





Join a community dedicated to learning open source

The Red Hat® Learning Community is a collaborative platform for users to accelerate open source skill adoption while working with Red Hat products and experts.



Network with tens of thousands of community members



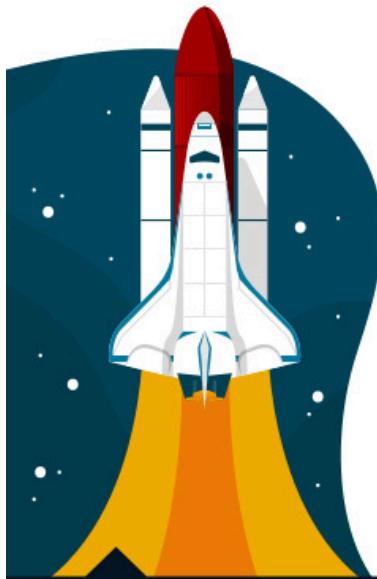
Engage in thousands of active conversations and posts



Join and interact with hundreds of certified training instructors



Unlock badges as you participate and accomplish new goals



This knowledge-sharing platform creates a space where learners can connect, ask questions, and collaborate with other open source practitioners.

Access free Red Hat training videos

Discover the latest Red Hat Training and Certification news

Connect with your instructor - and your classmates - before, after, and during your training course.

Join peers as you explore Red Hat products

Join the conversation learn.redhat.com



Copyright © 2020 Red Hat, Inc. Red Hat, Red Hat Enterprise Linux, the Red Hat logo, and Ansible are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

Red Hat OpenShift Installation Lab



OCP 4.6 DO322

Red Hat OpenShift Installation Lab

Edition 6 20230508

Publication date 20230508

Authors: Michael Jarrett, Alejandro Coma, Chris Caillouet, Benjamin Chardi
Course Architect: Fernando Lozano
DevOps Engineer: Jim Rigsbee
Editor: Nicole Muller

Copyright © 2023 Red Hat

The contents of this course and all its modules and related materials, including handouts to audience members, are
Copyright © 2023 Red Hat.

No part of this publication may be stored in a retrieval system, transmitted or reproduced in any way, including, but not limited to, photocopy, photograph, magnetic, electronic or other record, without the prior written permission of Red Hat.

This instructional program, including all material provided herein, is supplied without any guarantees from Red Hat, Inc. Red Hat, Inc. assumes no liability for damages or legal action arising from the use or misuse of contents or details contained herein.

If you believe Red Hat training materials are being used, copied, or otherwise improperly distributed, please send email to training@redhat.com [mailto:training@redhat.com] or phone toll-free (USA) +1 (866) 626-2994 or +1 (919) 754-3700.

Red Hat, Red Hat Enterprise Linux, the Red Hat logo, JBoss, OpenShift, Fedora, Hibernate, Ansible, CloudForms, RHCA, RHCE, RHCSA, Ceph, and Gluster are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

Linux™ is the registered trademark of Linus Torvalds in the United States and other countries.

Java™ is a registered trademark of Oracle and/or its affiliates.

XFS™ is a registered trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL™ is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js™ is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack™ Word Mark and OpenStack Logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. Red Hat is not affiliated with, endorsed or sponsored by the OpenStack Foundation or the OpenStack community.

All other trademarks are the property of their respective owners.

Contributors: **Eliezer Campos, Sajith Eyamkuzhy, David Sacco**

Document Conventions	ix
Admonitions	ix
Inclusive Language	x
Introduction	xi
Red Hat OpenShift Installation Lab	xi
Orientation to the Classroom Environment	xii
1. Describing the OpenShift Installation Process	1
Introducing OpenShift Installation Methods	2
Quiz: Introducing OpenShift Installation Methods	24
Running the OpenShift Installer	26
Quiz: Running the OpenShift Installer	55
Guided Exercise: Completing the OpenShift Installation Prerequisites	57
Introducing Hosted OpenShift	67
Quiz: Introducing Hosted OpenShift	70
Quiz: Chapter Review: Describing the OpenShift Installation Process	72
Summary	76
2. Installing OpenShift on a Cloud Provider	77
Introducing OpenShift Full-stack Automation Installation on a Cloud Provider	78
Quiz: Introducing OpenShift Full-stack Automation Installation on a Cloud Provider	94
Describing How to Install OpenShift on AWS Using Full-stack Automation	98
Quiz: Describing How to Install OpenShift on AWS Using Full-stack Automation	112
Demonstrating How to Install OpenShift on AWS Using Full-stack Automation	114
Quiz: Demonstrating How to Install OpenShift on AWS Using Full-stack Automation	129
Verifying the Installation of OpenShift on AWS	133
Quiz: Verifying the Installation of OpenShift on AWS	147
Quiz: Chapter Review: Installing OpenShift on a Cloud Provider	149
Summary	153
3. Installing OpenShift on a Virtualized Environment	155
Introducing OpenShift Installation on Hypervisors	156
Quiz: Introducing OpenShift Installation on Hypervisors	162
Describing the Installation of OpenShift on vSphere Using Full-stack Automation	164
Quiz: Describing the Installation of OpenShift on vSphere Using Full-stack Automation	171
Describing the Installation of OpenShift on vSphere Using Pre-existing Infrastructure	173
Quiz: Describing the Installation of OpenShift on vSphere Using Pre-existing Infrastructure	180
Quiz: Chapter Review: Installing OpenShift on a Virtualized Environment	182
Summary	186
4. Planning to Install OpenShift Without an Infrastructure Provider	187
Introducing OpenShift Installation Without an Infrastructure Provider	188
Quiz: Introducing OpenShift Installation Without an Infrastructure Provider	191
Configuring Network Services and Hosts for Installing OpenShift Without an Infrastructure Provider	193
Guided Exercise: Configuring Network Services and Hosts for Installing OpenShift Without an Infrastructure Provider	203
Quiz: Chapter Review: Planning to Install OpenShift Without an Infrastructure Provider	220
Summary	224
5. Installing OpenShift Without an Infrastructure Provider	225
Performing the Installation of OpenShift Without an Infrastructure Provider	226
Guided Exercise: Performing the Installation of OpenShift Without an Infrastructure Provider	230
Quiz: Chapter Review: Installing OpenShift Without an Infrastructure Provider	240
Summary	244

6. Completing the Installation of OpenShift Without an Infrastructure Provider	245
Performing Day 1 and Day 2 Operations	246
Guided Exercise: Performing Day 1 and Day 2 Operations	248
Replacing a Control Plane Node	262
Guided Exercise: Replacing a Control Plane Node	264
Quiz: Chapter Review: Completing the Installation of OpenShift Without an Infrastructure Provider	273
Summary	275
7. Comprehensive Review	277
Comprehensive Review	278
Lab: Installing a Compact OpenShift Cluster	280

Document Conventions

This section describes various conventions and practices that are used throughout all Red Hat Training courses.

Admonitions

Red Hat Training courses use the following admonitions:



References

These describe where to find external documentation that is relevant to a subject.



Note

Notes are tips, shortcuts, or alternative approaches to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on something that makes your life easier.



Important

Important sections provide details of information that is easily missed: configuration changes that apply only to the current session, or services that need restarting before an update applies. Ignoring these admonitions will not cause data loss, but might cause irritation and frustration.



Warning

Do not ignore warnings. Ignoring these admonitions will most likely cause data loss.

Inclusive Language

Red Hat Training is currently reviewing its use of language in various areas to help remove any potentially offensive terms. This is an ongoing process and requires alignment with the products and services that are covered in Red Hat Training courses. Red Hat appreciates your patience during this process.

Introduction

Red Hat OpenShift Installation Lab

Red Hat OpenShift Installation Lab (DO322) teaches essential skills for installing an OpenShift cluster in a range of environments, from proof of concept to production. Students also learn how to identify customizations that might be required because of the underlying cloud, virtual, or physical infrastructure.

Course Objectives

- Installing OpenShift on a cloud, virtual, or physical infrastructure.

Audience

- Cluster administrators, cluster engineers, and site reliability engineers interested in designing and deploying OpenShift clusters to meet performance and reliability requirements of different workloads.

Prerequisites

- Red Hat Certified Specialist in OpenShift Administration certification (on OpenShift 4) or equivalent knowledge is required.
- Red Hat Certified System Administrator (RHCSA) certification or equivalent knowledge of Red Hat Enterprise Linux system administration is also strongly recommended.

Orientation to the Classroom Environment

The Utility Machine

In this course, the main computer system used for hands-on learning activities (exercises) is **utility**.

The **utility** machine has a standard user account, **lab** with passwordless SSH access. To access the **utility** machine, first log in to the **workstation** machine as the **student** user, with password **student**. From the **workstation** machine, you can access the **utility** machine via SSH as the **lab** user. Some exercises in this course require the **root** user. You can become **root** in the **utility** machine with the command `sudo -i`.

It is from the **utility** machine that you type `oc` commands to manage the OpenShift cluster, that you will install as part of your classroom environment.

It is from the **workstation** machine that you run shell scripts required to complete exercises for this course.

If exercises require that you open a web browser to access any application or web site, then you are required to use the graphical console of the **workstation** machine and use the Firefox web browser from there.

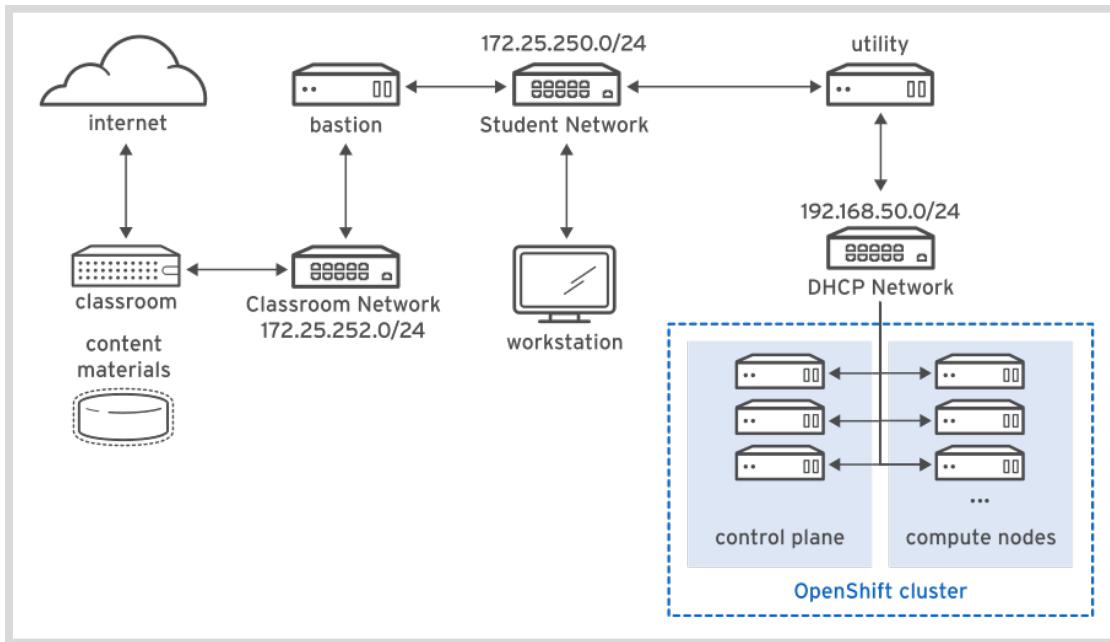
The Classroom Environment

Every student gets a complete classroom environment. As part of this course, every student will be asked to install an OpenShift cluster and perform administration tasks.

The classroom environment runs entirely as virtual machines in a large Red Hat OpenStack Platform cluster, which is shared among many students.

Red Hat Training maintains many OpenStack clusters, in different data centers across the globe, to provide lower latency to students from many countries.

Introduction



All machines on the *Student*, *Classroom*, and *Cluster* Networks run Red Hat Enterprise Linux 8 (RHEL 8), except those machines that are nodes of the OpenShift cluster. These run RHEL CoreOS.

The systems called **bastion**, **utility**, and **classroom** must always be running. They provide infrastructure services required by the classroom environment and its OpenShift cluster.

Usually, the `lab` commands from exercises access these machines when there is a requirement to setup your environment for the exercise, and will require no further action from you.

All systems in the *Student Network* are in the `lab.example.com` DNS domain, and all systems in the *Classroom Network* are in the `example.com` DNS domain.

The systems called **masterXX** and **workerXX** are nodes of the OpenShift 4 cluster that is part of your classroom environment.

All systems in the *Cluster Network* are in the `ocp4.example.com` DNS domain.

Classroom Machines

Machine name	IP addresses	Role
<code>workstation.lab.example.com</code>	172.25.250.9	Graphical workstation used for system administration.
<code>classroom.example.com</code>	172.25.254.254	Router linking the Classroom Network to the Internet.
<code>bastion.lab.example.com</code>	172.25.250.254	Router linking the Student Network to the Classroom Network.
<code>utility.lab.example.com</code>	172.25.250.253	Router linking the Student Network to the Cluster Network and also storage server.

Machine name	IP addresses	Role
master01.ocp4.example.com	192.168.50.10	Control plane and compute node.
master02.ocp4.example.com	192.168.50.11	Control plane and compute node.
master03.ocp4.example.com	192.168.50.12	Control plane and compute node.
worker01.ocp4.example.com	192.168.50.13	Compute node.
worker02.ocp4.example.com	192.168.50.14	Compute node.

Dependencies on Internet Services

Red Hat OpenShift Container Platform 4 requires access to two container registries to download container images for operators, S2I builders, and other cluster services. These registries are:

- `registry.redhat.io`
- `quay.io`

If either registry is unavailable when installing OpenShift, then the OpenShift cluster might not start or could enter a degraded state. To prevent such situations, there is a local registry with a mirror of the OpenShift release images available for every classroom environment.

If these container registries experiences an outage while the classroom environment is up and running, then it might not be possible to complete exercises until the outage is resolved.

The Dedicated OpenShift Cluster

The Red Hat OpenShift Container Platform 4 cluster inside the classroom environment will use the pre-existing infrastructure installation method; all nodes are treated as bare metal servers, even though they are actually virtual machines in an OpenStack cluster.

OpenShift cloud-provider integration capabilities are not enabled. A few features that depend on that integration, such as machine sets and autoscaling of cluster nodes, are not available.

Restoring Access to your OpenShift Cluster

If you suspect that you cannot log in to your OpenShift cluster with `cluster-admin` privileges anymore because you incorrectly changed your cluster authentication settings, then delete the lab environment from within the Red Hat Learning portal and create a new one.

Troubleshooting Access to your OpenShift Cluster

The `utility` machine will be used to run the OpenShift installer inside your classroom environment, and it is a useful resource to troubleshoot cluster issues.

