



## **Archived Solutions**

### **NetApp Solutions**

NetApp  
August 03, 2021

This PDF was generated from <https://docs.netapp.com/us-en/netapp-solutionshttps://www.netapp.com/us/media/nva-1124-design.pdf> on August 03, 2021. Always check [docs.netapp.com](https://docs.netapp.com) for the latest.

# Table of Contents

Archived Solutions .....	1
NVA-1149: NetApp HCI for Red Hat OpenShift on Red Hat Virtualization .....	1

# Archived Solutions

## NVA-1149: NetApp HCI for Red Hat OpenShift on Red Hat Virtualization

Alan Cowles and Nikhil M Kulkarni, NetApp

NetApp HCI for Red Hat OpenShift on Red Hat Virtualization (RHV) is a best-practice deployment guide for the fully automated install of Red Hat OpenShift through the Installer Provisioned Infrastructure (IPI) method onto the verified enterprise architecture of [NVA-1148: NetApp HCI with Red Hat Virtualization](#). The purpose of this NetApp Verified Architecture deployment guide is to provide a concise set of verified instructions to be followed for the deployment of the solution. The architecture and deployment methods described in this document have been validated jointly by subject matter experts at NetApp and Red Hat to provide a best-practice implementation of the solution.

### Use Cases

The NetApp HCI for Red Hat OpenShift on RHV solution is architected to deliver exceptional value for customers with the following use cases:

- Infrastructure to scale on demand with NetApp HCI
- Enterprise virtualized workloads in RHV
- Enterprise containerized workloads in Red Hat OpenShift

### Business Value

Enterprises are increasingly adopting DevOps practices to create new products, shorten release cycles, and rapidly add new features. Because of their innate agile nature, containers and microservices play a crucial role in supporting DevOps practices. However, practicing DevOps at a production scale in an enterprise environment presents its own challenges and imposes certain requirements on the underlying infrastructure, such as the following:

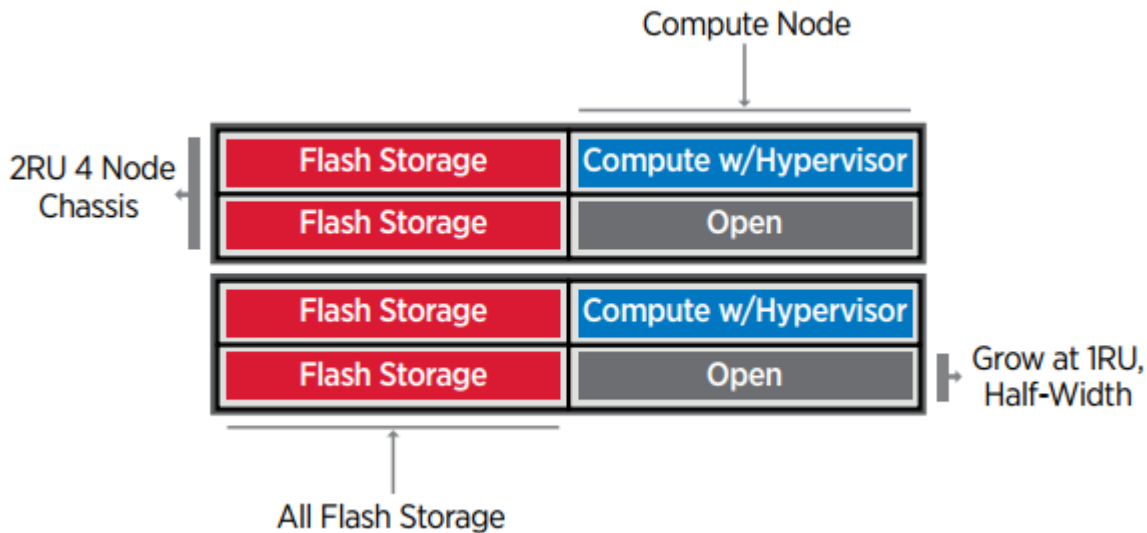
- High availability at all layers in the stack
- Ease of deployment procedures
- Nondisruptive operations and upgrades
- API-driven and programmable infrastructure to keep up with microservices agility
- Multitenancy with performance guarantees
- Ability to run virtualized and containerized workloads simultaneously
- Ability to scale infrastructure independently based on workload demands

NetApp HCI for Red Hat OpenShift on RHV acknowledges these challenges and presents a solution that helps address each concern by implementing the fully automated deployment of Red Hat OpenShift IPI on the RHV enterprise hypervisor. The remainder of this document details the components used in this verified architecture.

### Technology Overview

## NetApp HCI

NetApp HCI is an enterprise-scale, disaggregated hybrid cloud infrastructure (HCI) solution that delivers compute and storage resources in an agile, scalable, and easy-to-manage two-rack unit (2RU), four-node building block. It can also be configured with 1RU compute and server nodes. The minimum deployment depicted in the figure below consists of four NetApp HCI storage nodes and two NetApp HCI compute nodes. The compute nodes are installed as Red Hat Virtualization Hosts (RHV-H) hypervisors in a high-availability (HA) cluster. This minimum deployment can be easily scaled to fit customer enterprise workload demands by adding additional NetApp HCI storage or compute nodes to expand available resources.



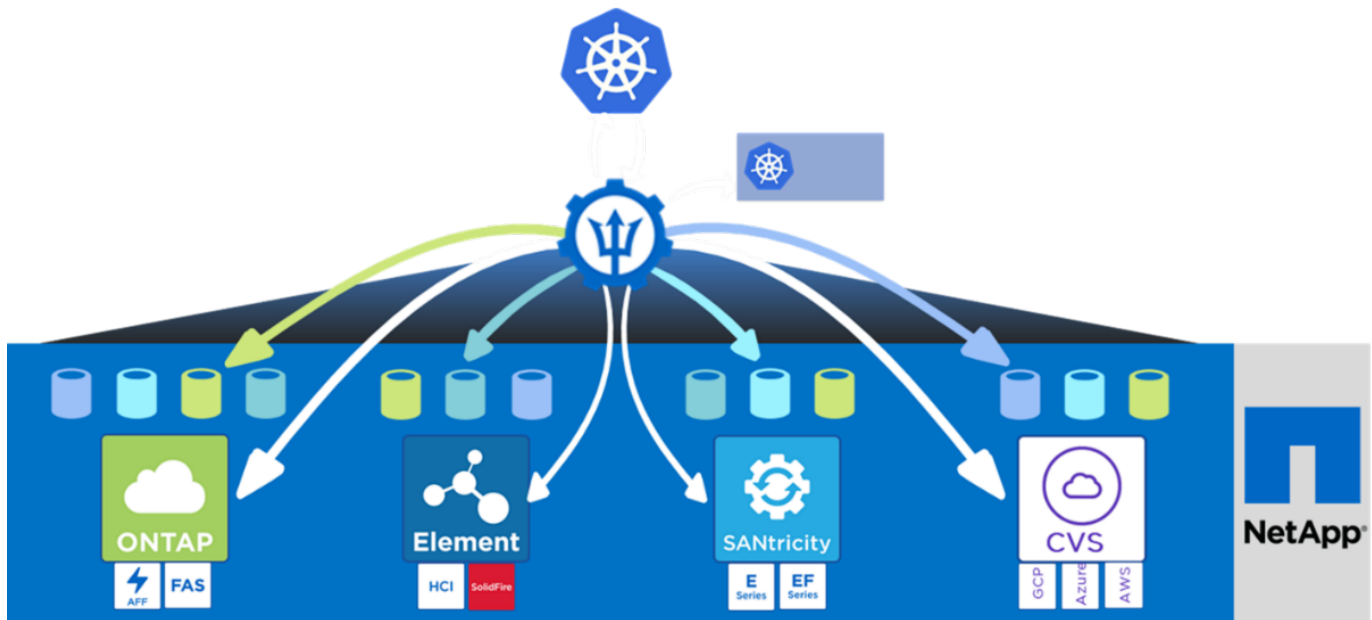
The design for NetApp HCI for Red Hat Virtualization consists of the following components in a minimum starting configuration:

- NetApp H-Series all-flash storage nodes running NetApp Element software
- NetApp H-Series compute nodes running the Red Hat Virtualization RHV-H hypervisor

For more information about compute and storage nodes in NetApp HCI, see [NetApp HCI Datasheet](#).

## NetApp Trident

Trident is a NetApp open-source and fully supported storage orchestrator for containers and Kubernetes distributions, including Red Hat OpenShift. It works with the entire NetApp storage portfolio, including the NetApp Element storage system that is deployed as a part of the NetApp HCI solution. Trident provides the ability to accelerate the DevOps workflow by allowing end users to provision and manage storage from their NetApp storage systems, without requiring intervention from a storage administrator. An administrator can configure a number of storage backends based on project needs, and storage system models that allow for any number of advanced storage features, such as: compression, specific disk types, or QoS levels that guarantee a certain performance. After they are defined, these backends can be leveraged by developers as part of their projects to create persistent volume claims (PVCs) and attach persistent storage to their containers on demand.



## Red Hat Virtualization

RHV is an enterprise virtual data center platform that runs on Red Hat Enterprise Linux (RHEL) and uses the KVM hypervisor.

For more information about RHV, see the [Red Hat Virtualization website](#).

RHV provides the following features:

- **Centralized management of VMs and hosts.** The RHV manager runs as a physical or virtual machine (VM) in the deployment and provides a web-based GUI for the management of the solution from a central interface.
- **Self-hosted engine.** To minimize the hardware requirements, RHV allows RHV Manager (RHV-M) to be deployed as a VM on the same hosts that run guest VMs.
- **High availability.** In event of host failures, to avoid disruption, RHV allows VMs to be configured for high availability. The highly available VMs are controlled at the cluster level using resiliency policies.
- **High scalability.** A single RHV cluster can have up to 200 hypervisor hosts enabling it to support requirements of massive VMs to hold resource-greedy, enterprise-class workloads.
- **Enhanced security.** Inherited from RHV, Secure Virtualization (sVirt) and Security Enhanced Linux (SELinux) technologies are employed by RHV for the purposes of elevated security and hardening for the hosts and VMs. The key advantage from these features is logical isolation of a VM and its associated resources.

## Red Hat Virtualization Manager

RHV-M provides centralized enterprise-grade management for the physical and logical resources within the RHV virtualized environment. A web-based GUI with different role-based portals are provided to access RHV-M features.

RHV-M exposes configuration and management of RHV resources via open-source, community-driven RESTful API. It also supports full-fledged integration with Red Hat CloudForms and Red Hat Ansible for automation and orchestration.

## Red Hat Virtualization Hosts

Hosts (also called hypervisors) are the physical servers that provide hardware resources for the VMs to run on. Kernel-based Virtual Machine (KVM) provides full virtualization support, and Virtual Desktop Server Manager (VDSM) is the host agent that is responsible for communication of the hosts with the RHV-M.

