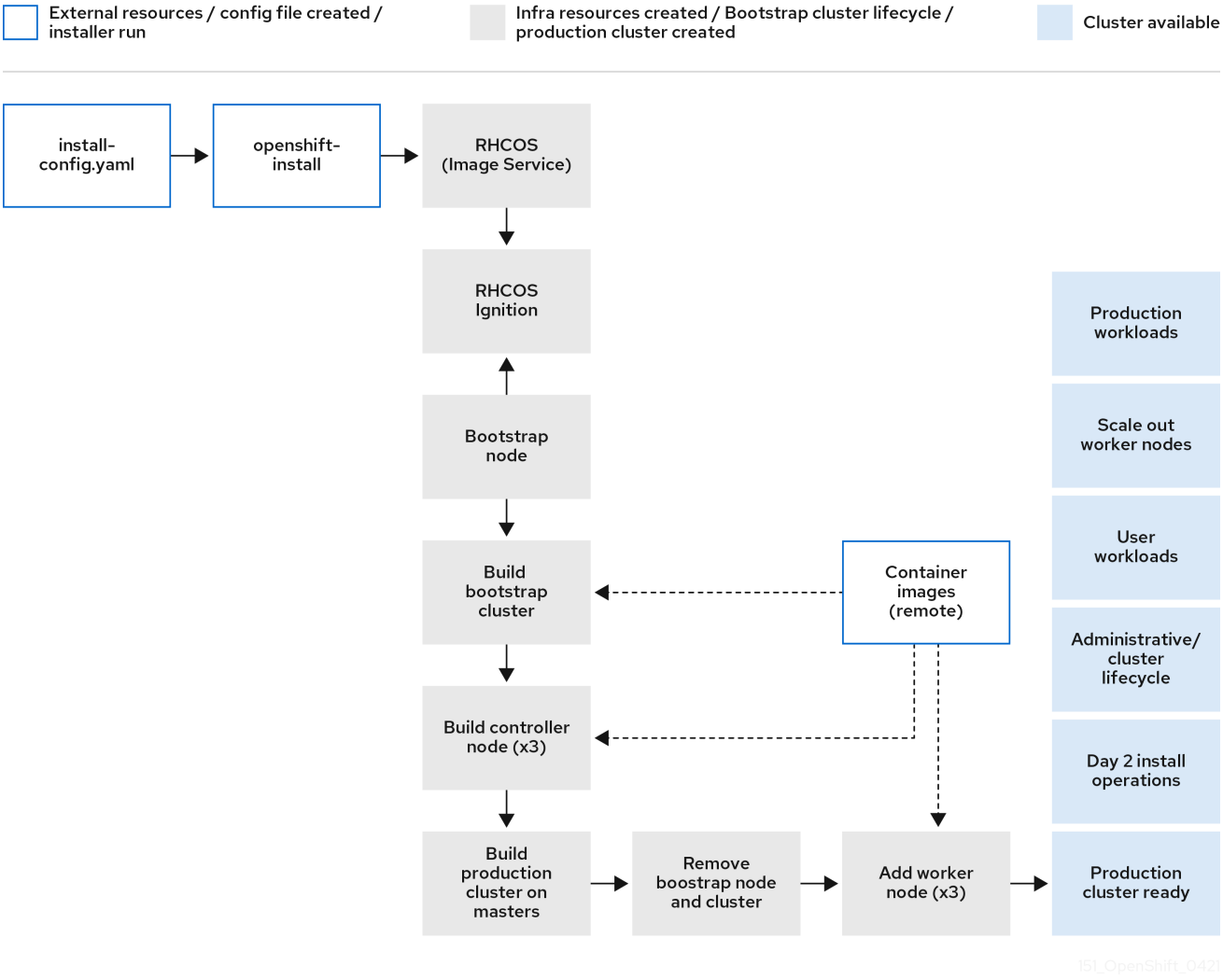
- **Pre-existing Infrastructure** or User-Provided Infrastructure (UPI)

- **Platform Agnostic** which was formerly referred to as Bare Metal installation.

3.3.2.1. Full stack automation

With full stack automation, the installer manages all aspects of the installation, including infrastructure provisioning with an opinionated best practices deployment of OpenShift. The installer is able to request, deploy, and manage the underlying infrastructure directly via an infrastructure-aware provider that has knowledge of that infrastructure. The full stack automation method results in production-ready installations from minimal configuration, which provides a managed service experience to deploying self-managed clusters.

Figure 3.1. The OpenShift Full Stack Automation installation workflow



3.3.2.1.1. Guided installation to generate a configuration file

The IPI install makes things easy via a guided workflow to create an install configuration file. The guided install prompts users for the majority of values needed for their OpenShift install. This results in a simple YAML file that describes the end state of the installation. The YAML file also allows an operator to add additional customisations to their installation within the installer-provisioned infrastructure footprint.

3.3.2.2. Pre-existing infrastructure

With the pre-existing Infrastructure installation method, administrators are responsible for creating and managing their own underlying infrastructure before installation. The installer uses an infrastructure-aware provider, but the administrator must prepare the components in advance. The OpenShift installer can then utilise the infrastructure provider to interact with the prepared infrastructure. For OpenShift on OpenStack, Red Hat provides sample Ansible scripts for this purpose.

Further details about this method can be found in the Selecting a cluster installation method and preparing it for users.

3.3.2.3. Platform agnostic (formerly Bare Metal installation)

A platform agnostic installation can be used on any underlying infrastructure because it does not utilise any infrastructure provider. The installer is not able to control any of the underlying infrastructure

operations, which means that an administrator must both prepare the infrastructure and orchestrate the installation via their own custom methods. This type of installation is the most flexible, but requires additional custom automation.

Full stack automation, pre-existing Infrastructure, and platform agnostic installation methods are fully supported by Red Hat.

For a comprehensive overview of installation patterns and platforms, see Selecting a cluster installation method and preparing it for users in the official documentation.

3.3.3. Red Hat tested solution: installation

This solution uses the full stack automation method for installing Red Hat OpenShift Platform onto Red Hat OpenStack Platform.

- We use the installer to generate an installation configuration file.
- We use the file to guide the installer to programmatically create all required portions of the networking, machines, and operating systems via the OpenStack API while utilizing the infrastructure-aware OpenShift provider.
- This results in an architecture which is highly available, fully tested, and is an entirely supported solution suitable for production.

NOTE

Full stack automation, installer-provisioned infrastructure, and IPI are interchangeable terms for this document.