During the guided exercises in this course, you will be requested to create a SSH key for troubleshooting.

You should not require SSH access to your OpenShift cluster nodes for regular administration tasks because OpenShift 4 provides the `oc debug` command. If necessary, you can access the OpenShift nodes using the `core` user and the SSH key generated for troubleshooting to access all cluster nodes. For example:

```
[lab@utility ~]$ ssh -i ~/.ssh/ocp4upi core@node_ip
```

In the preceding example, replace `node_ip` with the IP address of the desired cluster node.

Using Chromium as an Alternative to Firefox

The `workstation` machine runs RHEL 8.2. The latest version of Firefox that is available in the RHEL 8.2 repositories is Firefox 68. If you have issues with rendering any web pages, the EPEL repository is enabled on the `workstation` machine, so you can install the Chromium web browser. If necessary, execute the `sudo yum install chromium` command to install Chromium.

Chapter 1

Describing the OpenShift Installation Process

Goal

Describe and compare the full-stack automation and pre-existing infrastructure installation methods.

Objectives

- Describe and compare the full-stack automation and pre-existing infrastructure installation methods.
- Describe the OpenShift installer and its configuration files.
- Describe the differences between a self-managed OpenShift cluster and hosted OpenShift offerings.

Sections

- Introducing OpenShift Installation Methods (and Quiz)
- Running the OpenShift Installer (and Guided Exercise) (and Quiz)
- Introducing Hosted OpenShift (and Quiz)

Introducing OpenShift Installation Methods

Objectives

- Describe and compare the full-stack automation and pre-existing infrastructure installation methods.

Introducing OpenShift Installation

Because the Red Hat OpenShift Container Platform 3 installation process was quite complex, customers found it difficult to run smoothly. In response to this feedback, Red Hat has redesigned the installer in OpenShift 4 with only those options that are required to create a functional cluster in an opinionated manner.

After the installation completes, administrators can customize and scale it using Day 2 operations.

The opinionated nature of the OpenShift 4 installation process has numerous advantages.

- Automates most of the installation steps to simplify the OpenShift installation process.
- Minimizes human errors during the installation.
- Applies OpenShift 4 best practices.
- Facilitates the integration with future automation efforts such as the OpenShift Assisted Installer, Red Hat Advanced Cluster Management for Kubernetes (ACM), and cluster deployment integration into CI/CD pipelines.

Throughout this course, references to OpenShift refer to Red Hat OpenShift 4.



Note

Opinionated Software is a software product that assumes a certain business process is inherently best and the software is crafted around that approach. The Opinionated Software model helps the product consumer follow the best practices used in the product design.

Describing Ignition Configuration Files

To deploy an OpenShift Container Platform cluster, administrators must use the `openshift-install` binary. This command is also known as the OpenShift installer. The main assets generated by the OpenShift installer are the ignition configuration files for the bootstrap, control plane nodes, and compute nodes. With these three ignition configuration files and the underlying infrastructure correctly configured, administrators can start an OpenShift Container Platform cluster installation.

Ignition is a first boot provisioning tool designed to configure RHCOS systems early in the boot process. It uses the ignition configuration file (JSON-formatted `.ign` file) to declare the desired state of the RHCOS system. The ignition process applies the required configuration during the first boot of the RHCOS system to enforce the desired state on the system. The ignition process:

- Is based on the standard Linux startup process.

Chapter1 | Describing the OpenShift Installation Process

- Runs on physical nodes, virtual nodes, and cloud instances.
- Unifies the `kickstart` and `cloud-init` features on the RHCOS system boot.
- Is executed in the `initramfs` step of the RHCOS boot process.
- Configures storage, systemd units, certificates, users, and custom configurations on RHCOS systems when they first boot.
- Consumes the ignition configuration files generated with the `openshift-install` command and the machine config operator (MCO).
 - The `openshift-install` command uses the ignition configuration files to set the exact state of each node upon installation.
 - The machine config operator applies other changes to the nodes after installation, such as applying new certificates or ssh keys.

The ignition process loads the ignition file configuration files from one of the following locations:

- Local disk
- Cloud metadata
- Over the network using HTTP or HTTPS

You can use the following ignition configuration file example to configure an RHCOS with ignition:

```
[user@demo ~]$ sudo yum install jq
...output omitted...
[user@demo ~]$ cat bootstrap.ign | jq .
{
  "ignition": {
    "version": "3.1.0" 1
  },
  "passwd": { 2
    "users": [
      {
        "name": "core",
        "sshAuthorizedKeys": [
          "ssh-rsa AAA...hlw== lab@utility.lab.example.com\n",
          "ssh-rsa AAA...3DR\n"
        ]
      }
    ]
  },
  "storage": { 3
    "files": [
      {
        ...
      }
    ]
  }
}

{
  "overwrite": false,
  "path": "/etc/motd", 4
  "user": {
    "name": "root"
  },
  "append": [
    ...
  ]
}
```

```
{  
    "source": "data:text/plain;charset=utf-8;base64,VGh...lcg=="  
}  
],  
"mode": 420  
},  
...output omitted...  
"systemd": { ⑤  
    "units": [  
        {  
            "contents": "[Unit]\nDescription=Bootstrap a Kubernetes cluster  
\nRequires=crio-configure.service\nWants=kubelet.service\nAfter=kubelet.service  
crio-configure.service\nConditionPathExists=!/opt/openshift/.bootkube.done\n  
\n[Service]\nWorkingDirectory=/opt/openshift\nExecStart=/usr/local/bin/bootkube.sh  
\n\nRestart=on-failure\nRestartSec=5s\n",  
            "name": "bootkube.service"  
        },  
        ...output omitted...  
    ]  
}  
}
```

- ① Ignition version
- ② Configures SSH public keys for the user core.
- ③ Storage section
- ④ Appends to the /etc/motd file the base64 encoded source data.
- ⑤ Creates a systemd service unit called bootkube.service that executes the script /usr/local/bin/bootkube.sh.

For troubleshooting the ignition process, you can use the following commands.

- By default, the data stored in ignition files is encoded in base64. For inspection, you can decode this data in plain text using the base64 -d command.

```
[user@demo ~]$ echo "VGh...lcg==" | base64 -d  
This is the bootstrap node; it will be destroyed when the master is fully up.  
The primary services are release-image.service followed by bootkube.service. To  
watch their status, run e.g.  
  
journalctl -b -f -u release-image.service -u bootkube.service
```



Note

For more information about ignition configuration files examples: <https://coreos.github.io/ignition/examples>

- After the RHCOS system boots, you can check the ignition configuration used to configure the system.

```
[root@bootstrap ~]$ cat /boot/ignition/config.ign
{"ignition": {"version": "3.1.0"}, "passwd": {"users": [{"name": "core", "sshAuthorizedKeys": ["ssh-rsa..."]}]}}
```

- Also, you can check the ignition logs.

```
[root@bootstrap ~]# journalctl -t ignition
-- Logs begin at Wed 2021-01-27 17:48:36 UTC, end at Mon 2021-02-01 17:52:33 UTC.
--
Jan 27 17:48:38 localhost ignition[684]: Ignition 2.6.0
Jan 27 17:48:38 localhost ignition[684]: Stage: fetch-offline
Jan 27 17:48:38 localhost ignition[684]: reading system config file "/usr/lib/ignition/base.ign"
...output omitted...
Jan 27 17:48:38 localhost ignition[666]: reading system config file "/usr/lib/ignition/user.ign"
...output omitted...
Jan 27 17:48:44 bootstrap ignition[1176]: INFO      : Stage: umount
Jan 27 17:48:44 bootstrap ignition[1176]: INFO      : reading system config file "/usr/lib/ignition/base.ign"
...output omitted...
Jan 27 17:48:44 bootstrap ignition[1176]: INFO      : umount: umount passed
Jan 27 17:48:44 bootstrap ignition[1176]: INFO      : Ignition finished
  successfully
...output omitted...
```

As discussed in this chapter, you do not need to create or modify any ignition configuration files manually for installing OpenShift.

- You can edit the `install-config.yaml` configuration file to customize the OpenShift installation.
- Then, execute the `openshift-installer` command to create the Kubernetes manifests from the `install-config.yaml` configuration file.
- Finally, execute the `openshift-installer` command again to create the ignition files from the Kubernetes manifests.



Note

Modifying ignition configuration files is only supported if you follow Red Hat documented procedures or Red Hat support instructions. Otherwise, this is not supported.



Note

For more information, refer to the *Red Hat Enterprise Linux CoreOS* section of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/architecture

Explaining OpenShift Installation Process

The following sequence of detailed steps explains the OpenShift installation process:

- Step 1: The user runs the OpenShift installer. The installer asks for the necessary cluster information and then creates the installation configuration file `install-config.yaml` accordingly.
- Step 2: From the `install-config.yaml` installation configuration file content, the OpenShift installer creates the Kubernetes manifests. The Kubernetes manifests contain the necessary instructions to build the resources for the OpenShift installation.
- Step 3: From the manifests content, the OpenShift installation process creates the ignition configuration files for the bootstrap node `bootstrap.ign`, control plane nodes `master.ign`, and compute nodes `worker.ign`.

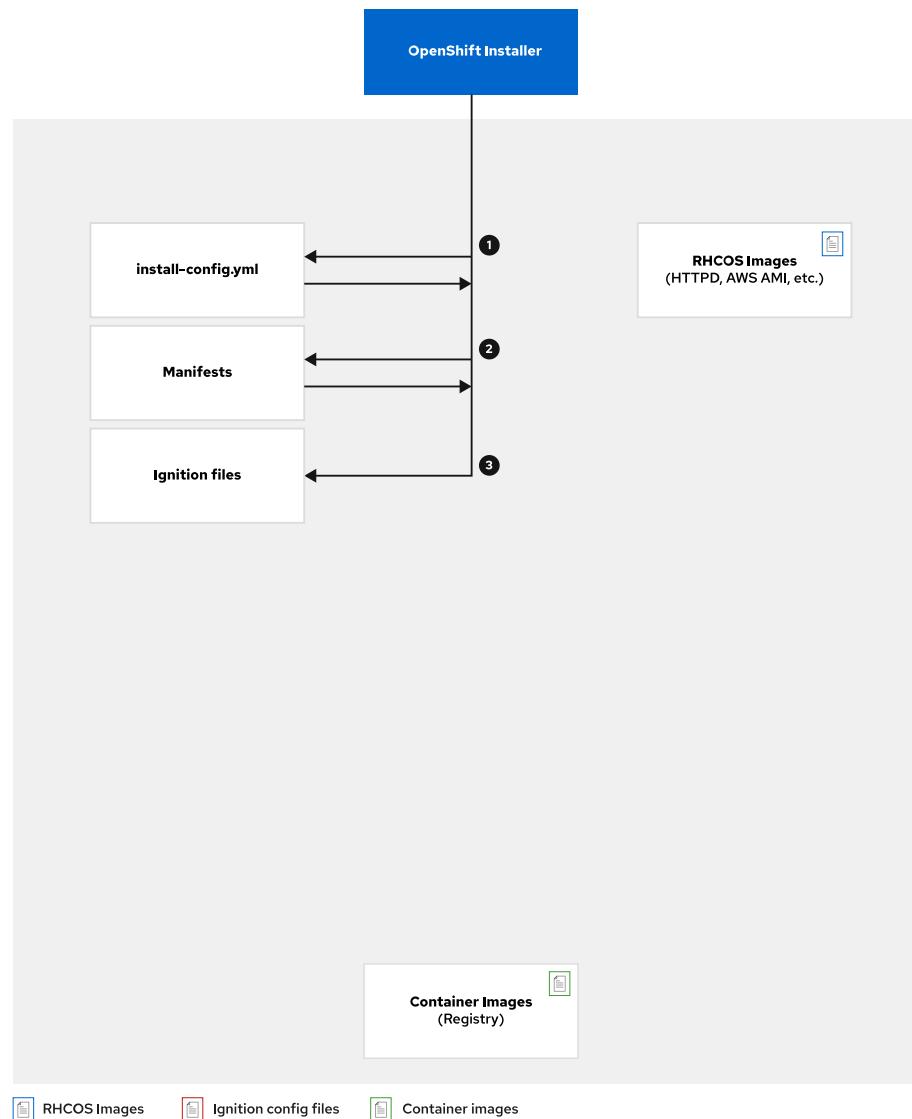


Figure 1.1: OpenShift installation process - Ignition configuration files stage

Chapter1 | Describing the OpenShift Installation Process

- Step 4: The bootstrap node boots and fetches its remote resources (bootstrap.ign) from the initial ignition data source, and then finishes booting. At this stage, the Kubernetes API is running on the bootstrap node.

The bootstrap node hosts the remote resources required for control plane nodes to boot (ignition configuration files) in the Machine Configuration Server (MCS). It also runs a single instance of the etcd cluster.

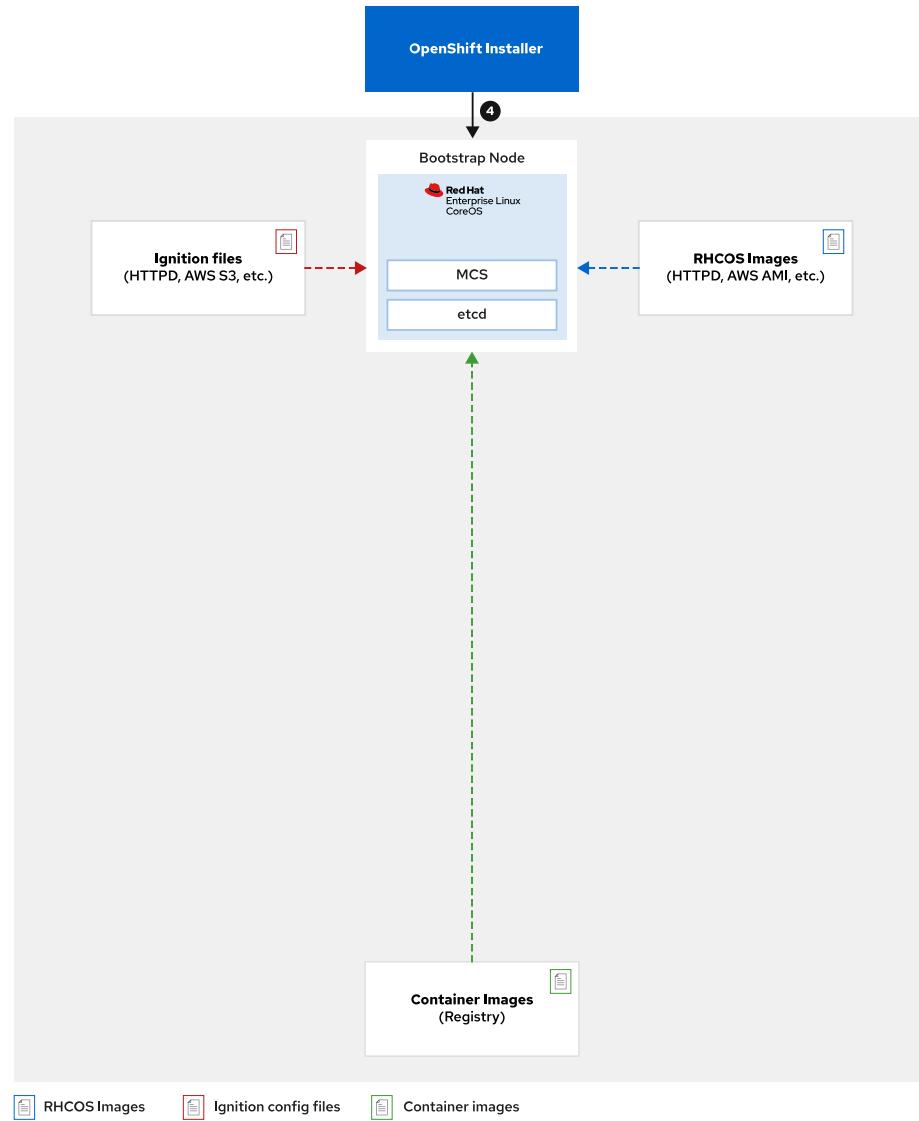


Figure 1.2: OpenShift installation process - Bootstrap (bootkube) stage

**Note**

In this course, references to the Kubernetes API or the OpenShift API refer to the **Kubernetes API Server**.