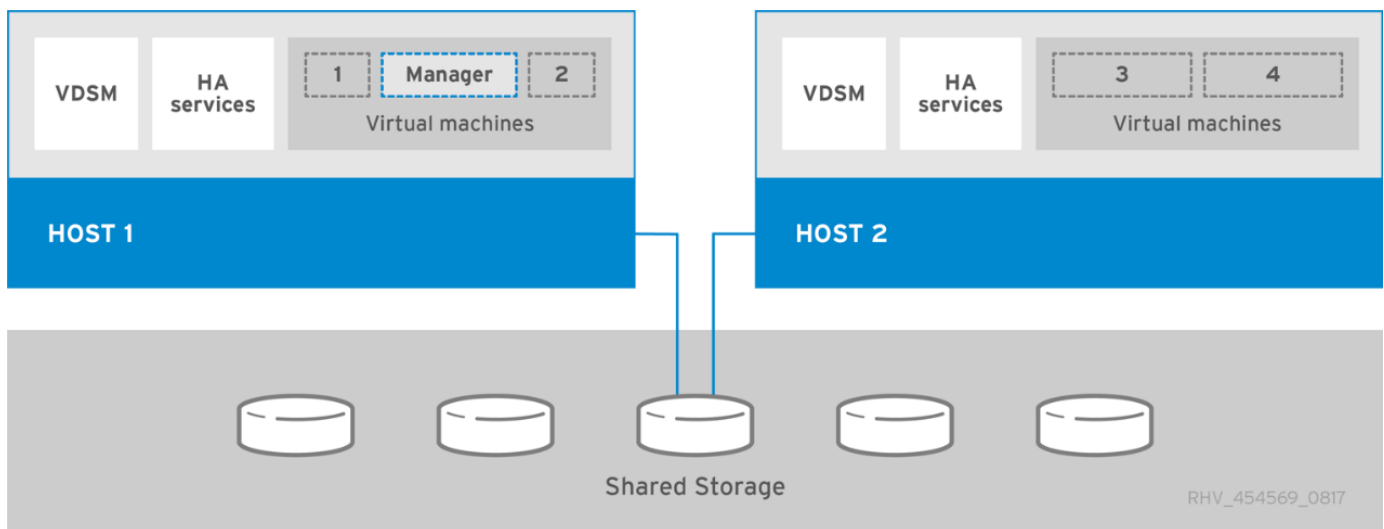
Two types of hosts are supported in RHV are RHV-H and RHEL hosts:

- RHV-H is a light-weight minimal operating system based on RHEL, optimized for ease of setting up physical servers as RHV hypervisors.
- RHEL hosts are servers that run the standard RHEL operating system and are later configured with the required subscriptions to install the packages required to permit the physical servers to be used as RHV hosts.

## Red Hat Virtualization Architecture

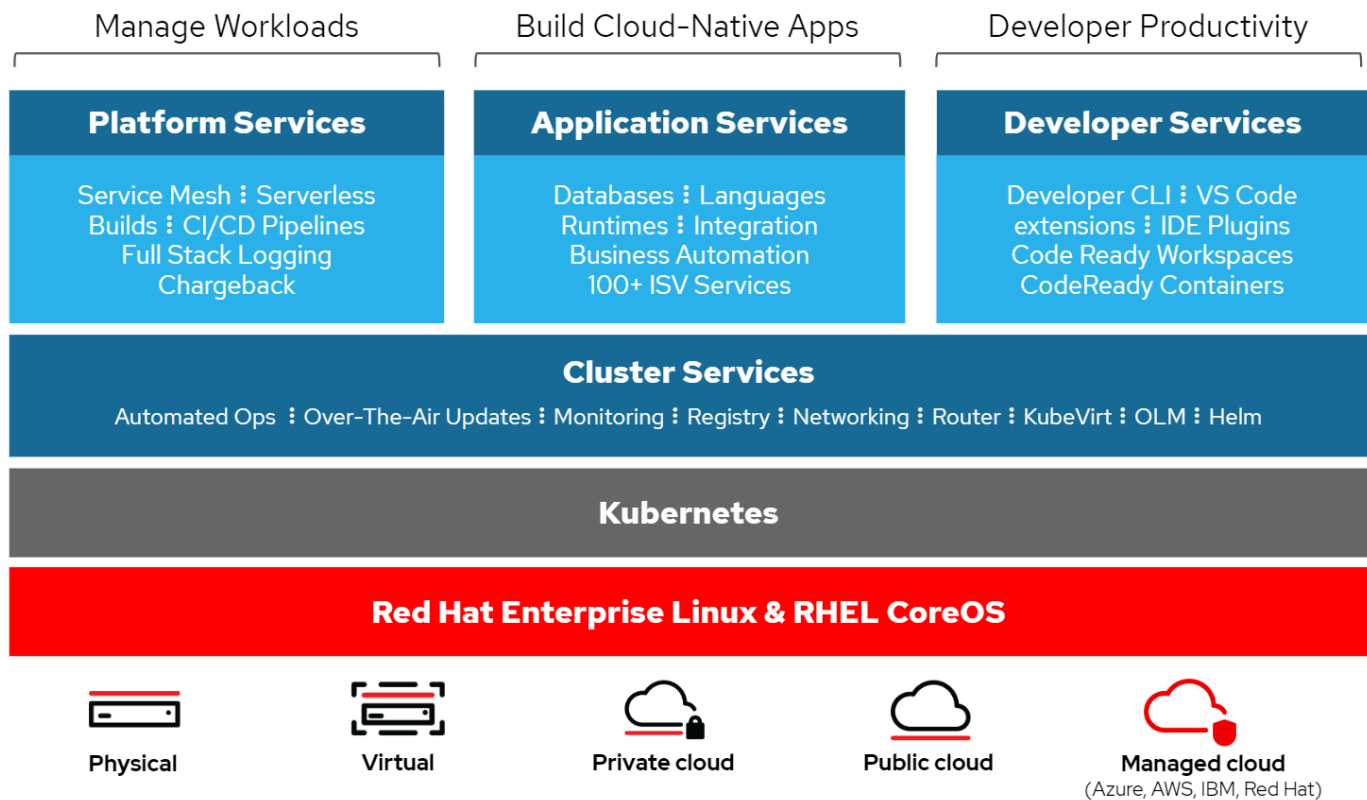
RHV can be deployed in two different architectures: with the RHV-M as a physical server in the infrastructure or with the RHV-M configured as a self-hosted engine. The self-hosted engine deployment, where the RHV-M is a VM hosted in the same environment as other VMs, is recommended and used specifically in this deployment guide.

A minimum of two self-hosted nodes are required for high availability of guest VMs and RHV-M as depicted in the figure below. For ensuring the high availability of the manager VM, HA services are enabled and run on all the self-hosted engine nodes.



## Red Hat OpenShift Container Platform

Red Hat OpenShift Container Platform is a fully supported enterprise Kubernetes platform. Red Hat makes several enhancements to open-source Kubernetes to deliver an application platform with all the components fully integrated to build, deploy, and manage containerized applications. With Red Hat OpenShift 4.4, the installation and management processes have been streamlined through the IPI method which has been deployed in this solution. By leveraging this deployment method, a fully functional OpenShift cluster providing metering and monitoring at both the cluster and application level can be fully configured and deployed on top of Red Hat Virtualization in less than an hour. OpenShift nodes are based upon RHEL CoreOS, an immutable system image designed to run containers, based on RHEL, which can be upgraded or scaled easily on demand as the needs of the end user require, helping to deliver the benefits of the public cloud to the local data center.



Next: [Architectural Overview: NetApp HCI for Red Hat OpenShift on RHV](#).

### Abstract

This NetApp HCI for Red Hat OpenShift on Red Hat Virtualization (RHV) deployment guide is for the fully automated installation of Red Hat OpenShift through the Installer Provisioned Infrastructure (IPI) method onto the verified enterprise architecture of NetApp HCI for Red Hat Virtualization described in NVA-1148: NetApp HCI with Red Hat Virtualization. This reference document provides deployment validation of the Red Hat OpenShift solution, integration of the NetApp Trident storage orchestrator, and a solution verification consisting of an example application deployment.

### Architectural Overview: NetApp HCI for Red Hat OpenShift on RHV

#### Hardware Requirements

The following table lists the minimum number of hardware components that are required to implement the solution. The hardware components that are used in specific implementations of the solution might vary based on customer requirements.

Hardware	Model	Quantity
NetApp HCI compute nodes	NetApp H410C	2
NetApp HCI storage nodes	NetApp H410S	4
Data switches	Mellanox SN2010	2
Management switches	Cisco Nexus 3048	2

## Software Requirements

The following table lists the software components that are required to implement the solution. The software components that are used in any implementation of the solution might vary based on customer requirements.

Software	Purpose	Version
NetApp HCI	Infrastructure (compute/storage)	1.8
NetApp Element	Storage	12.0
NetApp Trident	Storage orchestration	20.04
RHV	Virtualization	4.3.9
Red Hat OpenShift	Container orchestration	4.4.6

[Next: Design Considerations: NetApp HCI for Red Hat OpenShift on RHV](#)

## Design Considerations: NetApp HCI for Red Hat OpenShift on RHV

### Network Design

The Red Hat OpenShift on RHV on HCI solution uses two data switches to provide primary data connectivity at 25Gbps. It also uses two additional management switches that provide connectivity at 1Gbps for in-band management for the storage nodes and out-of-band management for IPMI functionality. OCP uses the logical network on the RHV for the cluster management. This section describes the arrangement and purpose of each virtual network segment used in the solution and outlines the pre-requisites for deployment of the solution.

### VLAN Requirements

The NetApp HCI for Red Hat OpenShift on RHV solution is designed to logically separate network traffic for different purposes by using virtual local area networks (VLANs). NetApp HCI requires a minimum of three network segments. However, this configuration can be scaled to meet customer demands or to provide further isolation for specific network services. The following table lists the VLANs that are required to implement the solution, as well as the specific VLAN IDs that are used later in the verified architecture deployment.

VLANs	Purpose	VLAN ID
Out-of-band management network	Management for HCI nodes and IPMI	16
In-band management network	Management for HCI nodes, ovirtmgmt, and VMs	1172
Storage network	Storage network for NetApp Element	3343
Migration network	Network for virtual guest migration	3345

### Network Infrastructure Support Resources

The following infrastructure should be in place prior to the deployment of the OpenShift Container Platform (OCP) on Red Hat Virtualization on NetApp HCI solution:

- At least one DNS server which provides a full host-name resolution that is accessible from the in-band management network and the VM network.



- At least one NTP server that is accessible from the in-band management network and the VM network.
- (Optional) Outbound internet connectivity for both the in-band management network and the VM network.
- RHV cluster should have at least 28x vCPUs, 112GB RAM, and 840GB of available storage (depending on the production workload requirements).