The installer-provisioned infrastructure method for OpenShift is a highly prescriptive installation pattern which performs a best practice installation. This method of installation allows for minimal manual variance. In general, infrastructure changes for an installer-provisioned infrastructure deployment should not be customised manually after deployment. All changes must be implemented by the installer itself through direct interaction with the underlying infrastructure and API's. Day 2 operations, such as machine scale out, are possible and supported, but the low level result of an installation should be entirely installer driven.

3.4. HIGH AVAILABILITY

High availability is a requirement for any production deployment. Our solution is highly available at the OpenStack, OpenShift, and Ceph layers.

3.4.1. OpenStack HA

Red Hat OpenStack Platform ensures high availability in the OpenStack control plane through the following actions:

- OpenStack Platform director deploys three controller nodes which run multiple instances of the OpenStack services and APIs simultaneously on all three controllers.
- HAproxy load balances connections across the controller API endpoints to ensure service availability.
- A Galera cluster protects the OpenStack state database.

- RabbitMQ queues are duplicated across all nodes to protect the message bus.

Full details for OpenStack are available in the High Availability Deployment and Usage guide.

3.4.2. OpenShift HA

The OpenShift IPI installer ensures OpenShift control plane high availability by deploying three control plane nodes by default. This ensures the etcd state database meets stated minimum HA requirements and is collocated across at least three separate nodes.

3.4.2.1. Node HA

When you deploy OpenShift on OpenStack, you must consider the placement of OpenShift nodes carefully:

- You must not place control plane nodes on the same underlying compute nodes.
- You must space worker nodes evenly across compute infrastructure.
- To ensure control plane HA, OpenStack must provide at least three distinct compute nodes. These compute nodes must never be on the same compute node as control plane nodes.

3.4.2.2. Control Plane HA

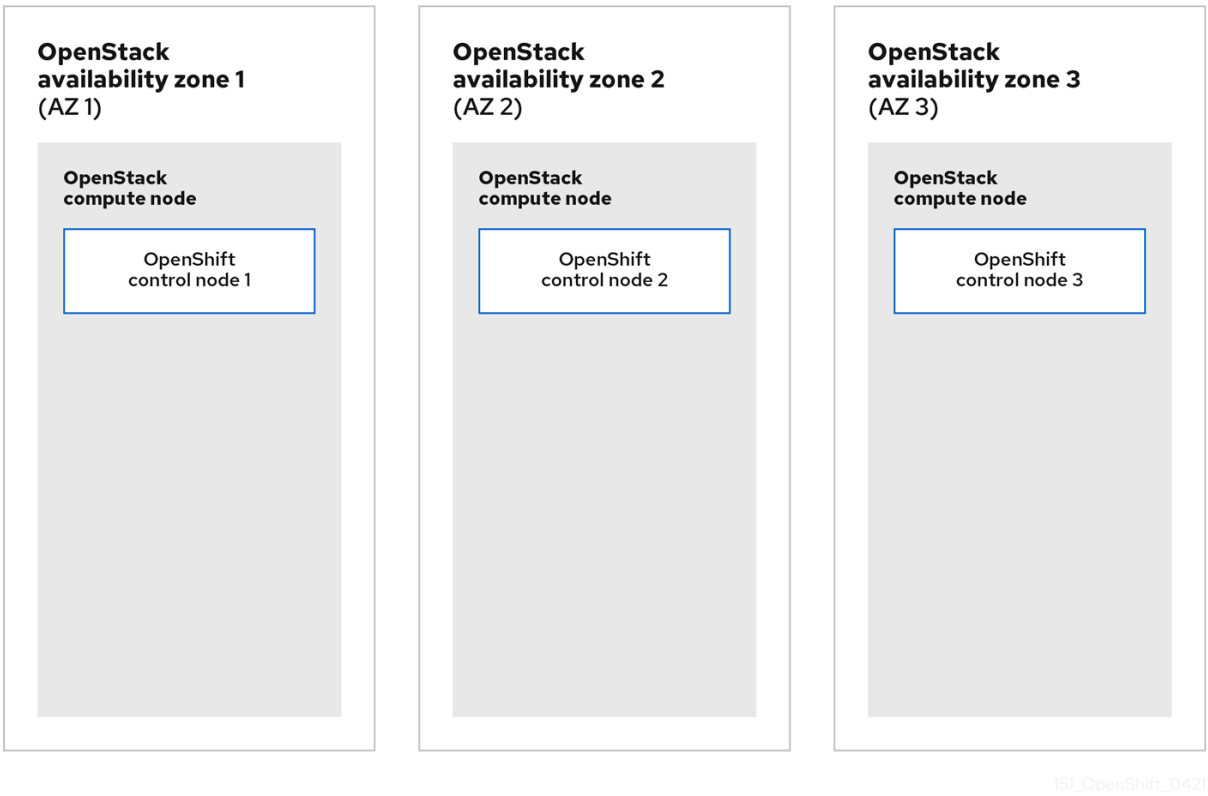
Red Hat offers a few supported ways to manage high availability for the OpenShift on OpenStack control plane VMs:

- Nova availability zones
- OpenStack soft anti-affinity policies

3.4.2.3. Nova availability zones

Using OpenStack availability zones is the preferred way to ensure that control plane nodes are properly managed for HA.

Figure 3.2. Placing OpenShift control plane nodes across computes using nova availability zones.



In OpenStack, the concept of availability zones allow administrators to logically partition their cloud and provide a simple abstraction of the physical infrastructure for cloud tenants to utilise easily. Availability zones are very granular in OpenStack. Some OpenStack services, such as nova, cinder, and neutron, have their own availability zone implementations.

For an HA OpenShift control plane, an openstack cloud administrator must provide a minimum of three availability zones and ensure that compute resources offered are properly configured.

For an IPI install, you must add control plane availability zones to the **install-config.yaml** file manually before installation as the guided install does not prompt for them. Zones are added to the configuration file as a YAML array.

Setting nova availability zones in the OpenShift installation configuration file.

```
controlPlane:
  name: master
  platform:
    openstack:
      zones: ['AZ0', 'AZ1', 'AZ2']
  replicas: 3
```

At install time, the OpenShift installer assigns each control plane node to a different AZ upon VM creation.

3.4.2.4. OpenStack soft anti-affinity policies

OpenStack supports the ability to deterministically place virtual machine instances through affinity policies. Affinity rules are grouped together into detailed policies allowing an administrator to control many aspects of placement. Through the use of these policies, combined with OpenStack server

groups, deployments without nova availability zones still remain HA.