During the OpenShift installation, the **Kubernetes API Server** runs first on the bootstrap node, and then it moves to the control plane nodes.

Chapter 1 | Describing the OpenShift Installation Process

- Step 5: The control plane nodes boot and fetch their remote resources (the `master.ign` ignition configuration file) from the bootstrap node, and then finish booting.
- Step 6: The bootstrap node starts a temporary control plane and installs the `etcd` operator.

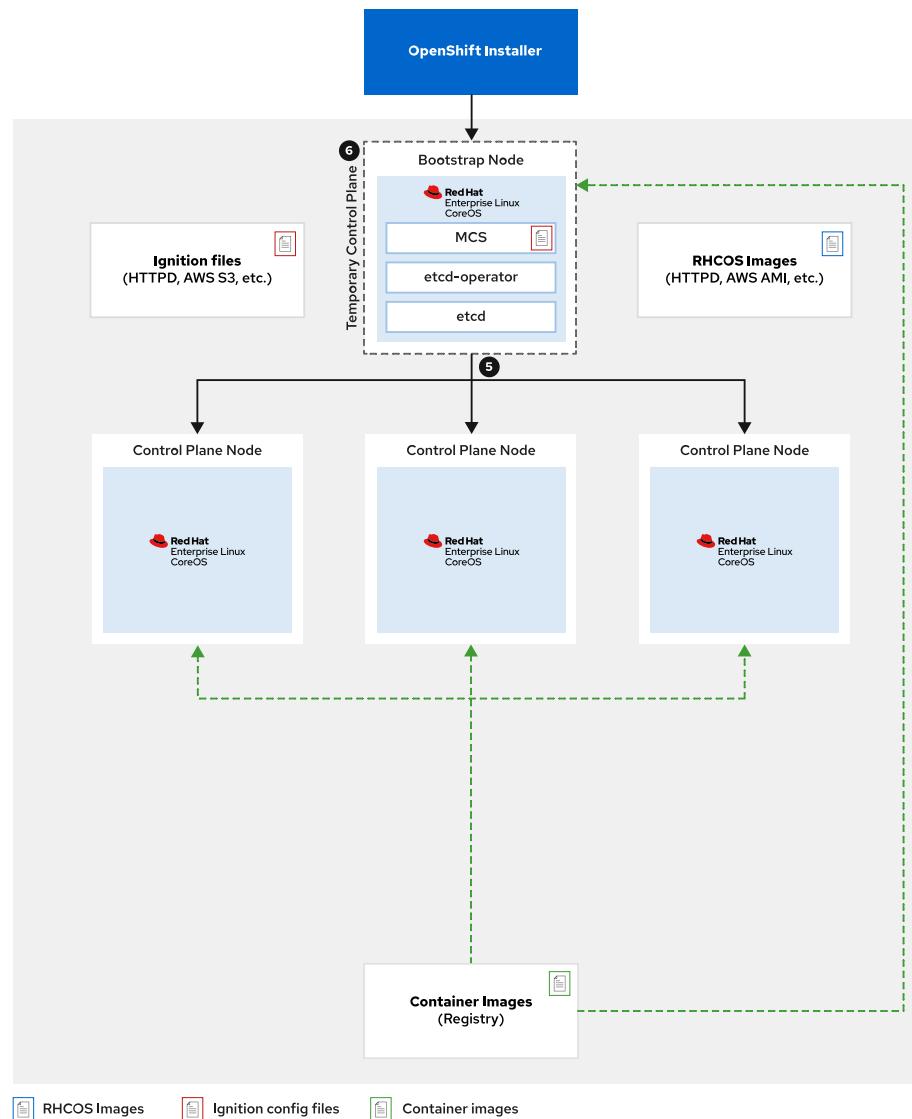


Figure 1.3: OpenShift installation process - Bootstrap (temporary control plane) stage



Note

During the control plane nodes installation, fetching the ignition configuration files happens in two stages: **stage-1** and **stage-2**. At the beginning of the control plane nodes installation (**stage-1**), the control plane nodes fetch their ignition configuration files (`master.ign`) from the initial ignition data source.

These ignition configuration files only contain a redirect instruction to get the corresponding ignition files from the Kubernetes API MCS. Finally, the control plane nodes fetch their ignition configuration from the Kubernetes API MCS (**stage-2**) and finish the installation.

Chapter 1 | Describing the OpenShift Installation Process

- Step 7: The etcd operator running on the bootstrap node scales up the etcd cluster to 3 instances using two control plane nodes.
- Step 8: The temporary control plane running on the bootstrap node schedules the production control plane to the control plane nodes. The OpenShift installation process transfers the etcd cluster to the control plane nodes.

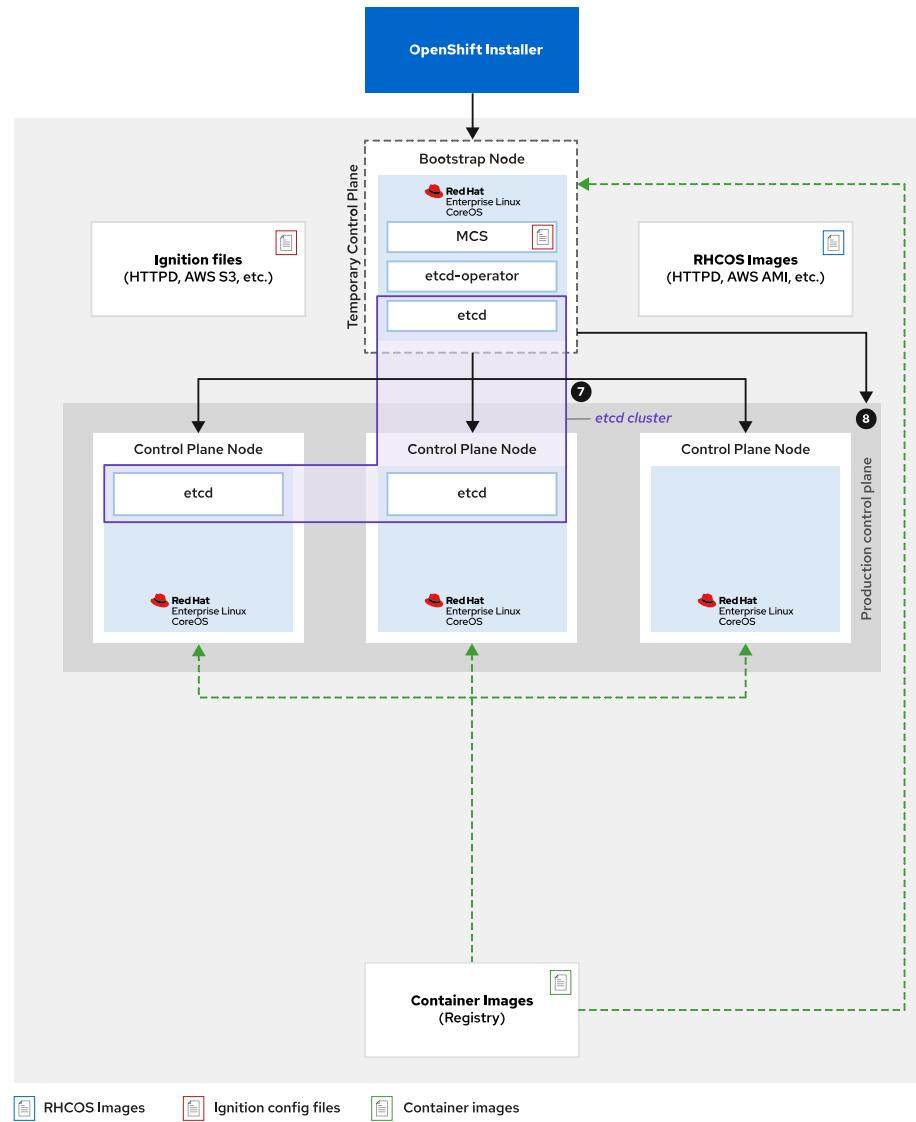


Figure 1.4: OpenShift installation process – Production control plane schedule stage



Note

The temporary control plane is used only during the OpenShift installation. The OpenShift installation process transfers the temporary control plane to the production control plane running on the control plane nodes.

The production control plane is the definitive control plane that manages the OpenShift cluster.

Chapter 1 | Describing the OpenShift Installation Process

- Step 9: The temporary control plane shuts down, yielding to the production control plane. At this stage, the Kubernetes API is running on the production control plane.
- Step 10: For full-stack automation installations, the installer shuts down the bootstrap node. Since this stage, the bootstrap node is no longer needed.

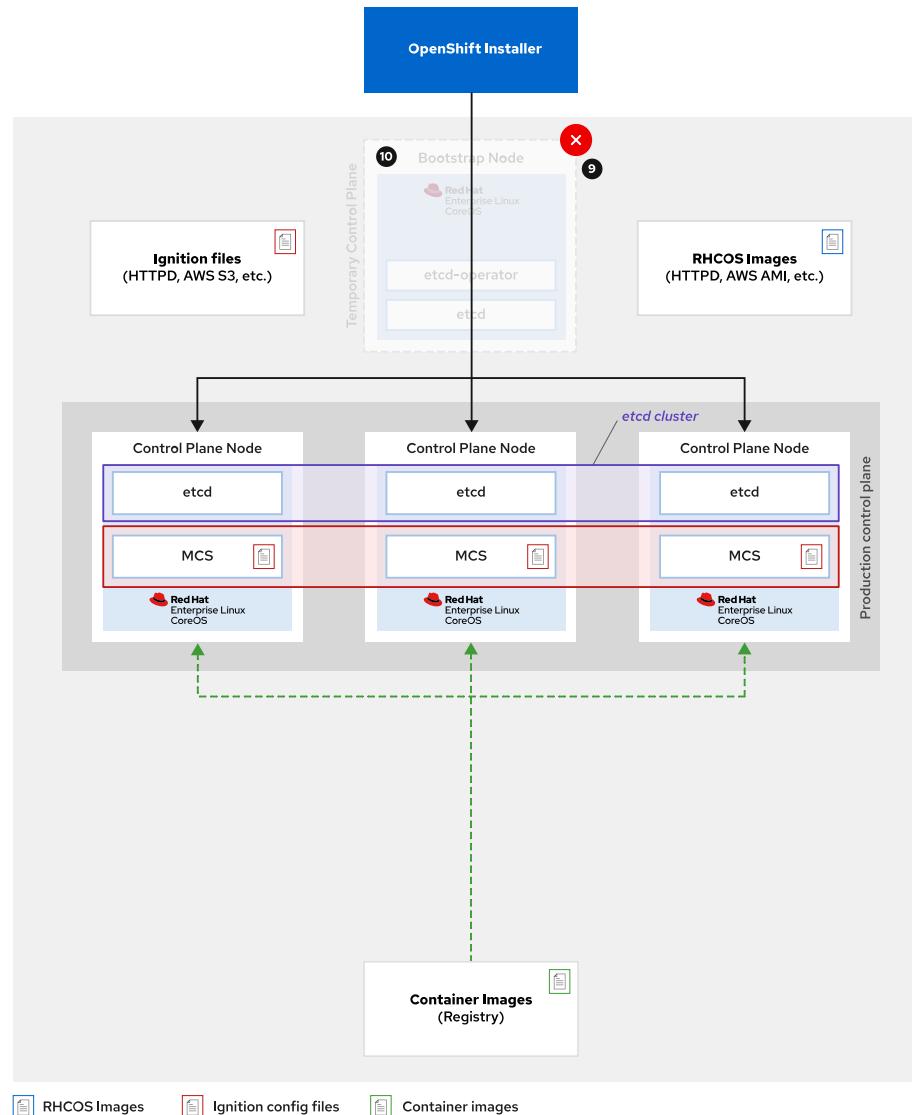


Figure 1.5: OpenShift installation process – Production control plane stage



Note

The etcd cluster runs an etcd pod on each control plane node.

The Kubernetes API MCS service runs a Machine Config Server pod on each control plane node. The Kubernetes API MCS service hosts the `master.ign` and `worker.ign` ignition files.

Chapter 1 | Describing the OpenShift Installation Process

- Step 11: At this stage, the production control plane hosts the cluster remote resources (ignition configuration files) for control plane nodes and compute nodes in their MCS. The compute nodes boot and fetch their remote resources (the `worker.ign` ignition configuration file) from the control plane nodes, finish booting, and join the cluster.

If the pre-existing infrastructure installation method is used, the OpenShift installation process can also install the compute nodes with the RHEL 7 operating system instead of using the default RHCOS operating system.

Support for using RHEL 7 compute nodes is deprecated and will be removed in a future release of OpenShift 4.