[Next: Deploying NetApp HCI for Red Hat OpenShift on RHV](#)

## Deployment Summary: NetApp HCI for Red Hat OpenShift on RHV

The detailed steps provided in this section provide a validation for the minimum hardware and software configuration required to deploy and validate the NetApp HCI for Red Hat OpenShift on RHV solution.

Deploying Red Hat OpenShift Container Platform through IPI on Red Hat Virtualization consists of the following steps:

1. [Create storage network VLAN](#)
2. [Download OpenShift installation files](#)
3. [Download CA cert from RHV](#)
4. [Register API/Apps in DNS](#)
5. [Generate and add SSH private key](#)
6. [Install OpenShift Container Platform](#)
7. [Access console/web console](#)
8. [Configure worker nodes to run storage services](#)
9. [Download and install Trident through Operator](#)

[Next: Validation Results: NetApp HCI for Red Hat OpenShift on RHV](#)

### 1. Create Storage Network VLAN: NetApp HCI for Red Hat OpenShift on RHV

To create a storage network VLAN, complete the following steps:

To support Element storage access for NetApp Trident to attach persistent volumes to pods deployed in OpenShift, the machine network being used for each worker in the OCP deployment must be able to reach the storage resources. If the machine network cannot access the Element storage network by default, an additional network/VLAN can be created in the Element cluster to allow access:

1. Using any browser, log in to the Element Cluster at the cluster's MVIP.
2. Navigate to Cluster > Network and click Create VLAN.
3. Before you provide the details, reserve at least five IP addresses from the network that is reachable from the OCP network (one for the virtual network storage VIP and one for virtual network IP on each storage node).

Enter a VLAN name of your choice, enter the VLAN ID, SVIP, and netmask, select the Enable VRF option, and enter the gateway IP for the network. In the IP address blocks, enter the starting IP of the other addresses reserved for the storage nodes. In this example, the size is four because there are four storage nodes in this cluster. Click Create VLAN.

## Create a New VLAN



VLAN Name

ocp\_storage

VLAN Tag

185

SVIP

10.61.185.205

Netmask

255.255.255.0

☒ Enable VRF

Gateway

10.61.185.1

Description

4

IP Address Blocks

Starting IP 10.61.185.201

Size 4

Add A Block

Create VLAN

Cancel

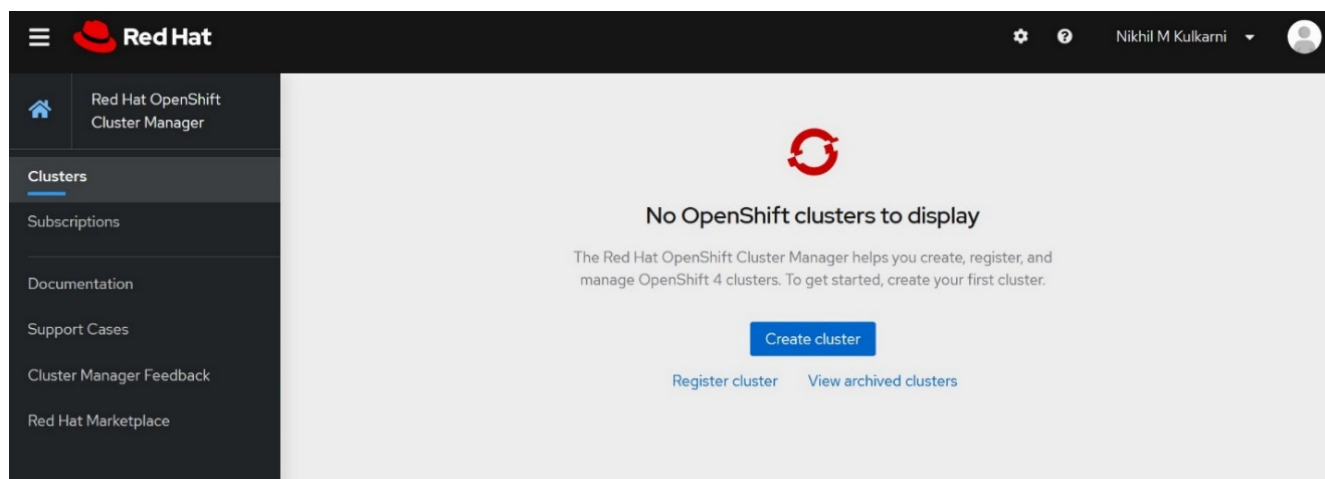
Next: [2. Download OpenShift Installation Files](#)

## 2. Download OpenShift Installation Files: NetApp HCI for Red Hat OpenShift on RHV

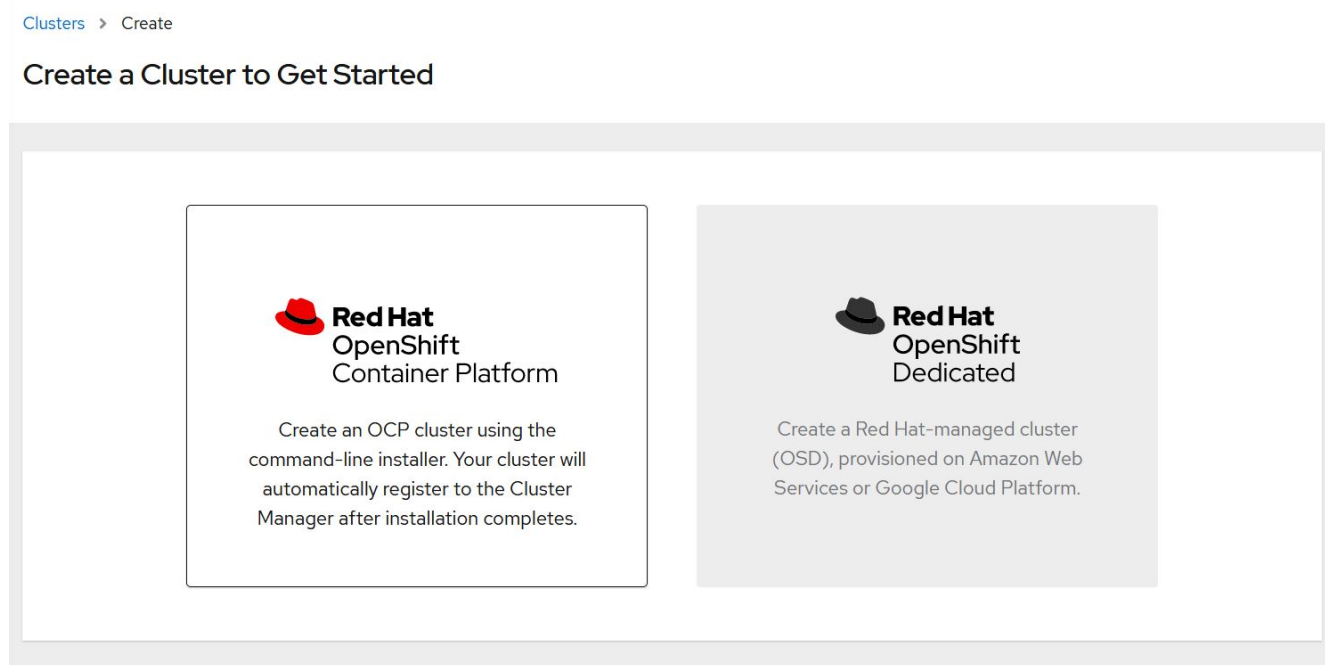
To download the OpenShift installation files, complete the following steps:

1. Go to the [Red Hat login page](#) and log in with your Red Hat credentials.

2. On the Clusters page, click Create Cluster.













3. Select OpenShift Container Platform.



4. Select Run on Red Hat Virtualization.

## Install OpenShift Container Platform 4

Select an infrastructure provider

 Run on Amazon Web Services	 Run on Microsoft Azure	 Run on Google Cloud Platform	 Run on VMware vSphere
 Run on Red Hat OpenStack	 Run on Red Hat Virtualization	 Run on Bare Metal	 Run on IBM Z
 Run on Power	 Run on Laptop <small>Powered by Red Hat CodeReady Containers</small>		

5. The next page allows you to download the OpenShift installer (available for Linux and MacOS), a unique pull secret that is required to create the `install-config` file and the `oc` command-line tools (available for Linux, Windows, and MacOS).


Download the files, transfer them to a RHEL administrative workstation from where you can run the OpenShift installation, or download these files directly using `wget` or `curl` on a RHEL administrative workstation.

**Downloads**

**OpenShift installer**  
Download and extract the install program for your operating system and place the file in the directory where you will store the installation configuration files. Note: The OpenShift install program is only available for Linux and macOS at this time.

Linux

**Pull secret**  
Download or copy your pull secret. The install program will prompt you for your pull secret during installation.

 [Copy pull secret](#)

**Command-line interface**  
Download the OpenShift command-line tools and add them to your `PATH`.

Linux

When the installer is complete you will see the console URL and credentials for accessing your new cluster. A `kubeconfig` file will also be generated for you to use with the `oc` CLI tools you downloaded.

Next: 3. Download CA Certificate from RHV

### 3. Download CA Certificate from RHV: NetApp HCI for Red Hat OpenShift on RHV

To download the CA certificate from RHV, complete the following steps:

1. In order to access the RHV manager from the RHEL machine during the deployment process, the CA certificate trust must be updated on the machine to trust connections to RHV-M. To download the RHV Manager's CA certificate, run the following commands:

```
sudo curl -k 'https://<engine-fqdn>/ovirt-engine/services/pki-resource?resource=ca-certificate&format=X509-PEM-CA' -o /tmp/ca.pem
[user@rhel7 ~]$ sudo curl -k 'https://rhv-m.cie.netapp.com/ovirt-engine/services/pki-resource?resource=ca-certificate&format=X509-PEM-CA' -o /tmp/ca.pem
```

% Total	% Received	% Xferd	Average Speed	Time	Time	Time	
Current			Dload	Upload	Total	Spent	Left
Speed							
100 1376	100 1376	0 0	9685	0	--:--:--	--:--:--	--:--:--
9690							

2. Copy the CA certificate to the directory for server certificates and update the CA trust.

```
[user@rhel7 ~]$ sudo cp /tmp/ca.pem /etc/pki/ca-trust/source/anchors/ca.pem
[user@rhel7 ~]$ sudo update-ca-trust
```

Next: [4. Register API/Apps in DNS](#)

#### 4. Register API/Apps in DNS: NetApp HCI for Red Hat OpenShift on RHV

To register API/Apps in DNS, complete the following steps:

1. Reserve three static IP addresses from the network being used for OCP: the first IP address for OpenShift Container Platform REST API, the second IP address for pointing to the wildcard application ingress, and the third IP address for the internal DNS service. The first two IPs require an entry in the DNS server.