Beginning with version 4.5 of the OpenShift on OpenStack installer, each control plane instance is placed in its own server group. These server groups are then used with an anti-affinity policy to ensure they do not land on the same OpenStack compute hosts.

3.4.2.5. Worker node HA

There are two ways to utilise availability zones for worker nodes:

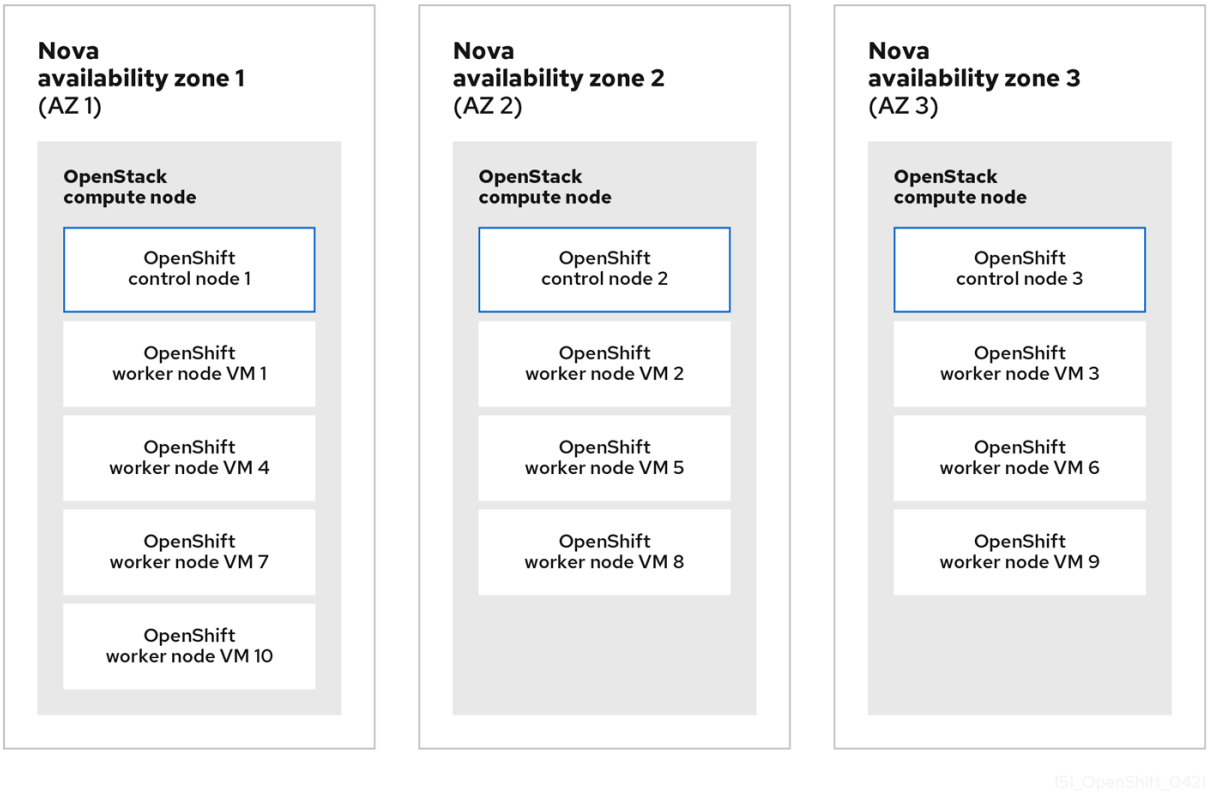
- At install time via the installation file
- During Day 2 operations, via manually created machineSets using the availabilityZone providerSpec values.

For worker nodes, the installer has the following placement rules:

- The installer creates a MachineSet per given availability zone.
- The installer places nodes and distributes the replicas as evenly as possible across zones.

The following diagram shows one way that you can configure your nodes into availability zones:

Figure 3.3. An example of OpenShift control plane and worker nodes deployed across three nova availability zones.



3.4.2.6. Load balancer HA

When using Kuryr-SDN, you can use the Octavia OVN driver (ovn-octavia). With this driver, Octavia does not create VMs for load balancing OpenShift services. Octavia balances via Openflow rules. Using this method reduces the resource consumption required for load balancing by eliminating the need for resource-hungry virtual machines.

A single Octavia load balancer VM is created with Kuryr-SDN. This load balancer VM provides an OpenShift API endpoint for all services.

For details about this topic, see Octavia Feature Support Matrix.

3.4.2.7. Storage layer HA

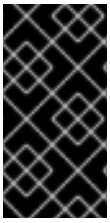
By default, a Red Hat Ceph Storage (RHCS) deployment is highly available. For this solution, we deploy RHCS with director. Deploying RHCS this way automatically integrates and provides HA storage for nova, glance, and cinder services.

3.4.3. Red Hat tested solution: high availability

3.4.3.1. OpenStack

Implementing high availability for our solution for OpenStack is done through multiple components:

- Red Hat OpenStack Platform director deploys three OpenStack controllers, three Ceph Storage OSDs, and three OpenStack computes.
- OpenStack services, such as Compute, Image, and Object storage are deployed on Ceph, and provide storage layer resilience.
- The OpenStack cloud is deployed with three nova availability zones with at least one compute host in each zone.



IMPORTANT

You must have at least three OpenStack computes available to ensure that your OpenShift control plane can be adequately separated. You must also ensure that you have the ability to quickly replace a failed compute or your control plane will not run as HA if an OpenStack compute fails.

Implementing load balancing for OpenShift applications is done with Octavia, the load balancing-as-a-service component for Red Hat OpenStack Platform 16.1.

3.4.3.2. OpenShift

We use nova availability zones to implement high availability for our OpenShift solution. We place each controller node in a separate availability zone to ensure that they are never on the same physical servers.

Worker nodes are distributed as evenly as possible across the same availability zones, which ensures an even distribution across physical infrastructure.

3.5. STORAGE

Select storage back ends that meet the needs of OpenShift applications while minimizing complexity. Both OpenStack Platform and OpenShift Container Platform have independent and mature storage solutions. However, combining solutions for each platform without planning can increase complexity and unwanted performance overhead.

Consider your workloads if you want to avoid duplicate storage components.

3.5.1. Overall storage back end

Red Hat Ceph Storage provides preferred, scalable, integrated, and supported cloud storage for OpenStack. Red Hat Ceph Storage is tightly integrated with the complete Red Hat cloud software stack. It provides block and object storage capabilities.