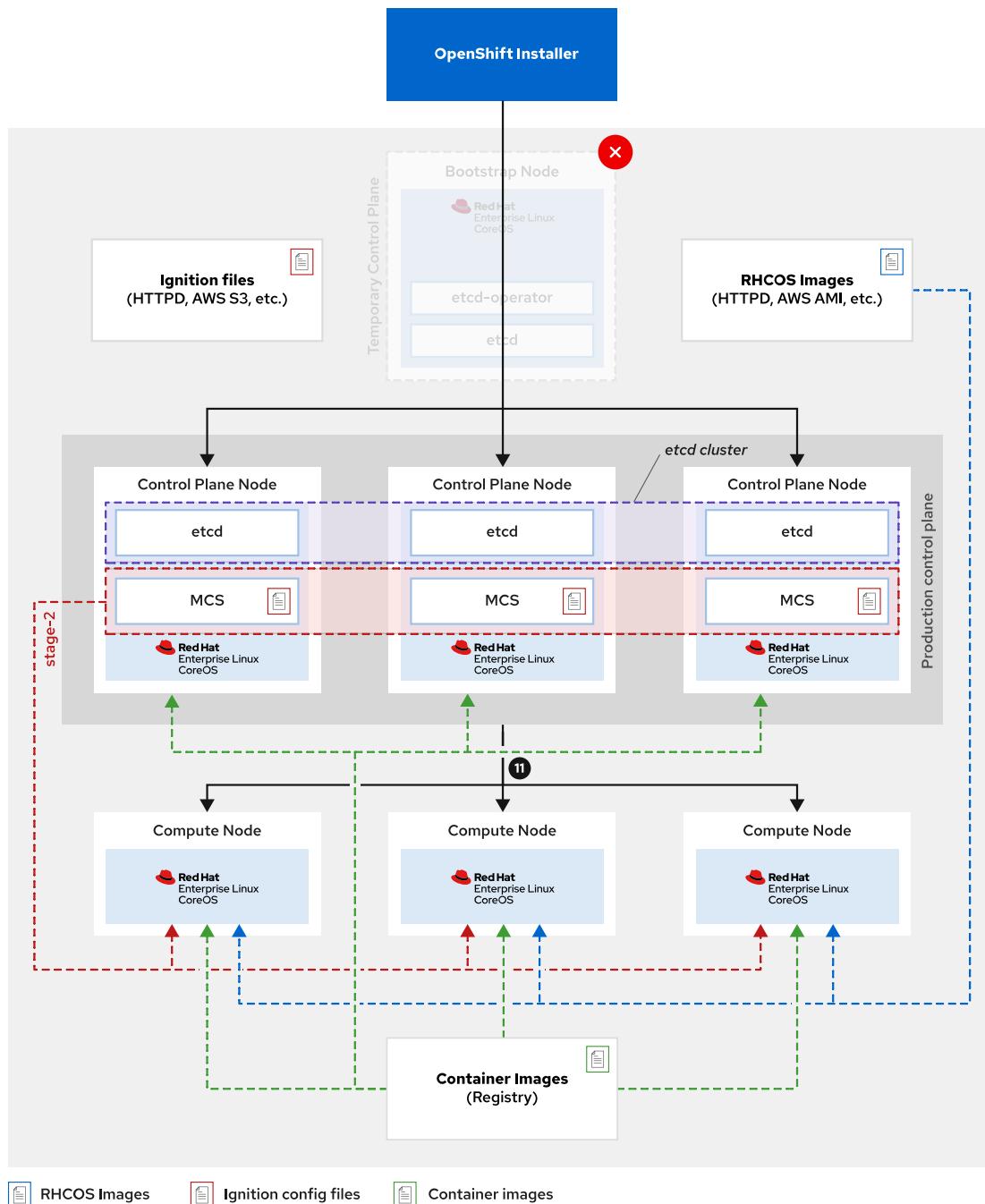


Figure 1.6: OpenShift installation process - Compute nodes installation stage

Chapter 1 | Describing the OpenShift Installation Process

Regarding OpenShift installations:

- Red Hat only supports the use of three control plane nodes.
- Red Hat has tested a maximum of 2000 compute nodes on a Red Hat OpenShift Container Platform 4.6 cluster.
- Red Hat recommends the use of at least two compute nodes to ensure the high availability of the applications running on the cluster.
- For more information, refer to the *Planning your environment according to object maximums* section of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/scalability_and_performance



Note

During the compute nodes installation, fetching the ignition configuration files happens in two stages: **stage -1** and **stage -2**. At the beginning of compute nodes installation (**stage -1**), the compute nodes fetch their ignition configuration files (`worker.ign`) from the initial ignition data source.

These ignition configuration files only contain a redirect instruction to get the corresponding ignition files from the Kubernetes API MCS. Finally, the cluster nodes fetch their ignition configuration from the Kubernetes API MCS (**stage -2**) and finish the installation.

Describing OpenShift Installation Methods

The OpenShift installer offers the following installation methods:

• Full-stack Automation

Using this installation method, administrators install OpenShift with minimal manual intervention in an opinionated manner.

The OpenShift installer deploys the cluster on infrastructure that the installer provisions and the cluster maintains.

• Pre-existing Infrastructure

Using this installation method, administrators have more flexibility when installing OpenShift than they do when using the full-stack automation method.

Administrators use the installer to deploy a cluster on infrastructure that they prepare and maintain themselves.

Full-stack Automation Installation Method

Using this installation method:

- Administrators can use the OpenShift installer to deploy an OpenShift cluster on infrastructure that the OpenShift installer provisions and the OpenShift cluster maintains.
- The OpenShift installer controls all installation areas, including infrastructure provisioning, with an opinionated best practices deployment of OpenShift.

Chapter 1 | Describing the OpenShift Installation Process

- This installation method is frequently called installer-provisioned infrastructure (IPI).

The following diagram describes the OpenShift installation workflow for the full-stack automation installation method.

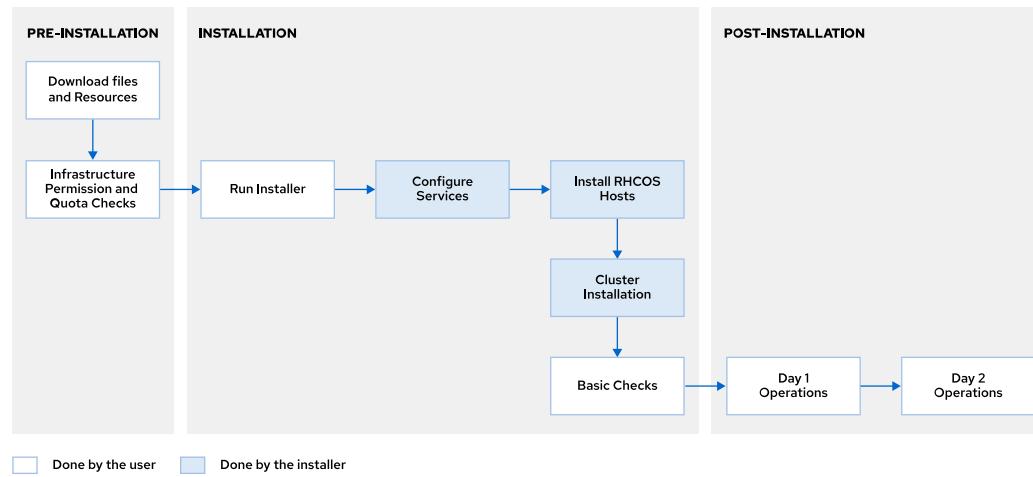


Figure 1.7: OpenShift installation workflow for full-stack automation method

Pre-existing Infrastructure Installation Method

Using this installation method:

- Administrators can use the OpenShift installer to deploy a cluster on infrastructure that they prepare and maintain themselves.
- Administrators are responsible for creating and managing their infrastructure, allowing greater infrastructure customization and operational flexibility.
- This installation method is frequently called user-provisioned infrastructure (UPI).

The following diagram describes the OpenShift installation workflow for the pre-existing infrastructure installation method.

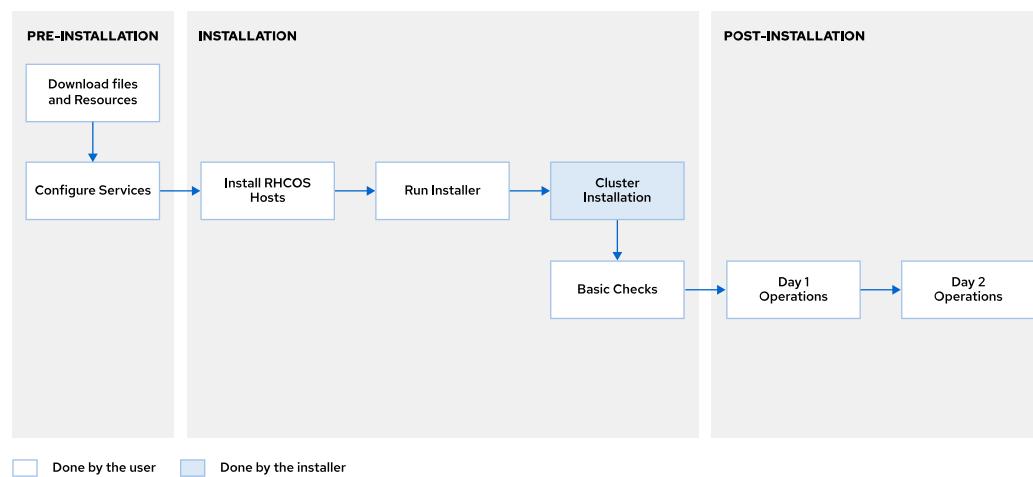


Figure 1.8: OpenShift installation workflow for pre-existing infrastructure method

**Note**

It is common to see the term **installer-provisioned** clusters when referring to clusters installed with the full-stack automation method, and the term **user-provisioned** clusters for clusters installed with the pre-existing infrastructure installation method.

Comparing Full-stack Automation and Pre-existing Infrastructure Installation Methods

The following table shows the basic differences between full-stack automation and pre-existing infrastructure installation methods.

Full-stack Automation & Pre-existing Infrastructure Comparison

Action	Full-stack Automation	Pre-existing Infrastructure
Build network	Installer	User
Setup load balancer	Installer	User
Configure DNS	Installer	User
Hardware or VM provisioning	Installer	User
OS installation	Installer	User
Generate ignition configs	Installer	Installer
Control plane node OS support	Installer: RHCOS	User: RHCOS
Compute node OS support	Installer: RHCOS (1)	User: RHCOS + RHEL 7 (1)
Configure persistent storage for the internal registry	Installer (2)	User
Configure dynamic storage provider	Installer (2)	User
Configure node provisioning and cluster autoscaling	Installer	Only for providers with OpenShift Machine API support (3)

- (1) Full-stack automation and pre-existing infrastructure installation methods support adding RHEL 7 compute nodes deployed by the user as a Day 2 operation.

Support for using RHEL 7 compute nodes is deprecated and will be removed in a future release of OpenShift 4. Only RHCOS cluster nodes will be supported in future OpenShift versions.

For more information, refer to the *Deprecated features* section at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/release_notes

- (2) Except in full-stack automation, installations on bare metal.

Chapter 1 | Describing the OpenShift Installation Process

- (3) Notice the distinction between a pre-existing infrastructure installation with cloud integration (UPI CI) and a pre-existing infrastructure installation on bare metal (UPI BM).
 - For UPI CI installations, if the infrastructure provider has integration with OpenShift Machine API, then the installer configures node provisioning and cluster autoscaling.
 - For UPI BM installations, where the infrastructure provider has no OpenShift Machine API integration support, the installer does not configure node provisioning and cluster autoscaling.

Describing OpenShift Installation Prerequisites

Before installing OpenShift, administrators must ensure that the infrastructure environment used for the installation meets the required prerequisites. The OpenShift installation prerequisites include general prerequisites that apply to both installation methods, and prerequisites specific to the selected installation method.

General Prerequisites

- Provision a bastion host.
- Generate an SSH key on the bastion host.
- Download and install `openshift-install` on the bastion host.
- Download and install `oc` on the bastion host.
- Get the registry `pull-secret`.
- OpenShift cluster nodes must have access to a Network Time Protocol (NTP) server.
- Ensure that the infrastructure network firewall fulfills the OpenShift network access requirements.

Full-stack Automation Installation Prerequisites

- Verify infrastructure permissions and quotas.

For cloud environments, a cloud account with required permissions and quotas is required.

Pre-existing Infrastructure Installation Prerequisites

- Configure network services.
- Provide hardware (physical or virtual) for cluster nodes.
- Install RHCOS on cluster nodes.

Describing OpenShift DNS Prerequisites

When installing OpenShift using the pre-existing infrastructure method, one of the network services that administrators must configure is the DNS service. Administrators must configure the DNS service using DNS records. A complete DNS record takes the form `<component>. <cluster_name>. <base_domain>`:

- `<cluster_name>` is the name of the cluster (for instance `ocp4`).
- `<base_domain>` is the cluster base domain configured in the `install-config.yaml` configuration file (for instance `example.com`).

Chapter 1 | Describing the OpenShift Installation Process

- The reverse DNS records (PTR) are also required.

The following table describes the DNS records required for installing OpenShift:

DNS Records Required for Installing OpenShift

Component	Record	Description
Kubernetes API	api.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME and PTR records to identify the API load balancer for the control plane nodes (resolvable by both clients external to the cluster and from all the nodes within the cluster).
Kubernetes API	api-int.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME and PTR records to identify the API load balancer for the control plane nodes (resolvable from all the nodes within the cluster).
Routes	*.apps.<cluster_name>.<base_domain>.	Wildcard DNS A/AAAA or CNAME record to identify the Application Ingress load balancer that targets the cluster nodes that run the ingress router pod (resolvable by both clients external to the cluster and from all the nodes within the cluster).
Cluster nodes	<name>.<cluster_name>.<base_domain>.	DNS A/AAAA or CNAME and PTR records to identify each cluster node, including bootstrap node (resolvable by the nodes within the cluster).