The default value of the `machineNetwork` subnet as created by IPI during OpenShift install is `10.0.0.0/16`. If the IPs you intend to use for your cluster's management network fall outside of this range, you might need to customize your deployment and edit these values before deploying the cluster. For more information, see the section [Use a Custom Install File for OpenShift Deployment](#).

2. Configure the API domain name by using the format `api.<openshift-cluster-name>.<base-domain>` pointing to the reserved IP.

New Host

Name (uses parent domain name if blank):  
api.rhv-ocp-cluster

Fully qualified domain name (FQDN):  
api.rhv-ocp-cluster.cie.netapp.com.

IP address:  
10.63.172.151

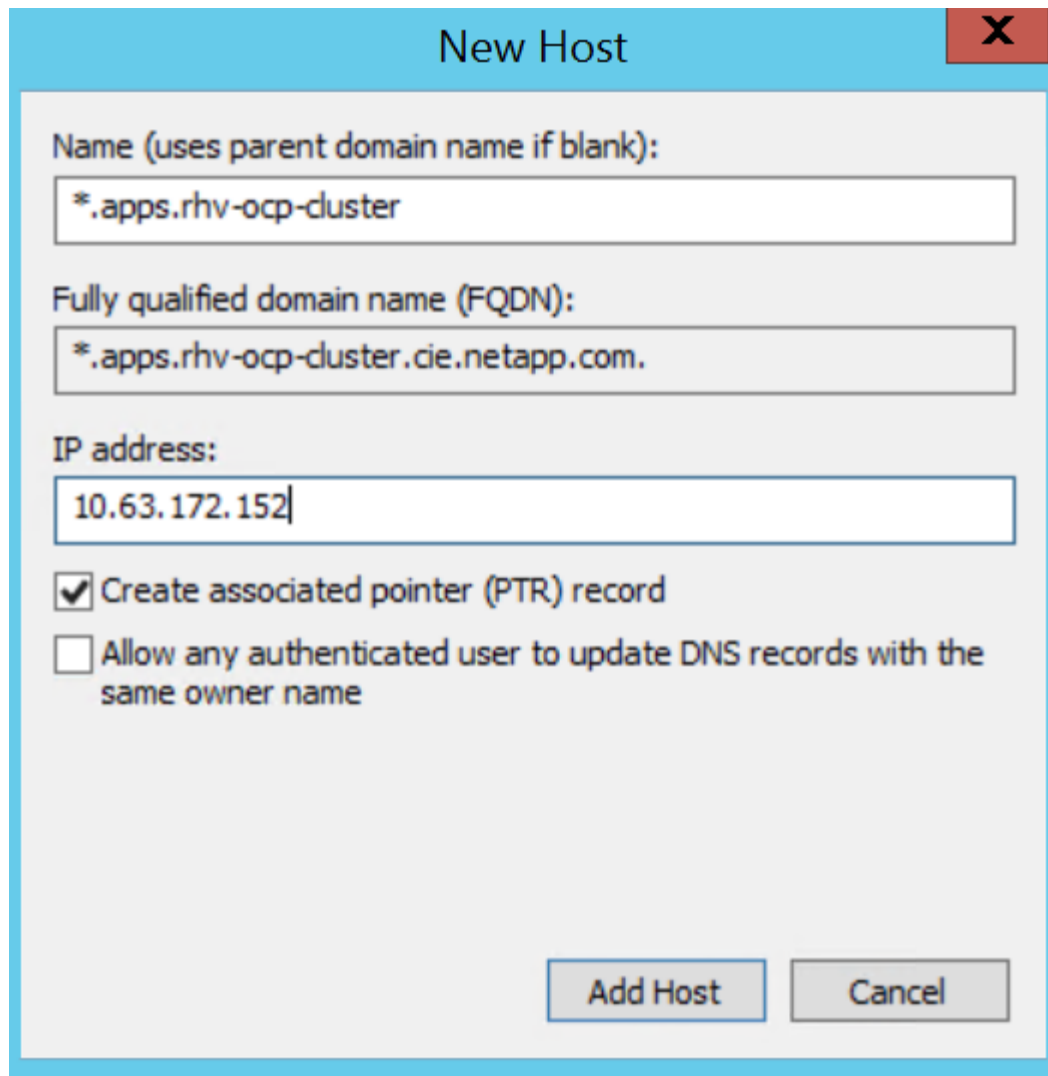
☒ Create associated pointer (PTR) record

☐ Allow any authenticated user to update DNS records with the same owner name

Add Host

Cancel

3. Configure the wildcard application ingress domain name by using the format `*.apps.<openshift-cluster-name>.<base-domain>` pointing to the reserved IP.

A screenshot of a 'New Host' dialog box. The dialog has a light blue title bar with the text 'New Host' and a red close button with a white 'X'. The main area is light gray and contains three text input fields. The first field is labeled 'Name (uses parent domain name if blank):' and contains the text '\*.apps.rhv-ocp-cluster'. The second field is labeled 'Fully qualified domain name (FQDN):' and contains the text '\*.apps.rhv-ocp-cluster.cie.netapp.com.'. The third field is labeled 'IP address:' and contains the text '10.63.172.152'. Below the fields are two checkboxes: the first is checked and labeled 'Create associated pointer (PTR) record', and the second is unchecked and labeled 'Allow any authenticated user to update DNS records with the same owner name'. At the bottom right are two buttons: 'Add Host' and 'Cancel'.

Next: [5. Generate and Add SSH Private Key](#)

## 5. Generate and Add SSH Private Key: NetApp HCI for Red Hat OpenShift on RHV

To generate and add an SSH private key, complete the following steps:

1. For the installation debugging or disaster recovery on the OpenShift cluster, you must provide an SSH key to both the `ssh-agent` and the installation program. Create an SSH key if one does not already exist for password-less authentication on the RHEL machine.

```
[user@rhel7 ~]$ ssh-keygen -t rsa -b 4096 -N '' -f ~/.ssh/id_rsa
```

2. Start the `ssh-agent` process and configure it as a background running task.

```
[user@rhel7 ~]$ eval "$(ssh-agent -s)"  
Agent pid 31874
```

3. Add the SSH private key that you created in step 2 to the `ssh-agent`, which enables you to SSH directly to the nodes without having to interactively pass the key.

```
[user@rhel7 ~]$ ssh-add ~/.ssh/id_rsa
```

Next: 6. Install OpenShift Container Platform

## 6. Install OpenShift Container Platform: NetApp HCI for Red Hat OpenShift on RHV

To install OpenShift Container Platform, complete the following steps:

1. Create a directory for OpenShift installation and transfer the downloaded files to it. Extract the OpenShift installer files from the tar archive.

```
[user@rhel7 ~]$ mkdir openshift-deploy
[user@rhel7 ~]$ cd openshift-deploy
[user@rhel7 openshift-deploy]$ tar xvf openshift-install-linux.tar.gz
README.md
openshift-install
[user@rhel7 openshift-deploy]$ ls -la
total 453260
drwxr-xr-x.  2 user user      146 May 26 16:01 .
dr-xr-x---. 16 user user    4096 May 26 15:58 ..
-rw-r--r--.  1 user user 25249648 May 26 15:59 openshift-client-
linux.tar.gz
-rwxr-xr-x.  1 user user 354664448 Apr 27 01:37 openshift-install
-rw-r--r--.  1 user user  84207215 May 26 16:00 openshift-install-
linux.tar.gz
-rw-r--r--.  1 user user    2736 May 26 15:59 pull-secret.txt
-rw-r--r--.  1 user user    706 Apr 27 01:37 README.md
```



The installation program creates several files in the directory used for installation of the cluster. Both the installation program and the files created by the installation program must be kept even after the cluster is up.



The binary files that you previously downloaded, such as `openshift-install` or `oc`, can be copied to a directory that is in the user's path (for example, `/usr/local/bin`) to make them easier to run.

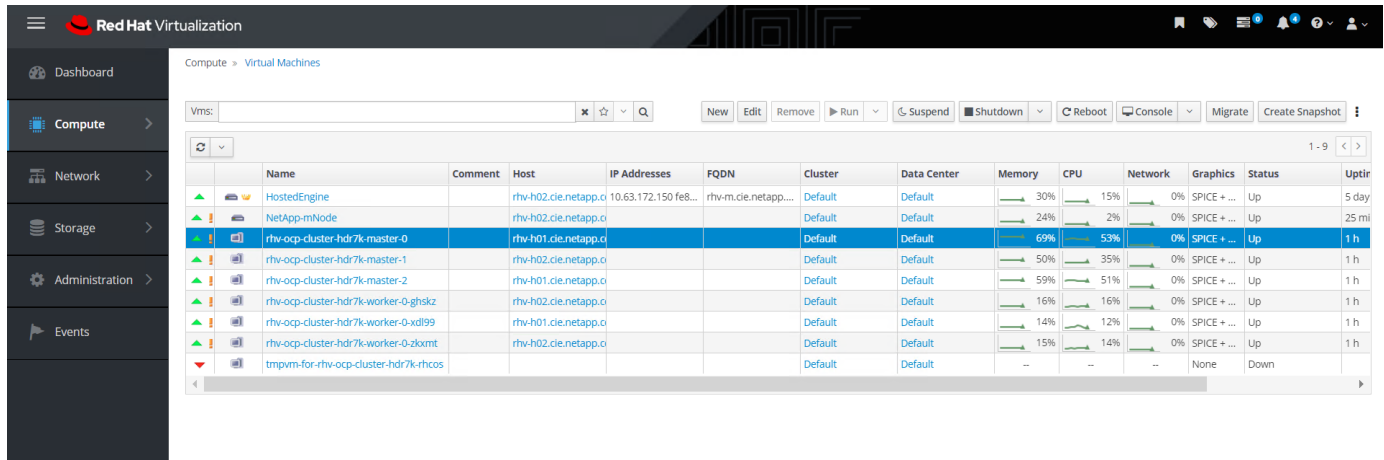
2. Create the cluster by running the `openshift-install create cluster` command and respond to the installation program prompts. Pass the SSH public key, select ovirt from the platform, provide the RHV infrastructure details, provide the three reserved IP addresses and the downloaded pull secret to the installation program prompts. After all the inputs are provided, the installation program creates and configures a bootstrap machine with a temporary Kubernetes control plane which then creates and configures the master VMs with the production Kubernetes control plane. The control plane on the master nodes creates and configures the worker VMs.