- Ceph cluster servers are divided into Monitors and Object Storage Device (OSD) nodes.
- Ceph monitors run the monitor daemon, which keeps a main copy of the cluster topology.
- Clients calculate the location of the data via an algorithm, which determines how to store and retrieve data (referred to as a "CRUSH map")
- Clients read or write the data directly to and from the OSDs.
- Monitors maintain cluster health, state, and topology.
- Clients interact directly with the OSDs. Clients only check with the Monitors to ensure the CRUSH map is up-to-date.
- Data is replicated across the physical disks within the nodes.

3.5.1.1. Ceph back ends

Red Hat OpenStack Platform 16.1 implements Red Hat Ceph Storage 4 as part of a director-driven storage. The BlueStore back end is used by default. Installations of Red Hat Ceph Storage 4 and later only support the use of the more performant BlueStore back end.

To learn more about storage back ends used for Ceph, see the architecture documentation covering the Ceph ObjectStore section in the official documentation.

To learn more about configuring Ceph, review the Red Hat Ceph Storage: Supported configurations knowledge base article.

3.5.2. Object Storage (OpenStack swift)

OpenStack supports access to Object Storage via the OpenStack Object Store service (swift).

By default, when installing Red Hat OpenStack Platform with director, a simple object storage deployment is placed on the controllers. This object storage is supported in production for minimal workloads, such as a back end for glance.

3.5.2.1. OpenShift registry

The OpenShift installer follows the recommended practice for a scaled registry and attempts to use OpenStack's object storage service as the preferred location for the internal image registry back end. To do this, the installer checks for an accessible object storage location and uses that if found. If it cannot find one, it uses the recommended best practice for a non-scaled registry and creates a ReadWriteOnce (RWO) cinder volume.

3.5.3. Image Storage (OpenStack glance)

When deploying Ceph with director, the default installation of the OpenStack Image Storage solution (glance), is implemented to be backed by Ceph. The Openshift installer utilises glance for two purposes:

- To store the openshift ignition files used to bring up the bootstrap cluster

- To store the Red Hat Enterprise Linux CoreOS image.

3.5.4. Persistent storage for OpenShift

The Kubernetes persistent volume (PV) framework allows OpenShift users to request storage from the underlying storage subsystem without needing to have any knowledge of the storage architecture.

OpenShift supports many methods for providing access to storage back ends. For OpenShift 4.6 and 4.7 on OpenStack 16.1, we suggest the use of a storage back end that supports the Container Storage Interface (CSI) standard when possible. CSI provides a standardized way to provide access to underlying block and file systems to containerized workloads.

To learn more about CSI, see the "Kubernetes CSI Developer" section in Kubernetes Container Storage Interface.

3.5.4.1. Manila-CSI for RWX PVs

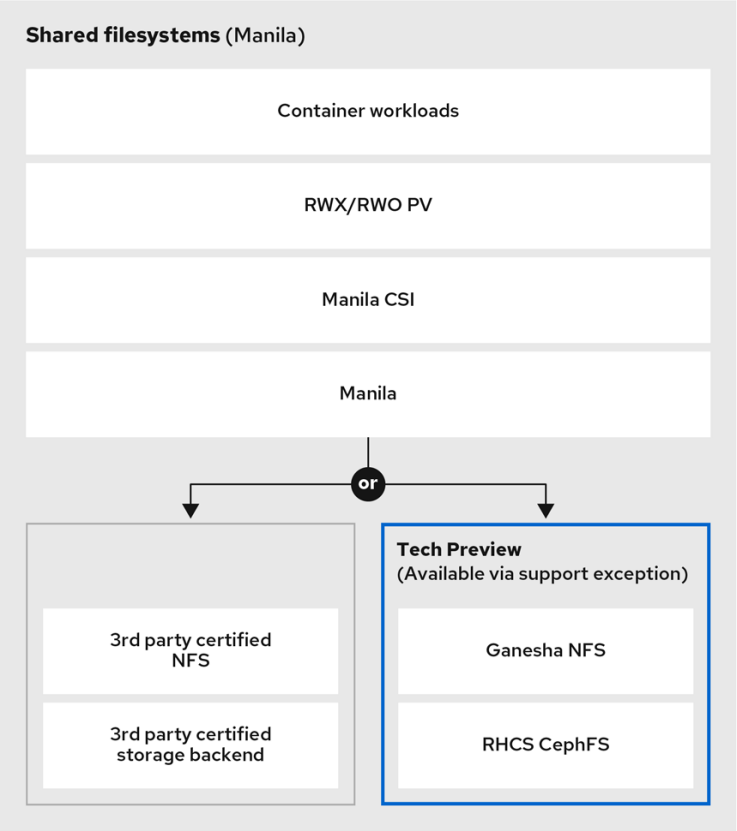
Beginning with version 4.5, OpenShift supports the OpenStack Manila CSI Driver Operator, so that OpenShift can interact with OpenStack's Shared File Systems service (manila) to provision PVs from remote, shareable file systems.

Beginning with version 4.6, OpenShift installs the Manila CSI Driver Operator by default when it detects the Shared File Systems service.

Accessing remote shared file systems by using the Shared File Systems service means that container workloads can use a storage back end that allows simultaneous access for compute instances, bare metal nodes, and containers. This kind of access, known to an OpenShift PV as its access mode, is called ReadWriteMany (RWX), and is required for complex container use cases.

Before the introduction of the Manila-CSI Driver Operator, container workloads on OpenStack did not have a supported RWX solution. With Manila-CSI, an OpenShift administrator can consume a remote shared file system easily via the OpenStack API and the Shared File Systems service.

Figure 3.4. Using Manila for Container Workloads



The OpenShift installer creates a storage class automatically if OpenShift detects Manila in the underlying OpenStack cloud. The installer does not set that storage class as the default.

The storage classes created by the Full Stack Automation installation method

```
$ oc get sc
NAME                 PROVISIONER                RECLAIMPOLICY   VOLUMEBINDINGMODE
ALLOWVOLUMEEXPANSION   AGE
csi-manila-default   manila.csi.openstack.org   Delete          Immediate
false                  37d
standard (default)   kubernetes.io/cinder       Delete          WaitForFirstConsumer   true
                   37d
```

To learn how to set a default StorageClass, see Changing the default storage class .

3.5.4.2. Cinder and Block Storage for RWO PVs

The in-tree Kubernetes Cinder driver has been the default driver in all OpenShift on OpenStack deployments. This driver is not a CSI compliant driver and is only a simple block storage provider. PVs provisioned with this driver are dynamically allocated, but only support the ReadWriteOnce (RWO) Access Mode. RWO can be limiting for some use cases.