The following BIND configuration shows the DNS configuration used for installing an OpenShift cluster named ocp4 in the example.com base domain:

```
[root@utility ~]# cat /etc/named.conf
...output omitted...
zone "example.com" {
    type master;
    file "example.com.db";
    allow-update { none; };
};

...output omitted...
```

```
[root@utility ~]# cat /var/named/example.com.db
$TTL 1D
@ IN SOA dns.ocp4.example.com. root.example.com. (
```

Chapter1 | Describing the OpenShift Installation Process

```
2019022400 ; serial
            ; refresh
            ; retry
            ; expire
            ; minimum
)
IN NS dns.ocp4.example.com.
dns.ocp4      IN A 192.168.50.254
api.ocp4      IN A 192.168.50.254
api-int.ocp4   IN A 192.168.50.254
*.apps.ocp4   IN A 192.168.50.254
bootstrap.ocp4 IN A 192.168.50.9
master01.ocp4  IN A 192.168.50.10
master02.ocp4  IN A 192.168.50.11
master03.ocp4  IN A 192.168.50.12
worker01.ocp4  IN A 192.168.50.13
worker02.ocp4  IN A 192.168.50.14
```

```
[root@utility ~]# cat /etc/named.conf
...output omitted...
zone "50.168.192.in-addr.arpa" IN {
    type master;
    file "example.com.reverse.db";
    allow-update { none; };
};
...output omitted...
```

```
[root@utility ~]# cat /var/named/example.com.reverse.db
$TTL 1D
@    IN SOA dns.ocp4.example.com. root.example.com. (
            2019022400 ; serial
            3h          ; refresh
            15          ; retry
            1w          ; expire
            3h          ; minimum
)
IN NS dns.ocp4.example.com.
254  IN PTR api.ocp4.example.com.
254  IN PTR api-int.ocp4.example.com.
9    IN PTR bootstrap.ocp4.example.com.
10   IN PTR master01.ocp4.example.com.
11   IN PTR master02.ocp4.example.com.
12   IN PTR master03.ocp4.example.com.
13   IN PTR worker01.ocp4.example.com.
14   IN PTR worker02.ocp4.example.com.
```

You can use the `dig` command to verify the DNS service configuration before installing OpenShift.

Chapter 1 | Describing the OpenShift Installation Process

```
[lab@utility ~]$ dig @dns.ocp4.example.com api.ocp4.example.com  
...output omitted...  
;; ANSWER SECTION:  
api.ocp4.example.com. 86400 IN A 192.168.50.254  
...output omitted...
```

```
[lab@utility ~]$ dig @dns.ocp4.example.com api-int.ocp4.example.com  
...output omitted...  
;; ANSWER SECTION:  
api-int.ocp4.example.com. 86400 IN A 192.168.50.254  
...output omitted...
```

```
[lab@utility ~]$ dig @dns.ocp4.example.com -x 192.168.50.254  
...output omitted...  
;; ANSWER SECTION:  
254.50.168.192.in-addr.arpa. 86400 IN PTR api.ocp4.example.com.  
254.50.168.192.in-addr.arpa. 86400 IN PTR api-int.ocp4.example.com.  
...output omitted...
```

```
[lab@utility ~]$ dig @dns.ocp4.example.com master01.ocp4.example.com  
...output omitted...  
;; ANSWER SECTION:  
master01.ocp4.example.com. 86400 IN A 192.168.50.10  
...output omitted...
```

```
[lab@utility ~]$ dig @dns.ocp4.example.com -x 192.168.50.10  
...output omitted...  
;; ANSWER SECTION:  
10.50.168.192.in-addr.arpa. 86400 IN PTR master01.ocp4.example.com.  
...output omitted...
```

Repeat this procedure for all the cluster nodes, including the bootstrap node.

**Note**

By default, when installing OpenShift using the full-stack automation method on supported cloud providers, the OpenShift installer automatically performs the DNS service configuration.

Administrators must configure the DNS service when installing OpenShift using the full-stack automation method on the following infrastructures:

- Red Hat OpenStack Platform cloud provider using an external DNS
- VMware vSphere
- Bare metal

Describing OpenShift Firewall Prerequisites

As explained in the *General Prerequisites* section, you must ensure that your infrastructure network firewall configuration meets the OpenShift network access requirements before installing

Chapter1 | Describing the OpenShift Installation Process

OpenShift. For connected installations, the OpenShift cluster needs access to the remote sites it requires to function. For more information, refer to the *Configuring your firewall* section of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing

Additionally, the OpenShift cluster nodes must have some network ports accessible on the cluster network. For more information, refer to the *Installing on bare metal* chapter of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_bare_metal

When using a network firewall, you must ensure network connectivity between the cluster nodes. Also, each cluster node must be able to resolve the host names of all other nodes in the cluster.

Connectivity Requirements from All Cluster Nodes to All Cluster Nodes

Protocol	Port	Description
ICMP	N/A	Network reachability tests
TCP	9000-9999	Host level services, including the node exporter on ports 9100-9101 and the Cluster Version Operator on port 9099
TCP	10250-10259	The default ports that Kubernetes reserves
TCP	10256	openshift-sdn
UDP	4789	VXLAN and Geneve
UDP	6081	VXLAN and Geneve
UDP	9000-9999	Host level services, including the node exporter on ports 9100-9101
TCP/UDP	30000-32767	Kubernetes node port

Connectivity Requirements from All Cluster Nodes to Control Plane Nodes

Protocol	Port	Description
TCP	2379-2380	etcd server, peer, and metrics ports
TCP	6443	Kubernetes API

Ensure that the load balancers used by the OpenShift cluster have both the front (load balancer) and back (cluster nodes) network ports accessible.

The API Load Balancer (API LB) provides a common endpoint to interact with and configure the platform. The Application Ingress Load Balancer (APP Ingress LB) provides an ingress point for application traffic flowing in from outside the cluster.

API Load Balancer Ports

TCP Port	Back-end nodes (pool members)	Internal access	External access	Description
6443	Bootstrap (temporary) and control plane nodes	Yes	Yes	Kubernetes API server
22623	Bootstrap (temporary) and control plane nodes	Yes	No	Machine Config server

Application Ingress Load Balancer Ports

TCP Port	Back-end nodes (pool members)	Internal access	External access	Description
443	Cluster nodes that run the Ingress router pods	Yes	Yes	HTTPS traffic
80	Cluster nodes that run the Ingress router pods	Yes	Yes	HTTP traffic

**Note**

The firewall configuration explained in this section assumes the following:

- You are installing an OpenShift cluster using the pre-existing infrastructure installation method.
- The network infrastructure used to install the cluster has network restrictions managed by a network firewall.

When installing an OpenShift cluster with the full-stack automation in a supported cloud provider:

- By default, the installer creates a new virtual private network (VPC) for the cluster with all the network access requirements in place.
- If you want to install the cluster on an existing virtual private network (VPC), then you must verify that the existing VPC fulfills all the network access requirements.

Describing OpenShift Installation Modes

Depending on the external connectivity of the cluster, administrators can use one of the following installation modes:

- **Connected**

The cluster nodes have internet access to pull container images from the quay.io and registry.redhat.io registries. The connected installation mode is supported when using the full-stack automation or the pre-existing infrastructure installation methods.

- **Disconnected**

Administrators use this installation mode when a connected installation is not possible. The OpenShift installer uses a local container registry to pull container images. Not all cloud providers support installation in disconnected mode.

Introducing Disconnected Installations

Administrators can install an OpenShift cluster on infrastructure that they provide in a restricted network. This installation mode is also known as the disconnected installation mode. The disconnected installation:

- Requires mirroring images to an API schema² specification-compliant local container registry.
- Uses the exact version of images provided in the payload by digest.
- Requires access to the Internet to mirror the container images from the source registries to the local container registry.
- Requires mirroring the images required for the OpenShift installation and subsequent product updates to the local container registry.

Before starting a disconnected installation, you must mirror the required images to the local container registry and obtain the `imageContentSources` data for your OpenShift version. Mirror the following content in the local container registry:

- The OpenShift Container Platform image repository

This repository provides the container images to use during OpenShift cluster installation or upgrade.

- Cluster samples operator image streams

Because most image streams in the `openshift` namespace use images located in the Red Hat registry `registry.redhat.io`, you must mirror those images and configure the image streams accordingly.

- Remote OperatorHub sources
- Remote Operator Hub sources

In disconnected clusters, the operator lifecycle manager (OLM) cannot access the Red Hat provided Operator Hub sources hosted on `quay.io`. Administrators must mirror the Operator Hub sources in the local container registry and configure OLM to install and manage the operators from the local sources instead of the default remote sources.

Instructions to mirror the OpenShift Container Platform image repository and the cluster samples operator image streams in the local registry are available in the *Creating a mirror registry for installation in a restricted network* section of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing.

Instructions to mirror the remote Operator Hub sources in the local registry are available in the *Using Operator Lifecycle Manager on restricted networks* section of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/operators/index#olm-restricted-networks.

OpenShift clusters in restricted networks have the following limitations:

- The `ClusterVersion` status shows an `Unable to retrieve available updates` error message.
- By default, the Developer Catalog contents are unavailable because the cluster does not have access to the required image stream tags.
- The OpenShift Telemetry service is disabled.

Customizing Disconnected Installations

After mirroring the local container registry, start the OpenShift installation process. You must modify the `install-config.yaml` file to customize the disconnected OpenShift installation.

The following is a sample `install-config.yaml` file used in a connected installation:

```
[user@demo ~]$ cat ${HOME}/ocp4-cluster/install-config.yaml
apiVersion: v1
baseDomain: example.com
#proxy: ①
# httpProxy: http://<username>:<pswd>@<ip>:<port>
# httpsProxy: http://<username>:<pswd>@<ip>:<port>
# noProxy: example.com
compute:
- hyperthreading: Enabled
  name: worker
  replicas: 2
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
metadata:
  name: ocp4
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  none: {}
fips: false
pullSecret: |
  {"auths":...output omitted...} ②
sshKey: |
  ssh-rsa AA...output omitted...
```

- ① If required, you can use a network proxy in a connected installation.
- ② The `pull-secret` contains the credentials to authenticate against `quay.io` and `registry.redhat.io`.

The following is a sample `install-config.yaml` file used in a disconnected installation:

```
[user@demo ~]$ cat ${HOME}/ocp4-cluster/install-config.yaml
apiVersion: v1
baseDomain: example.com
compute:
- hyperthreading: Enabled
  name: worker
  replicas: 2
controlPlane:
  hyperthreading: Enabled
```

```
name: master
replicas: 3
metadata:
  name: ocp4
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  networkType: OpenShiftSDN
  serviceNetwork:
    - 172.30.0.0/16
platform:
  none: {}
fips: false
pullSecret: |
  {"auths":...output omitted...} ①
sshKey: |
  ssh-rsa AA...output omitted...
additionalTrustBundle: | ②
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: ③
- mirrors:
  - nexus-registry-int.apps.tools.dev.nextcle.com/openshift/ocp4
    source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - nexus-registry-int.apps.tools.dev.nextcle.com/openshift/ocp4
    source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

- ① The pull-secret contains the credentials to authenticate against the `nexus-registry-int.apps.tools.dev.nextcle.com` local container registry.
- ② If the certificate used by the local container registry is not trusted, provide the certificate bundle that you used for the local container registry.
- ③ Add the `imageContentSources` section from the output of the command used to mirror the repository. This customization will configure the file `/etc/containers/registries.conf` on each cluster node to enable image mirroring.



Note

The classroom environment used to perform the guided exercises for this course has full internet connectivity. There is a local registry with a mirror of the OpenShift release images and other resources in the classroom environment.



References

For more information, refer to the *Installation and update* chapter of the Red Hat OpenShift Container Platform 4.6 documentation at
https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/architecture

► Quiz

Introducing OpenShift Installation Methods

Choose the correct answers to the following questions:

- ▶ 1. **Which two of the following are prerequisites for installing OpenShift using the connected mode? (Choose two.)**
 - a. Get the registry pull-secret
 - b. Use a local registry
 - c. Download and install the oc command on the bastion host
 - d. Install Red Hat Enterprise Linux 8 on the bootstrap node

- ▶ 2. **Which of the following OpenShift installation methods is recommended for installing OpenShift with minimal administrator intervention?**
 - a. Pre-existing infrastructure installation method
 - b. Full-stack automation installation method

- ▶ 3. **Which two of the following assets are created by the OpenShift installer? (Choose two.)**
 - a. Ignition configuration files
 - b. Operator images
 - c. Kubernetes manifests
 - d. RHCOS images

- ▶ 4. **Which three of the following OpenShift installation steps are done automatically by the OpenShift installer when using the full-stack automation installation method? (Choose three.)**
 - a. Setup an authentication provider.
 - b. Install RHCOS on cluster nodes.
 - c. Setup a load balancer for API traffic.
 - d. Install Red Hat Enterprise Linux 8 on compute nodes.
 - e. Build the network where the cluster is installed.

- ▶ 5. **Which of the following statements best describes the role of the bootstrap node in the OpenShift installation process?**
 - a. A temporary node that stores all installation resources as the RHCOS images
 - b. A temporary node that runs a minimal Kubernetes deployment used to install the OpenShift production control plane
 - c. A temporary node that runs the OpenShift production control plane

► Solution

Introducing OpenShift Installation Methods

Choose the correct answers to the following questions:

- ▶ 1. **Which two of the following are prerequisites for installing OpenShift using the connected mode? (Choose two.)**
 - a. Get the registry pull-secret
 - b. Use a local registry
 - c. Download and install the oc command on the bastion host
 - d. Install Red Hat Enterprise Linux 8 on the bootstrap node

- ▶ 2. **Which of the following OpenShift installation methods is recommended for installing OpenShift with minimal administrator intervention?**
 - a. Pre-existing infrastructure installation method
 - b. Full-stack automation installation method

- ▶ 3. **Which two of the following assets are created by the OpenShift installer? (Choose two.)**
 - a. Ignition configuration files
 - b. Operator images
 - c. Kubernetes manifests
 - d. RHCOS images

- ▶ 4. **Which three of the following OpenShift installation steps are done automatically by the OpenShift installer when using the full-stack automation installation method? (Choose three.)**
 - a. Setup an authentication provider.
 - b. Install RHCOS on cluster nodes.
 - c. Setup a load balancer for API traffic.
 - d. Install Red Hat Enterprise Linux 8 on compute nodes.
 - e. Build the network where the cluster is installed.

- ▶ 5. **Which of the following statements best describes the role of the bootstrap node in the OpenShift installation process?**
 - a. A temporary node that stores all installation resources as the RHCOS images
 - b. A temporary node that runs a minimal Kubernetes deployment used to install the OpenShift production control plane
 - c. A temporary node that runs the OpenShift production control plane

Running the OpenShift Installer

Objectives

- Describe the OpenShift installer and its configuration files.

Using OpenShift Installation Binary

Regardless of the installation method, you must use the OpenShift Installer binary `openshift-install` to start an OpenShift installation. This installer ensures simple, uniform, and opinionated behavior during the OpenShift installation process.



Note

Starting with OpenShift Container Platform 4.6, a full-stack installation on bare metal uses the OpenShift installer binary `openshift-baremetal-install`.

The following table shows some useful options available to the `openshift-install<command options>`

Option	Description
<code>help</code>	Prints help information.
<code>explain -h</code>	Explains the fields related to each supported <code>InstallConfig</code> API.
<code>explain installconfig</code>	Explains the <code>installconfig</code> resource and its fields.
<code>--log-level debug</code>	Enables debug mode.
<code>--dir</code>	Configures the directory path to store the generated assets.
<code>create install-config</code>	Creates installation configuration file <code>install-config.yaml</code> .
<code>create manifests</code>	Creates Kubernetes manifests.
<code>create ignition-configs</code>	Creates ignition configuration files.
<code>create cluster</code>	Deploys the cluster.
<code>wait-for bootstrap-complete</code>	Waits until the bootstrap installation phase ends.
<code>wait-for install-complete</code>	Waits until the cluster install ends.

Option	Description
destroy cluster	Removes the cluster

The following sequence shows the high-level steps required to install OpenShift:

1. Fulfill the installation prerequisites.
2. Create the installation directory.
3. Create the installer configuration file, `install-config.yaml`.
4. Generate the Kubernetes manifests.
5. Generate the ignition configuration files.
6. Deploy the OpenShift cluster.
7. Verify the OpenShift cluster health.

Creating Installation Directory

To start installing OpenShift, first create the installation directory. This directory stores all the files created by the OpenShift installer.