It can take approximately 30–45 minutes to get the complete cluster up and running.

```
[user@rhel7 openshift-deploy]$ ./openshift-install create cluster
--dir=/home/user/openshift-deploy --log-level=info
SSH Public Key /home/user/.ssh/id_rsa.pub
? Platform ovirt
? oVirt cluster Default
? oVirt storage domain data_domain
? oVirt network ovirtmgmt
? Internal API virtual IP 10.63. 172.151
? Internal DNS virtual IP 10.63. 172.153
? Ingress virtual IP 10.63. 172.152
? Base Domain cie.netapp.com
? Cluster Name rhv-ocp-cluster
? Pull Secret [? for help]
*****
*****
*****
*****
*****
INFO Obtaining RHCOS image file from 'https://releases-art-
rhcos.svc.ci.openshift.org/art/storage/releases/rhcos-
4.4/44.81.202004250133-0/x86_64/rhcos-44.81.202004250133-0-
openstack.x86_64.qcow2.gz?sha256=f8a44e0ea8cc45882dc22eb632a63afb90b4148
39b8aa92f3836ede001dfe9cf'
INFO The file was found in cache: /home/user/.cache/openshift-
installer/image_cache/e263efbc53c0caf612bcfaad10e3dff0. Reusing...
INFO Creating infrastructure resources...
INFO Waiting up to 20m0s for the Kubernetes API at https://api.rhv-ocp-
cluster.cie.netapp.com:6443...
INFO API v1.17.1 up
INFO Waiting up to 40m0s for bootstrapping to complete...
INFO Destroying the bootstrap resources...
INFO Waiting up to 30m0s for the cluster at https://api.rhv-ocp-
cluster.cie.netapp.com:6443 to initialize...
INFO Waiting up to 10m0s for the openshift-console route to be
created...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run
'export KUBECONFIG=/home/user/openshift-deploy/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.rhv-ocp-cluster.cie.netapp.com
INFO Login to the console with user: kubeadmin, password: NtsqU-p3qUb-
8Hscu-JfAq7
```

- When the cluster deployment is complete, the directions for accessing the OpenShift cluster, including a link to its web console and credentials for the kubeadmin user, are displayed. Make sure to take a note of these details.
- Log in to the RHV Manager and observe that the VMs relating to the OCP cluster are up and running.



Next: 7. Access Console/Web Console

## 7. Access Console/Web Console: NetApp HCI for Red Hat OpenShift on RHV

To access the console or web console, complete the following steps:

- To access the OCP cluster through the CLI, extract the `oc` command-line tools tar file and place its content in a directory that is in the user's path.

```
[user@rhel7 openshift-deploy]$ tar xvf openshift-client-linux.tar.gz
README.md
oc
kubectl
[user@rhel7 openshift-deploy]$ echo $PATH
/usr/local/bin: /usr/local/sbin:/sbin:/bin:/usr/sbin:/usr/bin
[user@rhel7 openshift-deploy]$ cp oc /usr/local/bin
```

- To interact with the cluster through the CLI, you can use the `kubeconfig` file provided by the IPI process located in the `/auth` directory inside the folder from where you launched the installation program. To easily interact with the cluster, export the file that is created in the directory. After a successful cluster deployment, the file location and the following command are displayed.

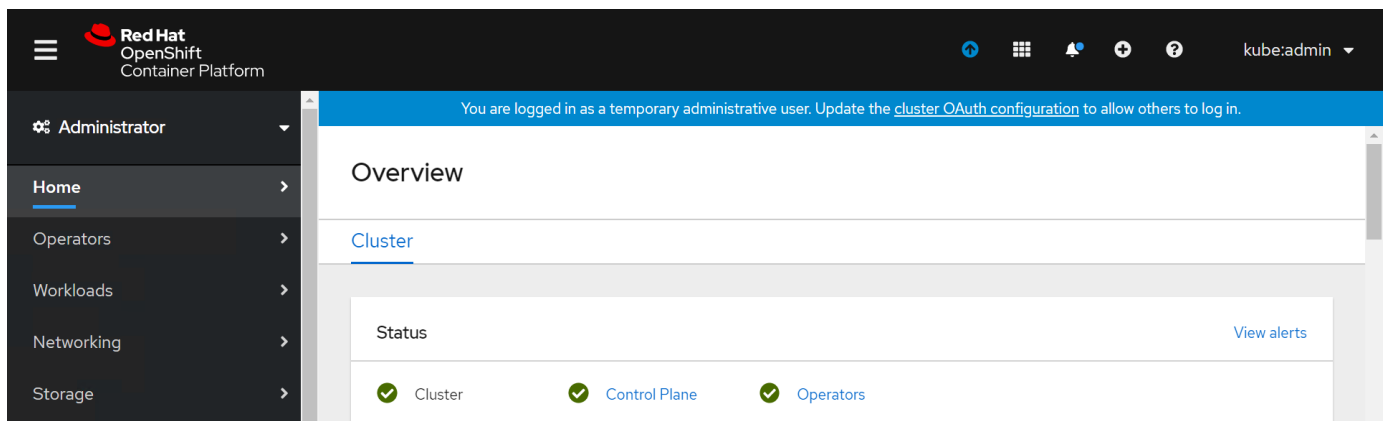
```
[user@rhel7 openshift-deploy]$ export KUBECONFIG=/home/user/openshift-deploy/auth/kubeconfig
```

- Verify whether you have access to the cluster and whether the nodes are in the Ready state.

```
[user@rhel7 openshift-deploy]$ oc get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
rhv-ocp-cluster-hdr7k-master-0	Ready	master	93m	v1.17.1
rhv-ocp-cluster-hdr7k-master-1	Ready	master	93m	v1.17.1
rhv-ocp-cluster-hdr7k-master-2	Ready	master	93m	v1.17.1
rhv-ocp-cluster-hdr7k-worker-0-ghskz	Ready	worker	83m	v1.17.1
rhv-ocp-cluster-hdr7k-worker-0-xdl99	Ready	worker	86m	v1.17.1
rhv-ocp-cluster-hdr7k-worker-0-zkxmt	Ready	worker	85m	v1.17.1

- Log in to the web console URL by using the credentials, both of which were provided after the successful deployment of the cluster, and then verify GUI access to the cluster.



Next: [8. Configure Worker Nodes to Run Storage Services](#)

## 8. Configure Worker Nodes to Run Storage Services: NetApp HCI for Red Hat OpenShift on RHV

To configure the worker nodes to run storage services, complete the following steps:

- To access storage from the Element system, each of the worker nodes must have iSCSI available and running as a service. To create a machine configuration that can enable and start the `iscsid` service, log in to the OCP web console and navigate to `Compute > Machine Configs` and click `Create Machine Config`. Paste the YAML file and click `Create`.

# Create Machine Config

Create by manually entering YAML or JSON definitions, or by dragging and dropping a file into the editor.

[? View shortcuts](#)

```
1  apiVersion: machineconfiguration.openshift.io/v1
2  kind: MachineConfig
3  metadata:
4    labels:
5      machineconfiguration.openshift.io/role: worker
6    name: worker-iscsi-configuration
7  spec:
8    config:
9      ignition:
10       version: 2.2.0
11      systemd:
12        units:
13          - name: iscsid.service
14            enabled: true
15            state: started
16  osImageURL: ""
```

Create

Cancel

 Download

2. After the configuration is created, it will take approximately 20–30 minutes to apply the configuration to the worker nodes and reload them. Verify whether the machine config is applied by using `oc get mcp` and make sure that the machine config pool for workers is updated. You can also log in to the worker nodes to confirm that the iscsid service is running.

```
[user@rhel7 openshift-deploy]$ oc get mcp
NAME          CONFIG                                UPDATED    UPDATING
DEGRADED
master        rendered-master-a520ae930e1d135e0dee7168    True       False
False
worker        rendered-worker-de321b36eeba62df41feb7bc    True       False
False
[user@rhel7 openshift-deploy]$ ssh core@10.63.172.22 sudo systemctl
status iscsid
• iscsid.service - Open-iSCSI
   Loaded: loaded (/usr/lib/systemd/system/iscsid.service; enabled;
   vendor preset: disabled)
   Active: active (running) since Tue 2020-05-26 13:36:22 UTC; 3 min ago
     Docs: man:iscsid(8)
           man:iscsiadm(8)
   Main PID: 1242 (iscsid)
   Status: "Ready to process requests"
     Tasks: 1
   Memory: 4.9M
     CPU: 9ms
   CGroup: /system.slice/iscsid.service
           └─1242 /usr/sbin/iscsid -f
```



It is also possible to confirm that the MachineConfig has been successfully applied and services have been started as expected by running the `oc debug` command with the appropriate flags.

[Next: 9. Download and Install NetApp Trident](#)

## 9. Download and Install NetApp Trident: NetApp HCI for Red Hat OpenShift on RHV

To download and install NetApp Trident, complete the following steps:

1. Make sure that the user that is logged in to the OCP cluster has sufficient privileges for installing Trident.

```
[user@rhel7 openshift-deploy]$ oc auth can-i '*' '*' --all-namespaces
yes
```

2. Verify that you can download an image from the registry and access the MVIP of the NetApp Element cluster.

```
[user@rhel7 openshift-deploy]$ oc run -i --tty ping --image=busybox
--restart=Never --rm -- ping 10.63.172.140
If you don't see a command prompt, try pressing enter.
64 bytes from 10.63.172.140: seq=1 ttl=63 time=0.312 ms
64 bytes from 10.63.172.140: seq=2 ttl=63 time=0.271 ms
64 bytes from 10.63.172.140: seq=3 ttl=63 time=0.254 ms
64 bytes from 10.63.172.140: seq=4 ttl=63 time=0.309 ms
64 bytes from 10.63.172.140: seq=5 ttl=63 time=0.319 ms
64 bytes from 10.63.172.140: seq=6 ttl=63 time=0.303 ms
^C
--- 10.63.172.140 ping statistics ---
7 packets transmitted, 7 packets received, 0% packet loss
round-trip min/avg/max = 0.254/0.387/0.946 ms
pod "ping" deleted
```

3. Download the Trident installer bundle using the following commands and extract it to a directory.

```
[user@rhel7 ~]$ wget
[user@rhel7 ~]$ tar -xvf trident-installer-20.04.0.tar.gz
[user@rhel7 ~]$ cd trident-installer
```

4. The Trident installer contains manifests for defining all the required resources. Using the appropriate manifests, create the TridentProvisioner custom resource definition.

```
[user@rhel7 trident-installer]$ oc create -f
deploy/crds/trident.netapp.io_tridentprovisioners_crd_post1.16.yaml

customresourcedefinition.apiextensions.k8s.io/tridentprovisioners.trident.netapp.io created
```

5. Create a Trident namespace, which is required for the Trident operator.

```
[user@rhel7 trident-installer]$ oc create namespace trident
namespace/trident created
```

6. Create the resources required for the Trident operator deployment, such as a ServiceAccount for the operator, a ClusterRole and ClusterRoleBinding to the ServiceAccount, a dedicated PodSecurityPolicy, or the operator itself.