Beginning with OpenShift 4.7, we offer a CSI compliant driver: Cinder CSI. This new driver is available and supported for OpenShift on OpenStack.

Although the default is still the in-tree driver, we plan to switch the default to Cinder CSI in OpenShift 4.8.

3.5.5. Red Hat OpenShift Container Storage 4.6.1+

Beginning with Red Hat OpenShift Container Storage 4.6.1 (OCS), the use of OCS with OpenShift on OpenStack deployments received a Technology Preview designation. We do not support Technology Preview features for production. Review the Technology Preview Features Support Scope for full details or contact a Red Hat Associate for more information.

Red Hat OpenShift Container Storage (OCS) is persistent software-defined storage that is integrated with and optimized for Red Hat OpenShift Container Platform. OCS offers storage to an OpenShift installation through a containerised solution that is run within OpenShift directly.

OCS has two modes of providing a storage for container workloads with OpenShift on OpenStack:

- External Mode: OpenShift workloads can directly access an external Red Hat Ceph Storage installation using this mode. OCS, running within OpenShift, provides container workloads access to the existing Ceph cluster. This mode is specifically engineered to work with a Ceph back end and is not supported with other back ends.

- Internal Mode: OCS uses OpenShift PVs provided by Cinder-CSI to create its own Ceph cluster on top of the OSP Cinder back end. This allows OCS Internal Mode to offer all the features of a full Ceph cluster hosted entirely within OpenShift. Because Internal Mode uses Cinder-backed PVs, this mode can also be used with any certified 3rd party storage back end for Cinder.

Red Hat plans to offer full support for OpenShift Container Storage on OpenStack in a future release of OpenShift Container Storage.

3.5.6. Red Hat tested solution: storage

In our solution, we use the built-in director templates provided by ceph-ansible to deploy Red Hat Ceph Storage 4 with director. The deployment uses dedicated storage nodes for the Ceph ODSs and colocates Monitors on the OpenStack controllers.

This director deployment pattern also automatically configures Ceph as the backing store for Compute (nova) VMs, cinder volumes, and glance images. This configuration provides fast, highly-available, and resilient storage for OpenStack's primary services.

We install the Shared File Systems service (manila) with director and configure it with a Red Hat Ceph Storage back end that uses Ganesha for NFS.

Please Note: The use of Manila CSI with Ganesha and Ceph is Technology Preview and requires a support exception for production workloads.

For supported and certified third party storage drivers and vendors for Manila CSI, see the Red Hat Partner Ecosystem Catalog.

In our solution, Red Hat Ceph Storage supports the following services:

- Block (cinder)

- Image (glance)

- Object (swift) via radosgw (RGW), which is a gateway for the Ceph RADOS object store.

In our solution, OpenShift consumes storage for a variety of purposes.

- Storage for container workloads is provided by cinder (RWO) & manila (RWX).

- Storage for the OpenShift registry is provided by ceph RGW (object storage) and the OpenStack tenant is granted the swift operator role.

- The RHCOS image file and the bootstrap ignition files for each tenant's cluster are stored in OpenStack's image service, glance. Each RHCOS image remains until the cluster is removed. The ignition image is temporary and deleted by the installer at the end of the installation process.

3.6. NETWORKING

Red Hat OpenStack Platform and Red Hat OpenShift Container Platform have independent and mature networking solutions.

Choosing the right solution is dependent on the specifics of your workloads. Try to select networking solutions that meet the needs of your OpenShift application and minimize complexity.

While these solutions can be layered, consider the requirements of your workloads to avoid duplicating networking functionality.

3.6.1. OpenStack networking

The OpenStack Neutron API provides a common abstraction layer for various back end network plug-ins. OpenStack Platform supports several back-end network plug-ins:

- ML2 OVS
- OVN
- Commercial SDNs

A provider network assigns a range of externally accessible floating IP addresses to tenant instances. Remote clients use the floating IP addresses to access the following services:

- OpenShift applications
- The OpenShift API endpoint
- The web console

For more information about OpenStack networking, see the Networking Guide.

3.6.1.1. OpenStack networking (neutron) back end plug-ins

Like many other OpenStack components, the networking subsystem (neutron) is pluggable so that customers can choose the solution that best fits their business and technological requirements. For Red Hat OpenStack Platform, customers can choose from supported options.

The default neutron back end implemented by Red Hat OpenStack Platform 16.1 is OVN.

3.6.1.1.1. Open vSwitch (OVS)

OVS is an open-source, multi-layer software switch designed to be used as a virtual switch in virtualized server environments and across multiple Red Hat products. OVS has been a default neutron back end in OSP for multiple releases and is fully tested and supported.

3.6.1.1.2. Open Virtual Network (OVN)

Red Hat OpenStack Platform 16.1 uses the Open Virtual Network (OVN) component of Open vSwitch

by default. OVN is a network virtualization solution that is built into the Open vSwitch (OVS) project. OVN supports the Neutron API, and offers a clean and distributed implementation of the most common networking capabilities such as bridging, routing, security groups, NAT, and floating IPs.

When using OVN as a back end for neutron, Generic Network Virtualization Encapsulation (GENEVE) is used for network encapsulation. Using OVN and GENEVE has the following advantages:

- Increased flexibility through an extendable header format
- Distribution of L3 traffic by default.

Versions earlier than Red Hat OpenStack Platform 16.1 used OVS's ML2 plug-in by default.

3.6.1.1.3. Third party SDN solution

Multiple networking vendors support their own plug-ins for Neutron. For future iterations of OpenShift on OpenStack, there might be increased partnerships and testing to support additional third party solutions.

3.6.1.2. Neutron back ends and Red Hat OpenStack Platform

You should choose a neutron back end that meets your unique circumstances. Beginning with the release of Red Hat OpenStack Platform 13, we support both OVS/ML2 and OVN.

When planning for OpenShift 4.6 and 4.7 on OpenStack 16.1, we strongly recommend the use of the default back end of OVN to benefit from the advantages of Kuryr and its use of Octavia.

To assist with choosing a neutron back end when deploying OpenShift on OpenStack, consider the following:

- OVN is fully tested and supported for OpenShift on OpenStack installations.
- OVN is the default neutron back end for Red Hat OpenStack Platform 16.1 and later.
- Red Hat QE has tested OpenShift on OpenStack with both OVN and OVS and, in many cases, found OVN to perform better for some workloads.
- OVN is preferred for Kuryr installations and is therefore the back end that we use in our solution.