```
[user@demo ~]$ mkdir ${HOME}/ocp4-cluster
```



Note

The OpenShift installer will create the installation directory if it does not exist.

Creating Installer Configuration File

After you create the installation directory, run the OpenShift installer to create the installer configuration file `install-config.yaml`. The installer prompts you for the necessary cluster information and then creates the `install-config.yaml` file accordingly.

```
[user@demo ~]$ openshift-install create install-config \
> --dir=${HOME}/ocp4-cluster
? SSH Public Key /home/user/.ssh/ocp4-cluster.pub
? Platform aws
INFO Credentials loaded from the "default" profile in file "/home/user/.aws/
credentials"
? Region us-east-2
? Base Domain mydomain.com
? Cluster Name ocp4
? Pull Secret [? for help] +++++
INFO Install-Config created in: /home/user/ocp4-cluster
```



Note

This example demonstrates the creation of the `install-config.yaml` configuration file for an OpenShift installation on AWS.

Chapter 1 | Describing the OpenShift Installation Process

The following is a sample `install-config.yaml` configuration file:

```
[user@demo ~]$ cat ${HOME}/ocp4-cluster/install-config.yaml
apiVersion: v1
baseDomain: mydomain.com
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 3
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform: {}
  replicas: 3
metadata:
  creationTimestamp: null
  name: ocp4
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-east-2
publish: External
pullSecret: |
  {"auths":...}
sshKey: |
  ssh-rsa AA...
```



Note

Before creating the Kubernetes manifests, you can edit the `install-config.yaml` configuration file to customize the OpenShift installation. The most common installation customizations are discussed elsewhere in this course.

Generating Kubernetes Manifests

After creating the `install-config.yaml` file, run the OpenShift installer to generate the Kubernetes manifests. A Kubernetes manifest is a file that describes one or more Kubernetes API objects, such as `Pods`, `Services`, `Deployments` or `MachineConfigs`. The Kubernetes manifests contain the instructions necessary to build the resources for the OpenShift installation.

```
[user@demo ~]$ openshift-install create manifests \
> --dir=${HOME}/ocp4-cluster
INFO Consuming Install Config from target directory
INFO Manifests created in: /home/user/ocp4-cluster/manifests and /home/user/ocp4-
cluster/openshift
```

You can review the Kubernetes manifests generated by the OpenShift installer in the installation directory.

```
[user@demo ~]$ find ${HOME}/ocp4-cluster/manifests
/home/user/ocp4-cluster/manifests
/home/user/ocp4-cluster/manifests/04-openshift-machine-config-operator.yaml
...output omitted...
```

The following is an example of an official procedure from Red Hat that explains how to create a new Kubernetes manifest to add the `loglevel=7` kernel argument to the control plane nodes during the installation. In the `${HOME}/ocp4-cluster/manifests/openshift` directory, create a file called `99-openshift-machineconfig-master-kargs.yaml` that defines a `MachineConfig` object for adding the kernel setting.

```
[user@demo ~]$ cd ${HOME}/ocp4-cluster/manifests/openshift
```

```
[user@demo ~]$ cat << EOF > 99-openshift-machineconfig-master-kargs.yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: master
  name: 99-openshift-machineconfig-master-kargs
spec:
  kernelArguments:
    - 'loglevel=7'
EOF
```

When creating the ignition files from the manifests, this new manifest is wrapped and added to the control plane nodes ignition file `master.ign`.

**Warning**

Modifying Kubernetes manifests is only supported if you follow Red Hat documented procedures or Red Hat support instructions. Otherwise, this is not supported.

Generating Ignition Configuration Files

After you generate the Kubernetes manifests, run the OpenShift installer to create the ignition configuration files from the manifests content. The OpenShift installer creates the bootstrap, control plane nodes, and the compute nodes ignition configuration files. When deploying the OpenShift cluster, the OpenShift installer uses these ignition configuration files to install and configure RHCOS on the bootstrap node, control plane nodes, and compute nodes.

```
[user@demo ~]$ openshift-install create ignition-configs \
> --dir=${HOME}/ocp4-cluster
INFO Consuming Master Machines from target directory
INFO Consuming OpenShift Install (Manifests) from target directory
INFO Consuming Openshift Manifests from target directory
INFO Consuming Worker Machines from target directory
INFO Consuming Common Manifests from target directory
INFO Ignition-Configs created in: /home/user/ocp4-cluster and /home/user/ocp4-
cluster/auth
```

You can review the ignition configuration files generated by the OpenShift installer in the installation directory.

```
[user@demo ~]$ find ${HOME}/ocp4-cluster -name '*.ign' | xargs ls -lrt
-rw-r-----. 1 user user 1732 Dec 27 19:35 /home/user/ocp4-cluster/master.ign
-rw-r-----. 1 user user 1732 Dec 27 19:35 /home/user/ocp4-cluster/worker.ign
-rw-r-----. 1 user user 307594 Dec 27 19:35 /home/user/ocp4-cluster/bootstrap.ign
```

**Note**

Ignition files are valid for 24 hours, after which the included certificates expire. If the cluster installation fails, check the ignition files to see if they are more than 24 hours old. If so, create new ignition files.

**Warning**

Modifying ignition configuration files is only supported if you follow Red Hat documented procedures or Red Hat support instructions. Otherwise, this is not supported.

Deploying OpenShift Cluster

After you generate the ignition configuration files, run the OpenShift installer to deploy the OpenShift cluster.

```
[user@demo ~]$ openshift-install create cluster \
> --dir=${HOME}/ocp4-cluster --log-level=debug
DEBUG OpenShift Installer 4.6.4
DEBUG Built from commit 6e02d049701437fa81521fe981405745a62c86c5
...output omitted...
INFO Consuming Bootstrap Ignition Config from target directory
INFO Consuming Worker Ignition Config from target directory
INFO Consuming Master Ignition Config from target directory
...output omitted...
DEBUG Apply complete! Resources: 122 added, 0 changed, 0 destroyed.
DEBUG OpenShift Installer 4.6.4
DEBUG Built from commit 6e02d049701437fa81521fe981405745a62c86c5
INFO Waiting up to 20m0s for the Kubernetes API at https://
api.ocp4.example.com:6443...
INFO API v1.19.0+9f84db3 up
INFO Waiting up to 30m0s for bootstrapping to complete...
```

```

...output omitted...
DEBUG Bootstrap status: complete
INFO Destroying the bootstrap resources...
...output omitted...
INFO Waiting up to 40m0s for the cluster at https://api.ocp4.example.com:6443 to initialize...
DEBUG Still waiting for the cluster to initialize: Working towards 4.6.4: 82%
complete
...output omitted...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/user/ocp4-aws-cluster/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.ocp4.example.com
INFO Login to the console with user: "kubeadmin", and password: "xxxx"
...output omitted...
INFO Time elapsed: 34m10s

```

In a pre-existing infrastructure installation, you cannot use the `openshift-install create cluster` command to deploy the cluster because you have already installed the cluster nodes. The bootstrap node installation triggers the cluster installation, so execute the following command sequence to monitor the cluster installation:

```
[user@demo ~]$ openshift-install wait-for bootstrap-complete \
> --dir=${HOME}/ocp4-cluster --log-level=debug
```

```
[user@demo ~]$ openshift-install wait-for install-complete \
> --dir=${HOME}/ocp4-cluster --log-level=debug
```



Note

The `openshift-install wait-for` commands do not trigger the cluster installation. It is a recommended practice to use them to monitor the cluster installation.

Monitoring OpenShift Installations

In addition to reviewing the installation logs, you can perform active installation process monitoring using the `oc` command. You can start using the `oc` command once the bootstrap has the Kubernetes API running on the temporary control plane.

To start using the `oc` command, authenticate against the Kubernetes API with `cluster-admin` privileges using one of the following methods:

- Configuring the `KUBECONFIG` environment variable to use the `kubeconfig` file.

```
[user@demo ~]$ export KUBECONFIG=${HOME}/ocp4-cluster/auth/kubeconfig
```

- Using the `kubeadmin` credentials stored in the `kubeadmin-password` file.



Warning

Save the kubeconfig and kubeadmin-password files in a secure location. These files grant `cluster-admin` privileges in the cluster. Handle them accordingly.

After logging in to the Kubernetes API as `cluster-admin`, use the following sequence of `oc` commands to monitor the installation process:

```
[user@demo ~]$ watch 'oc get clusterversion; oc get clusteroperators; \
> oc get pods --all-namespaces | grep -v -E "Running|Completed"; oc get nodes'
...output omitted...
```

Using the `watch` command, you obtain a real-time view of how the cluster version operator (CVO) installs the cluster operators, which pods are not yet running on the cluster, and the status of cluster nodes. The `watch` command automatically runs the `oc` command sequence every 2 seconds, so it is useful for monitoring the progression of the cluster installation.

```
[user@demo ~]$ watch 'oc get clusterversion; oc get clusteroperators; \
> oc get pods --all-namespaces | grep -v -E "Running|Completed"; oc get nodes'
NAME      VERSION AVAILABLE PROGRESSING SINCE STATUS
version    False     True      13m   Working towards 4.6.4: 98% complete

NAME                                     VERSION AVAILABLE PROGRESSING DEGRADED SINCE
authentication                         4.6.4   False   False   True   7m59s
cloud-credential                       True    False   False   11m
cluster-autoscaler                     4.6.4   True    False   False   6m34s
config-operator                        4.6.4   True    False   False   8m4s
console                                4.6.4   False   True    False   60s
...output omitted...

NAMESPACE          NAME           READY STATUS      RESTARTS AGE
openshift-apiserver  apiserver-5c4b794d87 0/2 Pending      0        2m13s
openshift-kube-apiserver  kube-apiserver-ip-10 0/5 Init:0/1  0        61s
...output omitted...

NAME      STATUS  ROLES      AGE      VERSION
master01  Ready   master    7m19s   v1.19.0+9f84db3
master02  Ready   master    7m18s   v1.19.0+9f84db3
master03  Ready   master    7m19s   v1.19.0+9f84db3
```

The `oc get events` command prints the OpenShift event messages in real-time, so it is useful for monitoring installation progress and background events in detail.

```
[user@demo ~]$ oc get events -A -w
...output omitted...
openshift-infra      0s      Warning NetworkNotReady
```

Troubleshooting OpenShift Installations

The OpenShift cluster deployment process has three main stages:

- **Bootstrap (Bootkube) Stage**

The OpenShift installation process deploys the bootstrap node. Next, the `release-image` systemd service running on the bootstrap node downloads the container images required to start the temporary control plane. Finally, `bootkube` systemd service running on the bootstrap node starts the temporary control plane.

- **Bootstrap (Temporary Control Plane) Stage**

The Kubernetes API and the temporary control plane are running on the bootstrap node. Next, the bootstrap node waits until the control plane nodes boot and form an etcd cluster. Finally, the bootstrap node schedules the production control plane to the control plane nodes.

- **Production Control Plane stage**

After the production control plane runs on the control plane nodes, the cluster version operator (CVO) finishes the OpenShift cluster deployment.

For troubleshooting installation issues, you must use different procedures depending on the installation stage. You can determine the stage by examining the OpenShift installation logs.

Troubleshooting Bootstrap (Bootkube) Stage

At this stage, the Kubernetes API is not yet available on the bootstrap node. For troubleshooting, you must get the logs from the bootstrap node using SSH. You can log in to the bootstrap node using SSH from the bastion host and inspect the journal and container logs. One of the prerequisites before installing OpenShift is generating an SSH key (cluster SSH key) for enabling SSH access from the bastion host to the cluster nodes.



Warning

Do not run the `ssh-keygen` command at this stage because it will overwrite the cluster SSH key. You will not have access to the cluster nodes with the new cluster SSH key. You must execute the `ssh-keygen` command before running the OpenShift installation.

```
[user@demo ~]$ ssh-keygen -t rsa -b 4096 -N '' -f ${HOME}/.ssh/ocp4-cluster
```

Before running the cluster deployment, you must include the cluster SSH public key `ocp4-cluster.pub` in the `install-config.yaml` file. After you install the cluster nodes, the cluster SSH key will give you SSH access from the bastion host. To troubleshoot the bootstrap node, run the `ssh` command from the bastion host to connect to the bootstrap node. The following example assumes that `192.168.50.9` is the bootstrap node IP address. Use the cluster SSH private key to connect to the bootstrap node from the bastion host:

```
[user@demo ~]$ ssh -i ${HOME}/.ssh/ocp4-cluster core@192.168.50.9
```

The bootstrap node runs two systemd services, the `release-image` service and the `bootkube` service.

- The `release-image` service downloads the OpenShift release container images used during the installation.
- The `bootkube` service orchestrates the bootstrap stage steps.

After logging in to the bootstrap node, run the following command to debug both services:

```
sh-4.2# journalctl -b -f -u release-image.service -u bootkube.service
```

Also, run the `cricctl` command on the bootstrap node to look for failed containers and print their logs:

```
sh-4.2# sudo bash  
sh-4.2# crictl ps -a  
sh-4.2# crictl logs <container_id>
```

Typical errors at this stage include:

- The bootstrap node cannot download container images from `quay.io` or the local registry due to authentication or networking issues.
- The cluster nodes cannot obtain the IP address of the Kubernetes API due to DNS issues.

Troubleshooting Bootstrap (Temporary Control Plane) Stage

At this stage, the Kubernetes API and the temporary control plane are running on the bootstrap node.

For troubleshooting, you must get the logs from the bootstrap node. You can use SSH to get logs from the bootstrap node, as explained elsewhere. Because the Kubernetes API is available on the bootstrap node, you can also use the `openshift-install` and `oc` commands to obtain the bootstrap and control plane nodes logs.