```
[user@rhel7 trident-installer]$ oc kustomize deploy/ >
deploy/bundle.yaml
[user@rhel7 trident-installer]$ oc create -f deploy/bundle.yaml
serviceaccount/trident-operator created
clusterrole.rbac.authorization.k8s.io/trident-operator created
clusterrolebinding.rbac.authorization.k8s.io/trident-operator created
deployment.apps/trident-operator created
podsecuritypolicy.policy/tridentoperatorpods created
```

7. Verify that the Trident operator is deployed.

```
[user@rhel7 trident-installer]$ oc get deployment -n trident
NAME                READY    UP-TO-DATE    AVAILABLE    AGE
trident-operator    1/1      1              1            56s
[user@rhel7 trident-installer]$ oc get pods -n trident
NAME                                READY    STATUS    RESTARTS    AGE
trident-operator-564d7d66f-qrz7v    1/1      Running    0           71s
```

8. After the Trident operator is installed, install Trident using this operator. In this example, TridentProvisioner custom resource (CR) was created. The Trident installer comes with definitions for creating a TridentProvisioner CR. These can be modified based on the requirements.

```
[user@rhel7 trident-installer]$ oc create -f
deploy/crds/tridentprovisioner_cr.yaml
tridentprovisioner.trident.netapp.io/trident created
```

9. Approve the Trident serving CSR certificates by using `oc get csr -o name | xargs oc adm certificate approve`.

```
[user@rhel7 trident-installer]$ oc get csr -o name | xargs oc adm
certificate approve
certificatesigningrequest.certificates.k8s.io/csr-4b7zh approved
certificatesigningrequest.certificates.k8s.io/csr-4hkwk approved
certificatesigningrequest.certificates.k8s.io/csr-5bgh5 approved
certificatesigningrequest.certificates.k8s.io/csr-5g4d6 approved
certificatesigningrequest.certificates.k8s.io/csr-5j9hz approved
certificatesigningrequest.certificates.k8s.io/csr-5m8qb approved
certificatesigningrequest.certificates.k8s.io/csr-66hv2 approved
certificatesigningrequest.certificates.k8s.io/csr-6rdgg approved
certificatesigningrequest.certificates.k8s.io/csr-6t24f approved
certificatesigningrequest.certificates.k8s.io/csr-76wgv approved
certificatesigningrequest.certificates.k8s.io/csr-78qsq approved
certificatesigningrequest.certificates.k8s.io/csr-7r58n approved
certificatesigningrequest.certificates.k8s.io/csr-8ghmk approved
certificatesigningrequest.certificates.k8s.io/csr-8sn5q approved
```

10. Verify that Trident 20.04 is installed by using the TridentProvisioner CR, and verify that the pods related to Trident are.

```
[user@rhel7 trident-installer]$ oc get tprov -n trident
NAME          AGE
trident        9m49s

[user@rhel7 trident-installer]$ oc describe tprov trident -n trident
Name:          trident
Namespace:     trident
Labels:        <none>
Annotations:   <none>
API Version:   trident.netapp.io/v1
Kind:          TridentProvisioner
Metadata:
  Creation Timestamp:  2020-05-26T18:49:19Z
  Generation:         1
  Resource Version:    640347
  Self Link:
/apis/trident.netapp.io/v1/namespaces/trident/tridentprovisioners/trident
  UID:               52656806-0414-4ed8-b355-fc123fafbf4e
Spec:
  Debug:  true
Status:
  Message:  Trident installed
  Status:   Installed
  Version:  v20.04
```



Events:

Type	Reason	Age	From
Message			
----	-----	----	----
-----			

Normal	Installing	9m32s	trident-operator.netapp.io
Installing Trident			

Normal	Installed	3m47s (x5 over 8m56s)	trident-operator.netapp.io
Trident installed			

```
[user@rhel7 trident-installer]$ oc get pods -n trident
```

NAME	READY	STATUS	RESTARTS	AGE
trident-csi-7f769c7875-s6fmt	5/5	Running	0	10m
trident-csi-cp7wg	2/2	Running	0	10m
trident-csi-hhx94	2/2	Running	0	10m
trident-csi-l72bt	2/2	Running	0	10m
trident-csi-xfl9d	2/2	Running	0	10m
trident-csi-xrhqx	2/2	Running	0	10m
trident-csi-zb7ws	2/2	Running	0	10m
trident-operator-564d7d66f-qrz7v	1/1	Running	0	27m

```
[user@rhel7 trident-installer]$ ./tridentctl -n trident version
```

```
+-----+
| SERVER VERSION | CLIENT VERSION |
+-----+
| 20.04.0        | 20.04.0        |
+-----+
```

11. Create a storage backend that will be used by Trident to provision volumes. The storage backend specifies the Element cluster in NetApp HCI. You also can specify sample bronze, silver, and gold types with corresponding QoS specs.

```
[user@rhel7 trident-installer]$ vi backend.json
{
    "version": 1,
    "storageDriverName": "solidfire-san",
    "Endpoint": "https://admin: admin- password@10.63.172.140/json-
rpc/8.0",
    "SVIP": "10.61.185.205:3260",
    "TenantName": "trident",
    "Types": [{"Type": "Bronze", "Qos": {"minIOPS": 1000, "maxIOPS":
2000, "burstIOPS": 4000}},
                {"Type": "Silver", "Qos": {"minIOPS": 4000, "maxIOPS":
6000, "burstIOPS": 8000}},
                {"Type": "Gold", "Qos": {"minIOPS": 6000, "maxIOPS": 8000,
"burstIOPS": 10000}}]
}
[user@rhel7 trident-installer]$ ./tridentctl -n trident create backend
-f backend.json
+-----+-----+
+-----+-----+-----+
|          NAME          | STORAGE DRIVER |          UUID          |
| STATE | VOLUMES | |
+-----+-----+-----+
+-----+-----+-----+
| solidfire_10.61.185.205 | solidfire-san  | 40f48d99-5d2e-4f6c-89ab-
8aee2be71255 | online |      0 |
+-----+-----+-----+
+-----+-----+-----+
```

Modify the `backend.json` to accommodate the details or requirements of your environment for the following values:

- Endpoint corresponds to the credentials and the MVIP of the NetApp HCI Element cluster.
- SVIP corresponds to the SVIP configured over the VM network in the section titled [Create Storage Network VLAN](#).
- Types corresponds to different QoS bands. New persistent volumes can be created with specific QoS settings by specifying the exact storage pool.

12. Create a StorageClass that specifies Trident as the provisioner and the storage backend as `solidfire-san`.

```
[user@rhel7 trident-installer]$ vi storage-class-basic.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: basic-csi
  annotations:
    storageclass.kubernetes.io/is-default-class: "true"
provisioner: csi.trident.netapp.io
parameters:
  backendType: "solidfire-san"
  provisioningType: "thin"