3.6.1.3. OpenStack load balancing

Red Hat OpenStack Platform uses the Octavia project for its Load Balancing as a Service (LBaaS) implementation. Octavia improves upon previous OpenStack LBaaS implementations and is the reference implementation for Neutron LBaaS version 2. Octavia implements load balancing by deploying a fleet of virtual machines, called amphorae, that run load balancing software.

Octavia has the following characteristics:

- It is a highly scalable solution.
- It can be a resource-heavy solution when used with OpenShift because it requires a virtual machine instance acting as a load balancer per OpenStack service for each tenant.

You can use Octavia with OVN or ml2/OVS OpenStack networking solutions. Using Octavia with OVN has the following benefits

- Octavia offers an additional driver that does not require the use of VMs for OpenShift services. The Octavia OVN driver (ovn-octavia) implements load balancing logic within OpenFlow rules. This means that load balancers are created faster and with less resource overhead than when you create a VM for each balancer via using the standard amphora driver (either with OVN or ml2/OVS).

- Ovn-octavia supports virtual networking for both VMs and containers, which means that it can be used with OpenShift's Kuryr CNI plug-in.

With Openshift on OpenStack install you will still see one VM-based (amphora) balancer. This is created by the Cluster Network Operator to manage access to the OpenShift API.

Kuryr is the recommended OpenShift SDN for OpenShift on OpenStack.

For more details on the OVN Octavia Provider, including its limitations, review the upstream documentation and consult a Red Hat Solution Architect or Support Associate.

3.6.1.4. Red Hat tested solution: networking

In our solution, we use the default neutron networking back end of Red Hat OpenStack Platform 16.1: OVN.

Octavia is deployed with director and the installation program uses the ovn-octavia driver for load balancing via Kuryr.

When choosing an SDN for an OpenShift on OpenStack deployment, you must consider the networking requirements for your workloads. When running OpenShift SDN on a standard OpenStack deployment, the network is subject to double encapsulation. Double encapsulation happens when you run an encapsulated OpenShift SDN over an already encapsulated OpenStack network. This is true for both OVS and OVN-based implementations.

Although running under double encapsulation can cause performance and complexity issues for an installation, all public clouds use it. Review the requirements of your workloads carefully to best understand the true impact of double encapsulation as it relates to your own workloads.

To learn more about networking and OpenShift, see Understanding networking in the official documentation.

3.6.1.5. OpenShift SDN

The OpenShift SDN is the default SDN solution for OpenShift on OpenStack. Full details can be found in the About OpenShift SDN section of the documentation.

3.6.1.6. OVN-Kubernetes

The OVN-Kubernetes Container Network Interface (CNI) plug-in is the next generation SDN solution for OpenShift networking. Built to take advantage of OVN (Open Virtual Network) and supported by a large, upstream vendor-agnostic community, OVN-Kubernetes has the following benefits:

- Uses OVN to manage network traffic flows

- Implements Kubernetes network policy support

- Introduces the Geneve (Generic Network Virtualization Encapsulation) protocol for overlay networking between nodes.

Beginning with OpenShift 4.6, OVN-Kubernetes is supported for use in OpenShift as an SDN provider if desired.

For more information about OVN-Kubernetes features, see the documentation.

3.6.1.7. Kuryr SDN

The Kubernetes Container Networking Interface (CNI) specification provides a mechanism to implement generic plug-in-based networking solutions for OpenShift application containers. You can replace the default OpenShift networking plug-in with Kuryr (kuryr-kubernetes plug-in), one of several alternative CNI plug-in options.

Kuryr can improve network performance through various methods:

- Kuryr enables OpenStack's Neutron-based networking solution to configure networking for OpenShift's Kubernetes containers, rather than layer the two SDNs on top of each other.
- The Kuryr plug-in makes it possible to run both OpenStack VMs and Kubernetes containers on the same Neutron network.
    - Kuryr achieves this by working with OpenStack Networking service (neutron), along with OpenStack's Load Balancing service component (Octavia). This provides combined networking for pods and services.

To learn more about Kuryr's effect on network performance, see this blog on performance testing.

3.6.1.7.1. Kuryr considerations

Kuryr has the following characteristics:

- Kuryr is a CNI plug-in
- Using Kuryr removes double encapsulation as it allows Neutron to control both VM and Containers, removing the need to "layer," or duplicate, networking functions.
- Kuryr interacts directly with OpenStack Load Balancing (Octavia) to implement relevant OpenShift Services on demand
- Kuryr interacts directly with OpenStack Load Balancing (Octavia) to create relevant OpenStack services on demand.
- For OpenShift on OpenStack, Kuryr must only be used with the ovn-octavia provider plug-in. The IPI installer selects it by default if Kuryr is specified for the networking type .

For a detailed review of Kuryr, including its limitations, see Installing a cluster on OpenStack with Kuryr .

3.6.2. Red Hat tested solution: Networking

For our solution, we use Kuryr-SDN for OpenShift networking.

3.7. DNS

The OpenShift 4 installer greatly simplifies the DNS requirements seen in previous OpenShift on OpenStack installations. All internal DNS resolution for the cluster, certificate validation, and bootstrapping is provided through a self-hosted, installer-controlled solution utilising the mDNS plugin

for CoreDNS. This solution was developed to perform DNS lookups based on discoverable information from mDNS. It is also able to resolve the name of the nodes.

With this solution, control plane nodes, worker nodes, and the bootstrap node do not need their IPs added, manually or dynamically, to any form of public DNS; it is entirely self-contained.

Please see more detailed documentation at the OpenStack installer-provisioned infrastructure Networking Infrastructure page.

With this solution there is no need to run a self-hosted name server and no requirement for external solutions such as the Designate DNSaaS project.

3.7.1. DNS considerations

You must meet the following DNS requirements to complete the installation:

- The installation host must resolve the OpenShift API address to an OpenStack floating IP (API DNS) that it can reach.
- The bootstrap node must be able to resolve the addresses of public container registries in order to build the bootstrap cluster.

You must meet the following DNS requirement after installation is complete:

- A wild card domain must resolve an additional floating IP to the ingress port.

3.7.1.1. API DNS

The OpenShift API floating IP is required to be in DNS before you install the cluster. The following address spaces must resolve:

```
api.<cluster name>.<base domain>
```

This floating IP is assigned in the **install-config.yaml** file via the **lbFloatingIP** value of the OpenStack platform section of **install-config.yaml**:

An example of how to assign the OpenShift's API address to a specific Floating IP value.

```
platform:
  openstack:
...
    lbFloatingIP:     "10.46.43.176"
```

IMPORTANT

In versions after, but not including, Openshift 4.7, we replace **lbFloatingIP** with **apiFloatingIP**. The functionality of this variable does not change, but the new name adds clarity by indicating that the IP referenced is the floating IP on the external network for use with the OpenShift API. **apiVIP** and **apiFloatingIP** do not refer to the same thing: **apiVIP** is for the Machine network and **apiFloatingIP** is from the external network.