```
[user@demo ~]$ export KUBECONFIG=${HOME}/ocp4-cluster/auth/kubeconfig
```

```
[user@demo ~]$ openshift-install gather bootstrap \  
> --dir=${HOME}/ocp4-cluster  
INFO Pulling debug logs from the bootstrap machine  
INFO Bootstrap gather logs captured here "/home/user/ocp4-cluster/log-  
bundle-20210107135825.tar.gz"
```

```
[user@demo ~]$ cd ocp4-cluster  
[user@demo ~]$ tar -xvzf log-bundle-20210107135825.tar.gz  
[user@demo ~]$ cd log-bundle-20210107135825  
[user@demo ~]$ ls  
bootstrap/ control-plane/ failed-units.txt rendered-assets/ resources/ unit-  
status/
```

For troubleshooting the bootstrap systemd services, use the log files stored in the `bootstrap/journals/` directory:

```
[user@demo ~]$ ls bootstrap/journals/  
approve-csr.log bootkube.log crio-configure.log crio.log ironic.log  
kubelet.log master-update-bmh.log release-image.log
```

For troubleshooting the bootstrap containers, use the log files stored in the `bootstrap/containers/` directory:

```
[user@demo ~]$ ls bootstrap/containers/
...output omitted...
cluster-version-operator-652...f5.inspect
cluster-version-operator-652...f5.log
...output omitted...
```

For troubleshooting the control plane nodes, use the log files stored in the `control-plane/` directory:

```
[user@demo ~]$ find control-plane/
control-plane/
control-plane/10.0.203.117
control-plane/10.0.203.117/unit-status
control-plane/10.0.203.117/journals
control-plane/10.0.203.117/journals/kubelet.log
control-plane/10.0.203.117/journals/crio.log
...output omitted...
```

**Note**

When using the pre-existing infrastructure installation method, you must specify the IP addresses of the bootstrap and control plane nodes when running the `openshift-install gather bootstrap` command.

```
[user@demo ~]$ openshift-install gather bootstrap \
> --dir=${HOME}/ocp4-cluster \
> --bootstrap <bootstrap_address> \
> --master <master_1_address> \
> --master <master_2_address> \
> --master <master_3_address>"
```

You can also access the Kubernetes API running on the bootstrap node and monitor the cluster installation progress. Use the standard `oc` commands from the bastion host for troubleshooting.

```
[user@demo ~]$ export KUBECONFIG=${HOME}/ocp4-cluster/auth/kubeconfig
```

```
[user@demo ~]$ oc get nodes
NAME      STATUS    ROLES     AGE      VERSION
master01   Ready     master    2m35s   v1.19.0+9f84db3
master02   Ready     master    2m34s   v1.19.0+9f84db3
master03   Ready     master    2m35s   v1.19.0+9f84db3
```

```
[user@demo ~]$ oc get clusterversion
NAME      VERSION AVAILABLE PROGRESSING SINCE STATUS
version    False      True       8m32s  Unable to apply 4.6.4: an unknown
error has occurred: MultipleErrors
```

Chapter 1 | Describing the OpenShift Installation Process

```
[user@demo ~]$ oc get clusteroperators
NAME          VERSION AVAILABLE PROGRESSING DEGRADED SINCE
authentication
cloud-credential   True     False    False   10m
console
csi-snapshot-controller
dns
etcd           4.6.4  Unknown  Unknown  False   1s
...output omitted...
```

You can follow the installation process in detail by watching the cluster events.

```
[user@demo ~]$ oc get events -A -w
...output omitted...
openshift-infra      0s      Warning  NetworkNotReady
```

At this stage, you can also use the `oc adm node-logs` command to gather the cluster node logs.

```
[user@demo ~]$ oc adm node-logs -u crio master01
...output omitted...
Jan 27 18:25:57.423858 master01 crio[1627]: time="2021-01-27 18:25:57.423599752Z"
  level=info msg="Checking image status: quay.io/openshift-release-dev/ocp-
v4.0-art-dev@sha256:009...fd5" id=39...fe name=/runtime.v1alpha2.ImageService/
ImageStatus
```

```
[user@demo ~]$ oc adm node-logs -u kubelet master01
...output omitted...
Jan 27 18:31:13.904091 master01 hyperkube[1661]: I0127 18:31:13.903991      1661
  prober.go:126] Readiness probe for "etcd-master01_openshift-etcd(63...40):etcd"
  succeeded
```

```
[user@demo ~]$ oc adm node-logs master01
...output omitted...
Jan 27 18:32:47.487425 master01 hyperkube[1661]: I0127 18:32:47.487445      1661
  kubelet.go:1914] SyncLoop (housekeeping)
Jan 27 18:32:47.536556 master01 hyperkube[1661]: I0127 18:32:47.536513      1661
  exec.go:60] Exec probe response: ""
Jan 27 18:32:47.536556 master01 hyperkube[1661]: I0127 18:32:47.536546      1661
  prober.go:126] Readiness probe for "ovs-z57pn_openshift-sdn(9e...2c):openvswitch"
  succeeded
```

**Note**

The OpenShift cluster nodes run very few local services because most of the system services run as containers. The main exceptions are the `cri-o` container engine and the `kubelet`, which are `systemd` service units.

Also, use the `oc debug` command to create a debug session on the control plane nodes for troubleshooting.

```
[user@demo ~]$ export KUBECONFIG=${HOME}/ocp4-cluster/auth/kubeconfig
```

```
[user@demo ~]$ oc get nodes
NAME      STATUS    ROLES     AGE      VERSION
master01   Ready     master    2m35s   v1.19.0+9f84db3
master02   Ready     master    2m34s   v1.19.0+9f84db3
master03   Ready     master    2m35s   v1.19.0+9f84db3
```

```
[user@demo ~]$ oc debug node/master01
...output omitted...
sh-4.2#
```

Once you log in, run the following command to debug the node logs:

```
sh-4.2# chroot /host
sh-4.2# journalctl -f
```

Also, use the command `cricctl` on the control plane node to search for failed containers and get their logs:

```
sh-4.2# sudo bash
sh-4.2# crictl ps -a
sh-4.2# crictl logs <container_id>
```



Note

Red Hat does not recommend using SSH to access the cluster nodes after the installation has finished. During the installation, if you are not able to use the `oc debug` command, then you can access the control plane nodes using SSH from the bastion host. The following example assumes that `192.168.50.10` is the IP address for one of the control plane nodes. Use the cluster SSH private key to connect to that control plane node from the bastion host.

```
[user@demo ~]$ ssh -i ${HOME}/.ssh/ocp4-cluster core@192.168.50.9
```

Typical errors at this stage include:

- Control plane nodes installation issues
- DNS resolution issues

Troubleshooting the Production Control Plane Stage

At this stage, the Kubernetes API and the production control plane are running on the control plane nodes. The cluster version operator (CVO) running on the production control plane installs all the operators that build the OpenShift cluster and finish the installation. You can apply the same troubleshooting techniques used in the previous section. Typical errors at this stage include:

- OpenShift operators installation issues

Verifying OpenShift Installations

After the OpenShift installation finishes successfully, administrators must ensure that the installed cluster is healthy and ready for day-2 tasks to onboard users and applications.

OpenShift Cluster Health

From the bastion host, you can perform a basic health check using the `oc` command.

- Configure the `KUBECONFIG` environment variable to authenticate against the Kubernetes API with `cluster-admin` permissions.

```
[user@demo ~]$ export KUBECONFIG=${HOME}/ocp4-cluster/auth/kubeconfig
```

- Verify that all the cluster nodes have their system clock synchronized with a Network Time Protocol (NTP) server.

```
[user@demo ~]$ oc debug node/master01
...output omitted...
sh-4.4# chroot /host
sh-4.4# cat /etc/chrony.conf
# Use public servers from the pool.ntp.org project.
# Please consider joining the pool (http://www.pool.ntp.org/join.html).
pool 2.rhel.pool.ntp.org iburst
...output omitted...

sh-4.4# sudo chronyc tracking
Reference ID      : 8AEC8070 (time.gac.edu)
Stratum          : 3
Ref time (UTC)   : Thu Feb 11 13:06:57 2021
System time      : 0.000034756 seconds fast of NTP time
Last offset      : -0.000001187 seconds
RMS offset       : 0.004707427 seconds
Frequency        : 28.194 ppm fast
Residual freq    : -0.000 ppm
Skew             : 0.136 ppm
Root delay       : 0.052070152 seconds
Root dispersion  : 0.018801220 seconds
Update interval  : 64.9 seconds
Leap status       : Normal
```

The `chronyd` systemd service running on the cluster node uses the NTP pool `2.rhel.pool.ntp.org`. The system clock is synchronized with the NTP server `time.gac.edu`.

Repeat this procedure on all the cluster nodes.

- Verify that all the cluster nodes are in a `Ready` status.

If a cluster node is not in a `Ready` status, it cannot communicate with the OpenShift control plane and is unavailable to the cluster.

```
[user@demo ~]$ oc get nodes
NAME      STATUS    ROLES     AGE      VERSION
master01  Ready     master    15h     v1.19.0+9f84db3
master02  Ready     master    15h     v1.19.0+9f84db3
master03  Ready     master    15h     v1.19.0+9f84db3
worker01  Ready     worker   15h     v1.19.0+9f84db3
worker02  Ready     worker   15h     v1.19.0+9f84db3
```

- Check that all the cluster nodes are reporting usage metrics.

```
[user@demo ~]$ oc adm top node
NAME      CPU(cores)    CPU%    MEMORY(bytes)    MEMORY%
master01  677m          19%    4747Mi          31%
master02  391m          11%    3300Mi          22%
master03  519m          14%    4037Mi          27%
worker01  273m          7%    2435Mi          35%
worker02  313m          8%    2906Mi          42%
```

- Ensure that there are no certificate signing requests (CSRs) pending approval.

```
[user@demo ~]$ oc get csr | grep Pending
```

- Confirm that the cluster version operator report shows that the OpenShift cluster is available and ready.

```
[user@demo ~]$ oc get clusterversion
NAME      VERSION    AVAILABLE  PROGRESSING  SINCE      STATUS
version  4.6.4      True       False        22h        Cluster version is 4.6.4
```

- Check that all the cluster operators are available and ready.

If the cluster is healthy, all the cluster operators should be available and not progressing unless one or more operators are still applying the configuration.

```
[user@demo ~]$ oc get clusteroperators
NAME                  VERSION AVAILABLE PROGRESSING DEGRADED SINCE
authentication        4.6.4   True     False     False   22h
cloud-credential     4.6.4   True     False     False   22h
cluster-autoscaler   4.6.4   True     False     False   22h
config-operator       4.6.4   True     False     False   22h
console              4.6.4   True     False     False   22h
...output omitted...
```

- Verify that there are not any pods with scheduling or execution issues in the cluster.

```
[user@demo ~]$ oc get pods --all-namespaces | grep -v -E 'Running|Completed'
NAMESPACE  NAME    READY  STATUS    RESTARTS  AGE
```

OpenShift Etcd Health

- Ensure that all the etcd cluster members are healthy.

```
[user@demo ~]$ oc get pods -n openshift-etcd | grep etcd-master
etcd-master01 3/3 Running 0 22h
etcd-master02 3/3 Running 0 22h
etcd-master03 3/3 Running 0 22h
```

```
[user@demo ~]$ oc rsh -n openshift-etcd etcd-master01
sh-4.4# etcdctl endpoint health --cluster
https://192.168.50.10:2379 is healthy: successfully committed proposal:
took=10.8ms
https://192.168.50.12:2379 is healthy: successfully committed proposal:
took=11.8ms
https://192.168.50.11:2379 is healthy: successfully committed proposal:
took=12.1ms
```

OpenShift API and Console Health

- Verify that the OpenShift API DNS record `api.ocp4.example.com` is configured to use the external load balancer IP address `192.168.50.254`.

```
[user@demo ~]$ dig api.ocp4.example.com
...output omitted...
;; QUESTION SECTION:
;api.ocp4.example.com. IN A

;; ANSWER SECTION:
api.ocp4.example.com. 85333 IN A 192.168.50.254

;; Query time: 0 msec
;; SERVER: 172.25.250.254#53(172.25.250.254)
;; WHEN: Thu Jan 28 05:11:46 EST 2021
;; MSG SIZE rcvd: 71
```

- Verify that the OpenShift API is available by requesting the Kubernetes version.

```
[user@demo ~]$ curl -k https://api.ocp4.example.com:6443/version
...output omitted...
"gitVersion": "v1.19.0+9f84db3",
...output omitted...
```

- Check that you can connect to the OpenShift Console.

```
[user@demo ~]$ curl -kis \
> https://console-openshift-console.apps.ocp4.example.com
...output omitted...
HTTP/1.1 200 OK
...output omitted...
```

```
[user@demo ~]$ firefox https://console-openshift-console.apps.ocp4.example.com
```

OpenShift Registry Health

- Ensure that the number of internal registry pods running on the OpenShift cluster matches its deployment configuration.

```
[user@demo ~]$ oc -n openshift-image-registry get deployment.apps/image-registry
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
image-registry  2/2     2            2           24h
```

- If there are multiple compute nodes, verify that each registry pod is running on a different compute node.

```
[user@demo ~]$ oc -n openshift-image-registry get pods -o wide
...output omitted...
NAME                  READY STATUS    RESTARTS AGE IP          NODE
image-registry-69d88984fb-tjpk1 1/1   Running   0        16m 10.128.2.32 worker02
image-registry-59b67d44f6-n7wkq 1/1   Running   0        16m 10.131.2.12 worker01
...output omitted...
```

- From any cluster node, check the internal registry health.

```
[user@demo ~]$ oc debug node/master01
...output omitted...
sh-4.4# chroot /host
sh-4.4# curl -kIs \
> https://image-registry.openshift-image-registry.svc:5000/healthz
...output omitted...
HTTP/2 200
...output omitted...
```

- Verify that the internal registry deployment is using persistent storage. Also, ensure that the image registry operator is in the Managed management state.

```
[user@demo ~]$ oc get configs.imageregistry.operator.openshift.io cluster -o yaml
...output omitted...
spec:
  managementState: Managed
  ...output omitted...
  storage:
    pvc:
      claim: registry-claim
  ...output omitted...
```

OpenShift Ingress Health

- Verify that the wildcard DNS record for applications, *.apps.ocp4.example.com, is configured to use the external load balancer IP address 192.168.50.254.