[user@rhel7 trident-installer]$ oc create -f storage-class-basic.yaml
storageclass.storage.k8s.io/basic created
```



In this example, the StorageClass created is set as a default, however an OpenShift administrator can define multiple storage classes corresponding to different QoS requirements and other factors based upon their applications. Trident selects a storage backend that can satisfy all the criteria specified in the parameters section in the storage class definition. End users can then provision storage as needed, without administrative intervention.

[Next: Validation Results: NetApp HCI for Red Hat OpenShift on RHV](#)

## Validation Results: NetApp HCI for Red Hat OpenShift on RHV

This section provides the steps to deploy a continuous integration/continuous delivery or deployment (CI/CD) pipeline with Jenkins in order to validate the operation of the solution.

### Create the Resources Required for Jenkins Deployment

To create the resources required for deploying the Jenkins application, complete the following steps:

1. Create a new project named Jenkins.

# Create Project

Name \*

Display Name

Description

Cancel

Create

2. In this example, we deployed Jenkins with persistent storage. To support the Jenkins build, create the PVC. Navigate to Storage > Persistent Volume Claims and click Create Persistent Volume Claim. Select the storage class that was created, make sure that the Persistent Volume Claim Name is jenkins, select the appropriate size and access mode, and then click Create.

## Create Persistent Volume Claim

[Edit YAML](#)

### Storage Class

SC basic ▼

Storage class for the new claim.

### Persistent Volume Claim Name \*

jenkins

A unique name for the storage claim within the project.

### Access Mode \*

☒ Single User (RWO) ☐ Shared Access (RWX) ☐ Read Only (ROX)

Permissions to the mounted drive.

### Size \*

100 GiB ▼

Desired storage capacity.

☐ Use label selectors to request storage

Use label selectors to define how storage is created.

Create

Cancel

### Deploy Jenkins with Persistent Storage

To deploy Jenkins with persistent storage, complete the following steps:

1. In the upper left corner, change the role from Administrator to Developer. Click +Add and select From Catalog. In the Filter by Keyword bar, search jenkins. Select Jenkins Service, with Persistent Storage.

## Developer Catalog

Add shared apps, services, or source-to-image builders to your project from the Developer Catalog. Cluster admins can install additional apps which will show up here automatically.

All Items

Languages

Databases

Middleware

CI/CD

Other

Type

☒ Operator Backed (0)

☐ Helm Charts (0)

☒ Builder Image (0)


☒ Template (4)

☐ Service Class (0)

All Items

jenkins


Group By: None ▾

Template

Jenkins

provided by Red Hat, Inc.


Jenkins service, with persistent storage. NOTE: You must have persistent volumes available in...

Template

Jenkins

provided by Red Hat, Inc.


Jenkins service, with persistent storage. NOTE: You must have persistent volumes available in...

Template

Jenkins (Ephemeral)

provided by Red Hat, Inc.

Jenkins service, without persistent storage. WARNING: Any data stored will be lost upon...

Template

Jenkins (Ephemeral)

provided by Red Hat, Inc.

Jenkins service, without persistent storage. WARNING:

2. Click Instantiate Template.



### Jenkins

Provided by Red Hat, Inc.



Instantiate Template

#### Provider

Red Hat, Inc.

#### Support

[Get support](#)

#### Created At

 May 26, 3:58 am

#### Description

Jenkins service, with persistent storage.

NOTE: You must have persistent volumes available in your cluster to use this template.

#### Documentation

[https://docs.okd.io/latest/using\\_images/other\\_images/jenkins.html](https://docs.okd.io/latest/using_images/other_images/jenkins.html)

3. By default, the details for the Jenkins application are populated. Based on your requirements, modify the parameters, and click Create. This process creates all the required resources for supporting Jenkins on

## Instantiate Template

**Namespace \***  

PR jenkins

**Jenkins Service Name**  

jenkins

The name of the OpenShift Service exposed for the Jenkins container.

**Jenkins JNLP Service Name**  

jenkins-jnlp

The name of the service used for master/slave communication.

**Enable OAuth in Jenkins**  

true

Whether to enable OAuth OpenShift integration. If false, the static account 'admin' will be initialized with the password 'password'.

**Memory Limit**  

1Gi

Maximum amount of memory the container can use.

**Volume Capacity \***  

50Gi

Volume space available for data, e.g. 512Mi, 2Gi.

**Jenkins ImageStream Namespace**  

openshift

The OpenShift Namespace where the Jenkins ImageStream resides.

**Disable memory intensive administrative monitors**  

false

Whether to perform memory intensive, possibly slow, synchronization with the Jenkins Update Center on start. If true, the Jenkins core update monitor and site warnings monitor are disabled.

**Jenkins ImageStreamTag**  

jenkins:2

Name of the ImageStreamTag to be used for the Jenkins image.

**Fatal Error Log File**  

false

When a fatal error occurs, an error log is created with information and the state obtained at the time of the fatal error.

**Allows use of Jenkins Update Center repository with invalid SSL certificate**  

false

Whether to allow use of a Jenkins Update Center that uses invalid certificate (self-signed, unknown CA). If any value other than 'false', certificate check is bypassed. By default, certificate check is enforced.



**Jenkins**  
INSTANT-APP JENKINS  
[View documentation](#) [Get support](#)

Jenkins service, with persistent storage.

NOTE: You must have persistent volumes available in your cluster to use this template.

- The following resources will be created:
- DeploymentConfig
  - PersistentVolumeClaim
  - RoleBinding
  - Route
  - Service
  - ServiceAccount

CreateCancel

4. The Jenkins pods take approximately 10–12 minutes to enter the Ready state.

Project: jenkins ▼

## Pods

Create Pod

Filter by name...

1Running

0Pending

0Terminating

0CrashLoopBackOff

1Completed

0Failed

0Unknown

Select all filters

1 of 2 Items

Name ↑	Namespace ↕	Status ↕	Ready ↕	Owner ↕	Memory ↕	CPU ↕	
<div><div>P</div>jenkins-1-c77n9</div>	<div><div>NS</div>jenkins</div>	<div><div>🔄</div>Running</div>	1/1	<div><div>RC</div>jenkins-1</div>	-	0.004 cores	<div></div>





5. After the pods are instantiated, navigate to Networking > Routes. To open the Jenkins webpage, click the URL provided for the jenkins route.

Project: jenkins ▼

## Routes

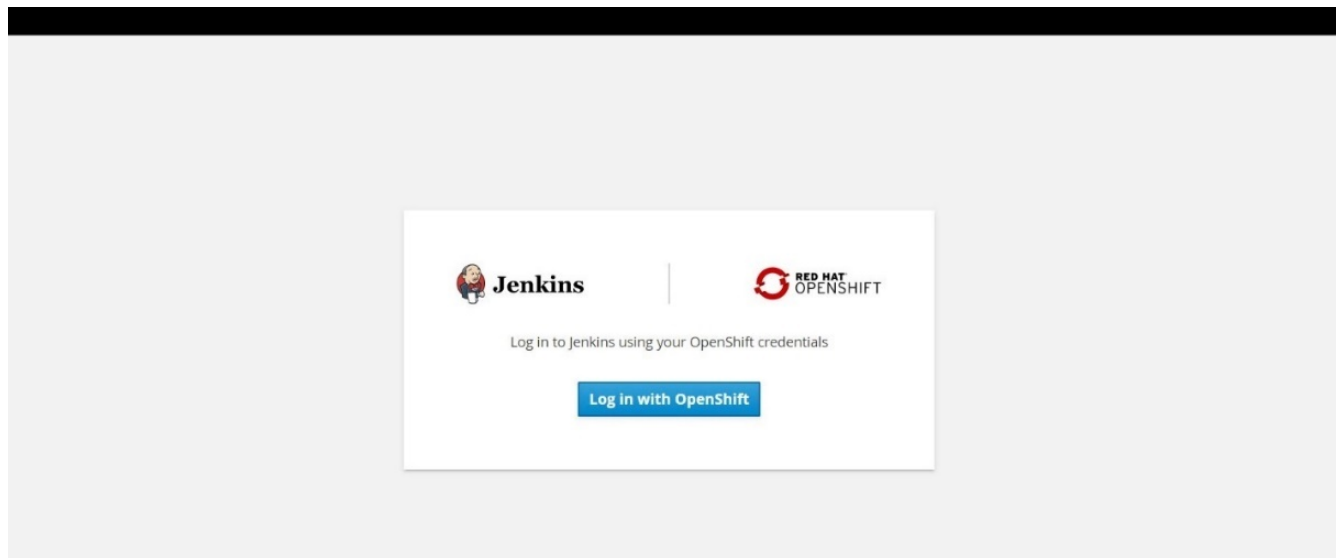
Create Route

Filter by name...

1 Accepted	0 Rejected	0 Pending	Select all filters		1 Item	
Name ↓	Namespace	Status	Location	Service		
 jenkins	 jenkins	 Accepted	<a href="https://jenkins-jenkins.apps.rhv-ocp-cluster.cie.netapp.com">https://jenkins-jenkins.apps.rhv-ocp-cluster.cie.netapp.com</a>	 jenkins	⋮	

6. Because the OpenShift OAuth was used while creating the Jenkins app, click Log in with OpenShift.





7. Authorize jenkins service-account to access the OpenShift users.

## Authorize Access

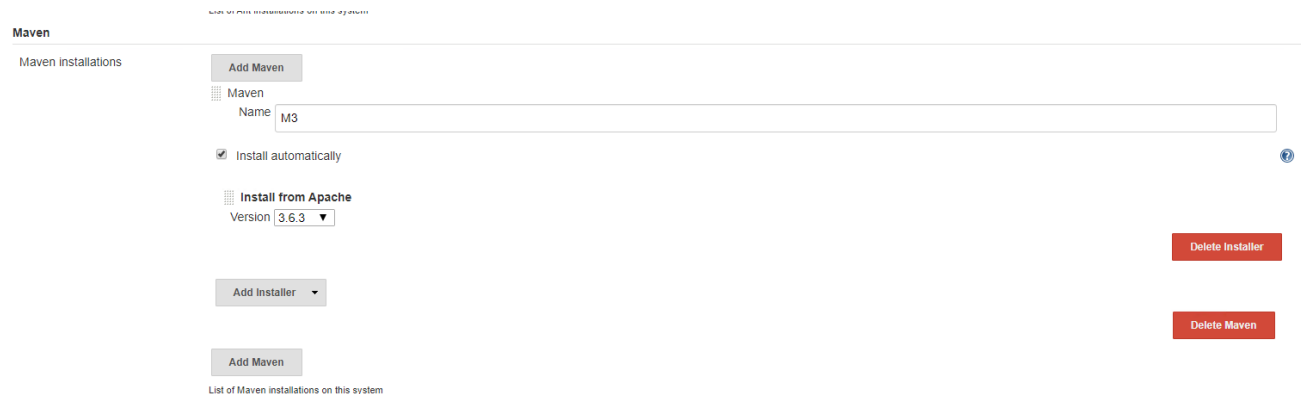
Service account `jenkins` in project `jenkins` is requesting permission to access your account (`kube:admin`)

Requested permissions

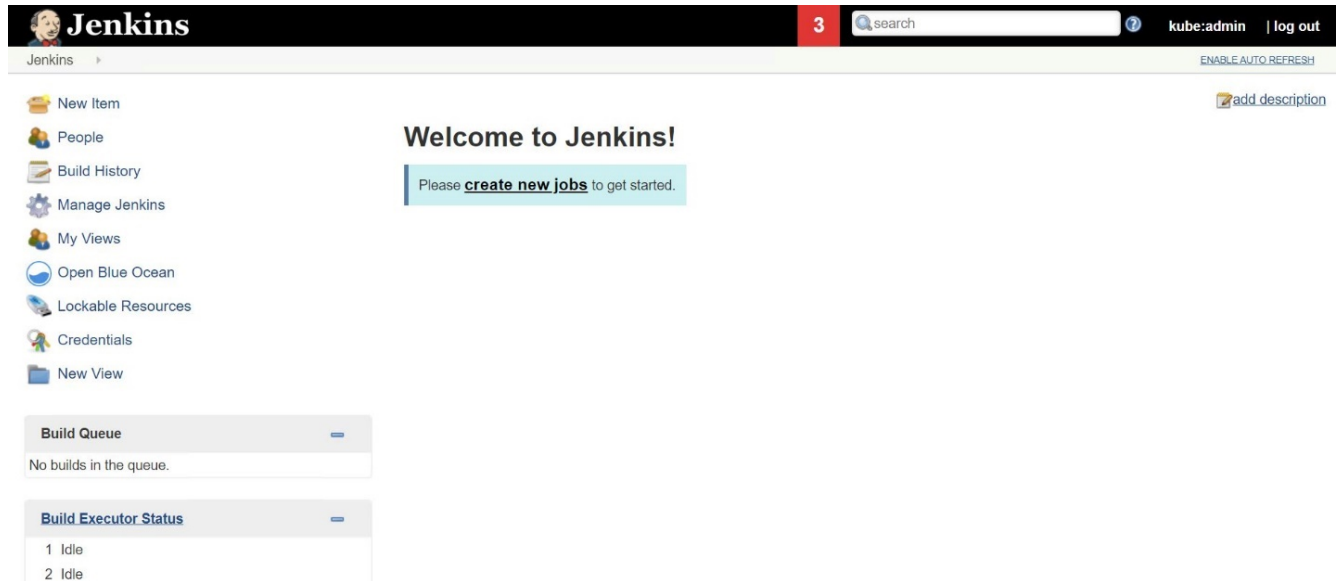
- ☒ **user:info**  
Read-only access to your user information (including username, identities, and group membership)
- ☒ **user:check-access**  
Read-only access to view your privileges (for example, "can I create builds?")

You will be redirected to <https://jenkins-jenkins.apps.rhv-ocp-cluster.cie.netapp.com/securityRealm/finishLogin>

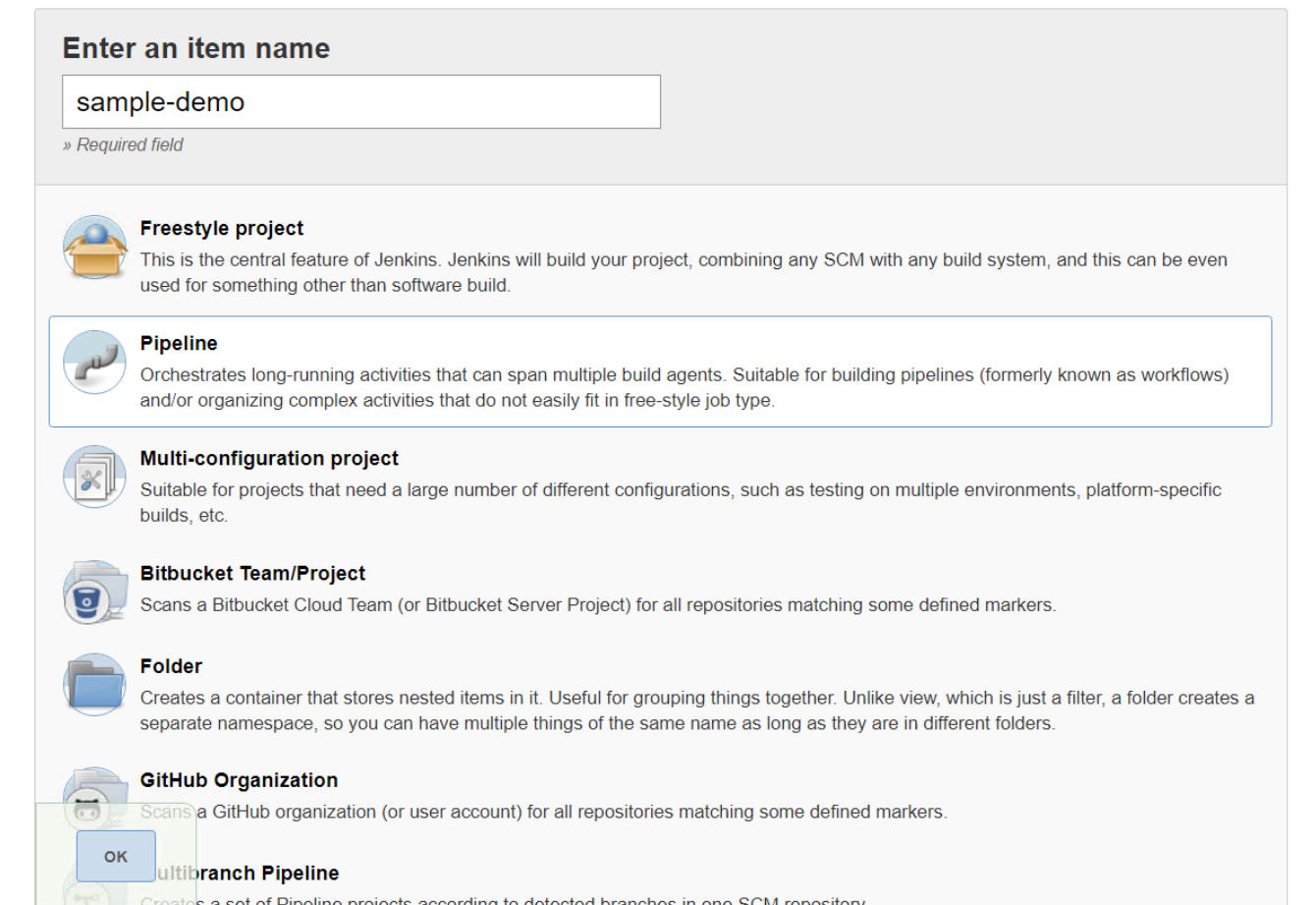
8. The Jenkins welcome page is displayed. Because we are using a Maven build, complete the Maven installation first. Navigate to Manage Jenkins > Global Tool Configuration, then in the Maven subhead, click Add Maven. Enter the name of your choice and make sure that the Install Automatically option is selected. Click Save.