3.7.1.2. Apps DNS

This domain is used for access to the applications running in OpenShift. In your DNS, you create a wildcard entry for this floating IP to resolve the following naming structure:

```
*.apps.<cluster name>.<base domain>.
```

Beginning with OpenShift 4.6, you can automatically associate this OpenShift apps IP to your cluster by using the **ingressFloatingIP** value in your **install-config.yaml**. This variable does not exist in earlier versions of OpenShift. Do not confuse **ingressVIP** with **ingressFloatingIP**: **ingressVIP** is for the Machine network and **ingressFloatingIP** is from the external network.

3.7.1.3. Bootstrap node

The bootstrap node must be able to resolve external registry domain names directly because it retrieves the basic resources to stand up the bootstrap and production clusters.

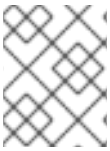
3.7.1.4. Additional DNS capabilities

3.7.1.4.1. externalDNS

OpenStack tenants whose clouds do not automatically offer name resolution to tenant subnets can set this during an OpenShift installation by using the optional **externalDNS** value in **install-config.yaml**. **externalDNS** instructs the installer to add the DNS IPs to the OpenShift subnet it creates. This value is an array so it can contain multiple entries:

An example of how to set each installer-created subnets DNS value.

```
externalDNS:      ["10.46.0.31","10.46.0.32"]
```



NOTE

You must add **externalDNS** manually to the install file. **externalDNS** is not available from the guided installation.

3.7.2. Red Hat tested solution: DNS

In our solution, we pre-allocate and pre-configure two FIPs in DNS to provide an API address and an application address.

We also use the **externalDNS** parameter to allow the installer-built subnets to offer external DNS resolution to their instances.

3.8. SECURITY AND AUTHENTICATION

Both OpenShift and OpenStack support role-based access control and flexible options for integrating with existing user authentication systems. Additionally, both inherit the security Red Hat Enterprise Linux native security features such as SELinux.

3.8.1. Authentication

The OpenStack identity service stores user credentials in two ways:

- In its local state database.

- In an external LDAP-compliant directory server.

OpenShift controller nodes issue tokens to authenticate user requests with the API. OpenShift supports various identity providers, including HTPassword and LDAP.

There is detailed integration between OpenShift and OpenStack identity providers. Administrators can configure the OpenShift keystone identity provider with the OpenStack Keystone service. This configuration allows users to log in to OpenShift Container Platform with their Keystone credentials. Full details are in the Configuring a Keystone identity provider section of the OpenShift documentation.

3.8.2. Security

Many of the security best practices are embedded into a default OpenStack Platform deployment. Red Hat OpenStack Platform meets various levels of standard security compliance such as ANSSI and FedRamp. This solution as documented here is not a comprehensive resource for securing OpenStack and assumes a relatively basic, but production supported, level of security. It might not be suitable for all enterprise requirements.

To learn more about the best practices for securing OpenStack, see Red Hat OpenStack Platform 16 Security and Hardening Guide.

3.8.2.1. OpenStack security groups

When installing OpenShift on OpenStack, the installer creates the necessary OpenStack security groups for a functional installation. The installer creates security groups for the control plane nodes, the worker nodes, and Kuryr.

You should review the security groups for a clear understanding of how they might affect your own internal security requirements.

For details of the rules created, you can review the upstream code for both control plane nodes and worker nodes.

For details on how these groups are used, see Security Group Rules in the upstream Cluster API documentation for OpenStack.

To learn how to add additional security groups directly at install time, see Additional Security Groups.

3.8.3. Red Hat tested solution: security and authentication

The installer-provisioned infrastructure method creates and manages all OpenStack security groups and rules necessary. These groups completely isolate the tenant's OpenShift install from others on the cloud.

We use Transport Layer Security (TLS) to encrypt the OpenStack public API endpoints.

- We use a self signed certificate and a local certificate authority on the installation host.
- After we enable the OpenStack TLS public endpoint encryption, we import the certificates to any hosts issuing commands against the endpoints.
- Because we use the director host to run the installer, all interaction with encrypted endpoints is from the director host.

For more information about this procedure on OpenStack, see Creating an SSL/TLS Certificate Signing Request.

CHAPTER 4. REQUIREMENTS AND OPTIONS FOR PROVIDING DISK TO OPENSHIFT NODES (ETCD)

When you install OpenShift, you must ensure that fast, redundant disk is available.

4.1. MINIMUM DISK REQUIREMENTS FOR ETCD

The control plane VMs must meet the known resource requirements to run the etcd key-value store. Fast disks are a requirement for etcd stability and performance. Red Hat requires that all controllers have access to fast disk (SSD or better) to ensure stability and guarantee supportability.

To learn more about the etcd disk requirements, see the etcd documentation.

4.2. OPTIONS TO PROVIDE DISK TO OPENSHIFT NODES

You have several options for providing storage to your cluster. To help you make a decision, we highlight the following three options:

- Ephemeral on Computes
- Ceph-backed Ephemeral
- Volumes Provided by Cinder

4.2.1. Ephemeral on Computes

Running VMs with their disk hosted locally on a compute node is the easiest way to provide storage to the control plane. This is the default for an OpenStack installation so you don't have to do any configuration. The volumes of all nova instances are stored directly on the compute hypervisor's disk. To do this, the disk must be SSD or better. When using this method, be aware that the loss of a physical OpenStack compute node destroys the VM instance and, therefore, an etcd member.

4.2.2. Ceph-backed ephemeral

This is the default configuration for a Ceph environment deployed with director. This configuration method sets the variable **NovaEnableRBDBackend** to **True**. This setting instructs all nova back end disks to utilize Ceph RBD which has many advantages:

- The volumes are ephemeral but are sourced from the Ceph cluster.
- The Ceph storage cluster itself provides resilience.
- The Ceph cluster is more customizable as Ceph can implement tiering options to define how it's storage disks are used.

We use this method in our solution because of the following benefits:

- We can configure specific disks within the Ceph cluster to be used for the fast disk requirements of etcd.
- We can ensure added resilience for live migration.

This configuration method is easily implemented by default when using director to deploy Ceph:

The default values set when using director to deploy Ceph

```
# cat /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-ansible.yaml
...
CinderEnableIscsiBackend: false
CinderEnableRbdBackend: true
CinderBackupBackend: ceph
NovaEnableRbdBackend: true
GlanceBackend: rbd
...
```

4.2.3. Volumes provided by cinder

Cinder is the OpenStack block storage service that provides an API front end for multiple back end storage providers, including Ceph.

Nova instances can request a volume from the Cinder API. Cinder then requests the storage, via the back end plug-in, to utilise it with the instance.

OpenStack deployments with Ceph create a default setup that you can use with the OpenShift installer to provide Ceph-backed Cinder volumes to OpenShift nodes. This method gives you a highly granular solution where you can do the following actions:

- Provision Ceph-backed Cinder volumes from a fast storage pool in Ceph to ensure etcd performance.
- Create complex and refined tiers (classes) of storage to present to the different nodes.
- Assign volumes from different back end storage tiers to control plane and worker nodes .

CHAPTER 5. SOLUTION INSTALLATION DETAILS

To assist the reader, the OpenStack install command and OpenShift install configuration files for this solution are provided here for reference. These examples are not intended to replace a thorough understanding or comprehensive testing when implementing OpenShift on OpenStack. They are offered without guarantee of accuracy or support from Red Hat.

5.1. OPENSTACK INSTALL COMMAND

```
 openstack overcloud deploy \
  --timeout 100 \
  --templates /usr/share/openstack-tripleo-heat-templates \
  --environment-file /usr/share/openstack-tripleo-heat-templates/environments/manila-
cephfsganesha-config.yaml \
  --environment-file /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-
rgw.yaml \
  --environment-file /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-
mds.yaml \
  --environment-file /usr/share/openstack-tripleo-heat-templates/environments/services/octavia.yaml \
  --libvirt-type kvm \
  --stack overcloud \
  -r /home/stack/hybrid_templates/roles/roles_data.yaml \
  -e /home/stack/hybrid_templates/roles/nodes.yaml \
-e /home/stack/hybrid_templates/internal.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/network-isolation.yaml \
-e /home/stack/hybrid_templates/network/network-environment.yaml \
-e /home/stack/hybrid_templates/network/dvr-override.yaml \
-e /home/stack/hybrid_templates/enable-tls.yaml \
-e /home/stack/hybrid_templates/inject-trust-anchor.yaml \
-e /home/stack/hybrid_templates/public_vip.yaml \
-e /usr/share/openstack-tripleo-heat-templates/environments/ssl/tls-endpoints-public-ip.yaml \
-e /home/stack/hybrid_templates/hostnames.yml \
-e /usr/share/openstack-tripleo-heat-templates/environments/ceph-ansible/ceph-ansible.yaml \
-n /usr/share/openstack-tripleo-heat-templates/network_data_ganesha.yaml    \
-e /home/stack/hybrid_templates/hostnames.yaml \
-e /home/stack/hybrid_templates/debug.yaml \
-e /home/stack/hybrid_templates/custom-storage.yaml \
-e /home/stack/hybrid_templates/config_heat.yaml \
-e ~/containers-prepare-parameter.yaml \
-e /home/stack/hybrid_templates/manila-cephganesha.yaml \
--log-file overcloud_deployment_50.log
```

5.2. OPENSHIFT INSTALL CONFIGURATION FILE

```
apiVersion: v1
baseDomain: "shiftstack.com"
clusterID:  "9242da7a-81a2-5fcc-b95f-ab2240f5170c"
compute:
- name: worker
platform:
    openstack:
      zones: ['AZ0', 'AZ1', 'AZ2']
      additionalNetworkIDs:
      - 2f256706-dd68-45c9-970a-8376f02410e8
```

```
  replicas: 3
controlPlane:
  name: master
  platform:
    openstack:
      zones: ['AZ0', 'AZ1', 'AZ2']
  replicas: 3
metadata:
  name: "ostest"
networking:
  clusterNetworks:
  - cidr:             10.128.0.0/14
    hostSubnetLength: 9
  serviceCIDR: 172.30.0.0/16
  machineCIDR: 10.196.0.0/16
  type: "Kuryr"
platform:
  openstack:
    cloud:            "shiftstack"
    externalNetwork:  "nova"
    region:           "regionOne"
    computeFlavor:    "m4.xlarge"
    externalDNS:      ["10.46.0.31"]
    lbFloatingIP:     "10.46.43.176"
    ingressFloatingIP:     "10.46.43.181"
pullSecret: |  {PRIVATE}
sshKey: |
  ssh-rsa ABCDEFGHIJKLMNOPQRSTUVWXZY1234567890
```

CHAPTER 6. SUMMARY

With Red Hat OpenShift Platform 4, Red Hat OpenStack Platform 16.1, and Red Hat Ceph Storage 4, organisations have access to a comprehensive and prescriptive installation experience for their on-premises container infrastructure.

This Red Hat Tested Solution showcases a prescriptive and pre-validated private cloud solution from Red Hat that provides rapid provisioning and lifecycle management of containerized infrastructure, virtual machines (VMs), and associated application and infrastructure services.

The Red Hat Quality Engineering teams (QE) have tested and validated the implementation as presented in this solution. Organizations seeking to operationalize this solution quickly can be assured that all options represented are both fully tested as well as fully supported by Red Hat.

Red Hat OpenShift Container Platform, Red Hat OpenStack Platform, and Red Hat Ceph Storage are the key architectural components of this solution. This integration is a key component to hybrid and multi-cloud solutions with OpenShift Container Platform serving as the common container and platform across any deployment footprint.

CHAPTER 7. ADDITIONAL LINKS AND REFERENCES

7.1. PREVIOUS REFERENCE ARCHITECTURES

- Deploying Red Hat OpenShift Container Platform 3.11 on Red Hat OpenStack Platform 13
- Deploying Red Hat OpenShift Container Platform 4.4 on Red Hat OpenStack Platform 13 and 16

7.2. HELPFUL DOCUMENTATION LINKS

7.2.1. Red Hat OpenShift Container Platform

- Official Product Documentation
- Installing OpenShift Container Platform on OpenStack clusters
- Upstream installer code and documentation
- Installation configuration file parameters for 4.6
- Installation configuration file parameters for 4.7
- Learn more about OpenShift Container Platform

7.2.2. Red Hat OpenStack Platform

- Official Product Documentation
- Deploying an overcloud with containerized Red Hat Ceph

7.2.3. Red Hat Ceph Storage

- Official Product Documentation

7.2.4. Red Hat OpenShift Container Storage

- Official Product Documentation