9. You can now create a pipeline to demonstrate the CI/CD workflow. On the home page, click Create New Jobs or New Item from the left- hand menu.



10. On the Create Item page, enter the name of your choice, select Pipeline, and click Ok.



11. Select the Pipeline tab. From the Try Sample Pipeline drop- down menu, select Github + Maven. The code is automatically populated. Click Save.

GeneralBuild TriggersAdvanced Project OptionsPipeline

Advanced...

## Pipeline

DefinitionPipeline script

Script

1 node {  
2 def mvnHome  
3 stage('Preparation') { // for display purposes  
4 // Get some code from a GitHub repository  
5 git 'https://github.com/jglick/simple-maven-project-with-tests.git'  
6 // Get the Maven tool.  
7 // \*\* NOTE: This 'M3' Maven tool must be configured  
8 // \*\* in the global configuration.  
9 mvnHome = tool 'M3'  
10 }  
11 stage('Build') {  
12 // Run the maven build  
13 withEnv(["MVN\_HOME=\$mvnHome"]) {  
14 if (isUnix()) {  
15 sh "\$MVN\_HOME/bin/mvn" -Dmaven.test.failure.ignore clean package'  
16 } else {  
17 bat("/%MVN\_HOME%\bin\mvn" -Dmaven.test.failure.ignore clean package/)  
18 }  
19 }  
20 }

GitHub + Maven


☒ Use Groovy Sandbox

[Pipeline Syntax](#)

Save

Apply

- Click Build Now to trigger the development through the preparation, build, and testing phase. It can take several minutes to complete the whole build process and display the results of the build.

Jenkins

Jenkins > sample-demo >

Back to Dashboard

Status

Changes

Build Now

Delete Pipeline

Configure

Full Stage View

Open Blue Ocean

Rename

Pipeline Syntax

Build History

trend

find

#1

May 27, 2020 3:53 PM

Atom feed for all

Atom feed for failures

# Pipeline sample-demo

Last Successful Artifacts

simple-maven-project-with-tests-1.0-SNAPSHOT.jar

1.71 KB

view

Recent Changes

## Stage View

Average stage times:  
(Average full run time: ~7s)

#1

May 27 08:53

No Changes

Preparation	Build	Results
2s	4s	69ms
2s	4s	69ms

Latest Test Result (no failures)

## Permalinks

- [Last build \(#1\), 1 min 23 sec ago](#)
- [Last stable build \(#1\), 1 min 23 sec ago](#)
- [Last successful build \(#1\), 1 min 23 sec ago](#)
- [Last completed build \(#1\), 1 min 23 sec ago](#)

13. Whenever there are any code changes, the pipeline can be rebuilt to patch the new version of software enabling continuous integration and continuous delivery. Click Recent Changes to track the changes from the previous version.

Jenkins

Jenkins

sample-demo

Back to Dashboard

Status

Changes

Build Now

Delete Pipeline

Configure

Full Stage View

Open Blue Ocean

Rename

Pipeline Syntax

Build History

find

#2

May 27, 2020 3:56 PM

#1

May 27, 2020 3:53 PM

Atom feed for all

Atom feed for failures

Pipeline sample-demo

Last Successful Artifacts

simple-maven-project-with-tests-1.0-SNAPSHOT.jar

1.71 KB

view

Recent Changes

Stage View

Average stage times:

(Average full run time: ~6s)

#2

May 27 08:56

No Changes

#1

May 27 08:53

No Changes

Preparation	Build	Results
2s	4s	86ms
1s	4s	104ms
2s	4s	69ms

Latest Test Result

(no failures)

Permalinks

- Last build (#2), 19 sec ago
- Last stable build (#2), 19 sec ago
- Last successful build (#2), 19 sec ago
- Last completed build (#2), 19 sec ago

Next: [Best Practices for Production Deployments](#)

## Best Practices for Production Deployments - NetApp HCI for Red Hat OpenShift on RHV

This section lists several best practices that an organization should take into consideration before deploying this solution into production.

### Deploy OpenShift to an RHV Cluster of at Least Three Nodes

The verified architecture described in this document presents the minimum hardware deployment suitable for HA operations by deploying two RHV-H hypervisor nodes and ensuring a fault tolerant configuration where both hosts can manage the hosted-engine and deployed VMs can migrate between the two hypervisors. Because Red Hat OpenShift initially deploys with three master nodes, it is ensured in a two-node configuration that at least two masters will occupy the same node, which can lead to a possible outage for OpenShift if that specific node becomes unavailable. Therefore, it is a Red Hat best practice that at least three RHV-H hypervisor nodes be deployed as part of the solution so that the OpenShift masters can be distributed evenly, and the solution receives an added degree of fault tolerance.

## Configure Virtual Machine/Host Affinity

Ensuring the distribution of the OpenShift masters across multiple hypervisor nodes can be achieved by enabling VM/host affinity. Affinity is a way to define rules for a set of VMs and/or hosts that determine whether the VMs run together on the same host or hosts in the group or on different hosts. It is applied to VMs by creating affinity groups that consist of VMs and/or hosts with a set of identical parameters and conditions. Depending on whether the VMs in an affinity group run on the same host or hosts in the group or separately on different hosts, the parameters of the affinity group can define either positive affinity or negative affinity. The conditions defined for the parameters can be either hard enforcement or soft enforcement. Hard enforcement ensures that the VMs in an affinity group always follows the positive/negative affinity strictly without any regards to external conditions. Soft enforcement, on the other hand, ensures that a higher preference is set out for the VMs in an affinity group to follow the positive/negative affinity whenever feasible. In a two or three hypervisor configuration as described in this document soft affinity is the recommended setting, in larger clusters hard affinity can be relied on to ensure OpenShift nodes are distributed. To configure affinity groups, see the [Red Hat 6.11. Affinity Groups documentation](#).

## Use a Custom Install File for OpenShift Deployment

IPI makes the deployment of OpenShift clusters extremely easy through the interactive wizard discussed earlier in this document. However, it is possible that there are some default values that might need to be changed as a part of a cluster deployment. In these instances, the wizard can be run and tasked without immediately deploying a cluster, but instead outputting a configuration file from which the cluster can be deployed later. This is very useful if any IPI defaults need to be changed, or if a user wants to deploy multiple identical clusters in their environment for other uses such as multitenancy. For more information about creating a customized install configuration for OpenShift, see [Red Hat OpenShift Installing a Cluster on RHV with Customizations](#).

[Next: Videos and Demos: NetApp HCI for Red Hat OpenShift on Red Hat Virtualization](#)

## Videos and Demos: NetApp HCI for Red Hat OpenShift on RHV

The following video demonstrates some of the capabilities documented in this document:

 | *NetApp HCI for Red Hat OpenShift on Red Hat Virtualization*

[Next: Additional Information: NetApp HCI for Red Hat OpenShift on Red Hat Virtualization](#)

## Additional Information: NetApp HCI for Red Hat OpenShift on RHV

To learn more about the information described in this document, review the following websites:

- NetApp HCI Documentation <https://www.netapp.com/us/documentation/hci.aspx>
- NetApp Trident Documentation <https://netapp-trident.readthedocs.io/en/stable-v20.04/>
- Red Hat Virtualization Documentation [https://access.redhat.com/documentation/en-us/red\\_hat\\_virtualization/4.3/](https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.3/)
- Red Hat OpenShift Documentation [https://access.redhat.com/documentation/en-us/openshift\\_container\\_platform/4.4/](https://access.redhat.com/documentation/en-us/openshift_container_platform/4.4/)

## Copyright Information

Copyright © 2021 NetApp, Inc. All rights reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means-graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system-without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP "AS IS" AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

## Trademark Information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.