```
[user@demo ~]$ dig test.apps.ocp4.example.com
...output omitted...
;; QUESTION SECTION:
;test.apps.ocp4.example.com. IN A
```

Chapter 1 | Describing the OpenShift Installation Process

```
;; ANSWER SECTION:  
test.apps.ocp4.example.com. 86358 IN A 192.168.50.254  
  
;; Query time: 0 msec  
;; SERVER: 172.25.250.254#53(172.25.250.254)  
;; WHEN: Thu Jan 28 05:11:46 EST 2021  
;; MSG SIZE rcvd: 71
```

- Check that you can access an application exposed by an OpenShift Ingress route.

```
[user@demo ~]$ oc get routes -A | grep downloads  
openshift-console downloads downloads-openshift-console.apps.ocp4.example.com
```

```
[user@demo ~]$ curl -kIs \  
> https://downloads-openshift-console.apps.ocp4.example.com  
...output omitted...  
HTTP/1.0 200 OK  
...output omitted...
```

- Ensure that the number of router pods running on the OpenShift cluster matches its deployment configuration.

```
[user@demo ~]$ oc -n openshift-ingress get deployment.apps/router-default  
NAME READY UP-TO-DATE AVAILABLE AGE  
router-default 2/2 2 2 24h
```

- If there are multiple compute nodes, verify that each router pod is running on a different compute node.

```
[user@demo ~]$ oc -n openshift-ingress get pods -o wide | grep router  
NAME READY STATUS RESTARTS AGE IP NODE  
router-default-b7567-l2z4d 1/1 Running 1 16h 192.168.50.13 worker01  
router-default-b7567-qf8x4 1/1 Running 1 16h 192.168.50.14 worker02
```

OpenShift Dynamic Storage Provider Health

When installing OpenShift in a supported cloud provider, the installer configures a dynamic storage provider. In this case, you must verify the status of the dynamic storage provider.

During the OpenShift installation on AWS using the full-stack automation method, the installer configures an AWS EBS dynamic storage provider. This dynamic storage provider uses the aws-ebs storage provisioner.

The OpenShift installation process creates a storage class named gp2 that uses the AWS EBS dynamic storage provider as the back end. The gp2 storage class dynamically provisions persistent storage for the containerized applications running on the OpenShift cluster. The OpenShift installation process configures the gp2 storage class as the default storage class. Unless you specify a different storage class in the PVC definition, any PVC request will use the gp2 storage class to create and bind the PV dynamically.

- Check the AWS EBS gp2 storage class status.

```
[user@demo ~]$ oc get sc
NAME          PROVISIONER      RECLAIMPOLICY BINDINGMODE      EXPANSION AGE
gp2 (default)  kubernetes.io/aws-ebs  Delete  WaitForFirstConsumer true   32m
gp2-csi       ebs.csi.aws.com    Delete  WaitForFirstConsumer true   32m
```

The gp2 storage class uses the `WaitForFirstConsumer` volume binding mode. This volume binding mode delays the binding and provisioning of a `PersistentVolume` until a pod using the `PersistentVolumeClaim` is created. This configuration is immutable for this storage class.

- Verify that the gp2 storage class works as expected.

Create a simple httpd application that uses persistent storage for its `DocumentRoot` directory at `/var/www/html`.

```
[user@demo ~]$ oc new-project httpd-persistent
...output omitted...
[user@demo ~]$ cat /tmp/httpd-persistent.yaml
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: httpd-claim
  namespace: httpd-persistent
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 3Gi
  ---
  apiVersion: v1
  kind: Pod
  metadata:
    name: httpd
    namespace: httpd-persistent
  spec:
    containers:
    - image: registry.redhat.io/rhel8/httpd-24:latest
      name: httpd
      ports:
      - containerPort: 8080
        name: http
        protocol: TCP
      volumeMounts:
      - mountPath: /var/www/html
        name: httpd-claim
    volumes:
    - name: httpd-claim
    persistentVolumeClaim:
      claimName: httpd-claim
```

Chapter 1 | Describing the OpenShift Installation Process

```
[user@demo ~]$ oc create -f /tmp/httpd-persistent.yaml
persistentvolumeclaim/httpd-claim created
pod/httpd created
```

Verify that the PVC creation automatically triggers the PV provisioning and binding through the gp2 default storage class.

```
[user@demo ~]$ oc get pvc
NAME          STATUS    VOLUME      CAPACITY   ACCESS MODES  STORAGECLASS  AGE
httpd-claim   Bound     pvc-d965cb1f  3Gi        RWO          gp2           29s
```

```
[user@demo ~]$ oc rsh httpd
sh-4.4$ df -h
Filesystem      Size  Used Avail Use% Mounted on
...output omitted...
/dev/nvme2n1    2.9G  9.0M  2.9G   1% /var/www/html
...output omitted...
```

OpenShift Application Build and Deployment Test

- Build and deploy a test application to verify the OpenShift application build cycle.

```
[user@demo ~]$ oc new-project validate  
...output omitted...  
[user@demo ~]$ oc new-app django-psql-example  
...output omitted...
```

```
[user@demo ~]$ oc get pods -n validate  
NAME                      READY   STATUS    RESTARTS   AGE  
django-psql-example-1-build   0/1     Completed  0          11m  
django-psql-example-1-deploy  0/1     Completed  0          10m  
django-psql-example-1-vfb5l   1/1     Running   0          10m  
postgresql-1-bdgkk           1/1     Running   0          11m  
postgresql-1-deploy          0/1     Completed  0          11m
```

```
[user@demo ~]$ oc logs -f django-psql-example-1-build  
...output omitted...  
Successfully pushed image-registry.openshift-image-registry.svc:5000/validate/  
django-psql-example@sha256:b97b...ff82  
Push successful
```

```
[user@demo ~]$ oc logs -f django-psql-example-1-deploy  
...output omitted...  
--> Scaling django-psql-example-1 to 1  
--> Success
```

```
[user@demo ~]$ oc get routes -n validate  
NAME              HOST/PORT          PATH  
SERVICES          PORT  TERMINATION WILDCARD  
django-psql-example django-psql-example-validate.apps.ocp4.example.com  
django-psql-example <all>           None
```

```
[user@demo ~]$ curl -Is \  
> django-psql-example-validate.apps.ocp4.example.com  
...output omitted...  
HTTP/1.1 200 OK  
...output omitted...
```

```
[user@demo ~]$ firefox http://django-psql-example-validate.apps.ocp4-  
aws.example.com
```

OpenShift Cluster Network

- Verify the OpenShift cluster network configuration.

```
[user@demo ~]$ oc get network.config/cluster -o yaml
apiVersion: config.openshift.io/v1
kind: Network
metadata:
...output omitted...
status:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
    clusterNetworkMTU: 8142
    networkType: OpenShiftSDN
    serviceNetwork:
      - 172.30.0.0/16
```

OpenShift Etcd Storage Performance

- Verify the etcd storage performance.

```
[user@demo ~]$ oc get pods -n openshift-etcd | grep etcd-master
etcd-master01 3/3 Running 0 22h
etcd-master02 3/3 Running 0 22h
etcd-master03 3/3 Running 0 22h
```

```
[user@demo ~]$ oc rsh -n openshift-etcd etcd-master01
...output omitted...
sh-4.4# etcdctl check perf --load="s"
60 / 60 Booooooooooooooooooooooo! 100.00% 1m0s
PASS: Throughput is 150 writes/s
PASS: Slowest request took 0.220329s
PASS: Stddev is 0.018010s
PASS

sh-4.4# etcdctl check perf --load="m"
60 / 60 Booooooooooooooo! 100.00% 1m0s
PASS: Throughput is 964 writes/s
PASS: Slowest request took 0.379547s
PASS: Stddev is 0.022218s
PASS

sh-4.4# etcdctl check perf --load="l"
60 / 60 Booooooooooooooo! 100.00% 1m0s
```

Chapter1 | Describing the OpenShift Installation Process

```
FAIL: Throughput too low: 4586 writes/s
PASS: Slowest request took 0.258474s
PASS: Stddev is 0.032695s
FAIL
```

From the test result, the etcd cluster performs well for a medium cluster (`--load="m"`) and fails for a large cluster (`--load="l"`). For more information about etcd performance, visit the etcd documentation page at <https://etcd.io/docs/current/op-guide/hardware/>

For more detailed etcd storage performance information, use the `fio` tool from the `etcd-perf` container to run a performance test on the control plane nodes. The performance test output reports whether the disk is fast enough to host etcd by comparing the 99th percentile of the `fsync` metric captured from the run to see if it is less than **10 ms**.

```
[user@demo ~]$ oc debug node/master01
...output omitted...
sh-4.4# chroot /host
sh-4.4# podman run --volume /var/lib/etcd:/var/lib/etcd:z
quay.io/openshift-scale/etcd-perf
...output omitted...
{
  "fio version" : "fio-3.7",
...output omitted...
  "global options" : {
    "rw" : "write",
    "ioengine" : "sync",
    "fdatasync" : "1",
    "directory" : "/var/lib/etcd",
    "size" : "22m",
    "bs" : "2300"
  },
...output omitted...
  "write" : {
    "iops" : 328.808892,
...output omitted..
  },
...output omitted...
}
-----
99th percentile of fsync is 6193152 ns
99th percentile of the fsync is within the recommended threshold - 10 ms, the disk can be used to host etcd
```

The `fio` performance test produces the following result:

1. This test writes 22 MiB of data in blocks of 2300 bytes on the `/var/lib/etcd` directory.
2. The 99th percentile of the `fsync` is 6193152 ns, which is equivalent to **6 ms** of write latency.
3. The operating system has achieved an average of **328 IOPS** during the test.

OpenShift Machine API

When installing OpenShift in a supported cloud provider, the installer configures the compute node autoscaling using the OpenShift Machine API component. In this case, you must verify the status of the compute node autoscaling. One of the essential advantages of using the full-stack automation installation method on AWS is that the OpenShift installation process configures the compute node autoscaling.

The OpenShift Machine API is the component that defines and manages the OpenShift Machines resource. The OpenShift Machines resource represents the OpenShift cluster nodes. The OpenShift Machine API:

- Creates, updates, and deletes Machines
- Creates the infrastructure (instance or VM) for the node

You can use the OpenShift MachineSets resource to control sets of Machines. A Machineset represents:

- A set of Machines
- An abstraction of the underlying infrastructure

When installing OpenShift on AWS using the full-stack automation method, the OpenShift installer creates and configures a MachineSet for each availability zone in the selected region.

[user@demo ~]\$ oc get machines -n openshift-machine-api					
NAME	PHASE	TYPE	REGION	ZONE	AGE
ocp4-aws-9r678-master-0	Running	m5.2xlarge	us-east-2	us-east-2a	16h
ocp4-aws-9r678-master-1	Running	m5.2xlarge	us-east-2	us-east-2b	16h
ocp4-aws-9r678-master-2	Running	m5.2xlarge	us-east-2	us-east-2c	16h
ocp4-aws-9r678-worker-us-east-2a-gq2ps	Running	m5.4xlarge	us-east-2	us-east-2a	16h
ocp4-aws-9r678-worker-us-east-2b-slp7l	Running	m5.4xlarge	us-east-2	us-east-2b	16h
ocp4-aws-9r678-worker-us-east-2c-vj7pj	Running	m5.4xlarge	us-east-2	us-east-2c	16h

[user@demo ~]\$ oc get machinesets -n openshift-machine-api					
NAME	DESIRED	CURRENT	READY	AVAILABLE	AGE
ocp4-aws-9r678-worker-us-east-2a	1	1	1	1	16h
ocp4-aws-9r678-worker-us-east-2b	1	1	1	1	16h
ocp4-aws-9r678-worker-us-east-2c	1	1	1	1	16h

[user@demo ~]\$ oc get nodes --label-columns \ > failure-domain.beta.kubernetes.io/region,failure-domain.beta.kubernetes.io/zone					
NAME	REGION	ZONE	STATUS	ROLES	AGE
ip-10-0-151-177.us-east-2.compute.internal	v1.19.0+9f84db3	us-east-2	Ready	master	16h

Chapter1 | Describing the OpenShift Installation Process

ip-10-0-157-4.us-east-2.compute.internal v1.19.0+9f84db3 us-east-2 us-east-2a	Ready	worker	16h
ip-10-0-166-182.us-east-2.compute.internal v1.19.0+9f84db3 us-east-2 us-east-2b	Ready	worker	16h
ip-10-0-180-27.us-east-2.compute.internal v1.19.0+9f84db3 us-east-2 us-east-2b	Ready	master	17h
ip-10-0-205-233.us-east-2.compute.internal v1.19.0+9f84db3 us-east-2 us-east-2c	Ready	master	17h
ip-10-0-217-153.us-east-2.compute.internal v1.19.0+9f84db3 us-east-2 us-east-2c	Ready	worker	16h

Using the worker MachineSets, you can scale up (or down) the number of compute nodes running on the cluster. When scaling up a worker MachineSet:

- The OpenShift Machine API automatically provisions and starts an AWS EC2 instance for the new compute node.
- The new compute node gets its ignition configuration file and installs RHCOS.
- The new compute node joins the OpenShift cluster automatically.

```
[user@demo ~]$ oc scale machineset ocp4-aws-9r678-worker-us-east-2c \
> --replicas=2 -n openshift-machine-api
machineset.machine.openshift.io/ocp4-aws-9r678-worker-us-east-2c scaled
```

NAME	DESIRED	CURRENT	READY	AVAILABLE	AGE
ocp4-aws-9r678-worker-us-east-2a	1	1	1	1	17h
ocp4-aws-9r678-worker-us-east-2b	1	1	1	1	17h
ocp4-aws-9r678-worker-us-east-2c	2	2	1	1	17h

After a few minutes, the new compute node must be in Ready status.

NAME	DESIRED	CURRENT	READY	AVAILABLE	AGE
ocp4-aws-9r678-worker-us-east-2a	1	1	1	1	17h
ocp4-aws-9r678-worker-us-east-2b	1	1	1	1	17h
ocp4-aws-9r678-worker-us-east-2c	2	2	2	2	17h

```
[user@demo ~]$ oc get machines -n openshift-machine-api
...output omitted..
ocp4-aws-9r678-worker-us-east-2c-65hln     Running   m5.4xlarge   us-east-2
us-east-2c    3m41s
ocp4-aws-9r678-worker-us-east-2c-vj7pj     Running   m5.4xlarge   us-east-2   us-
east-2c   17h
```

```
[user@demo ~]$ oc get nodes --label-columns \
> failure-domain.beta.kubernetes.io/region,failure-domain.beta.kubernetes.io/zone
...output omitted..
ip-10-0-196-32.us-east-2.compute.internal     Ready   worker   2m31s
  v1.19.0+9f84db3    us-east-2   us-east-2c
ip-10-0-205-233.us-east-2.compute.internal     Ready   master   17h
  v1.19.0+9f84db3    us-east-2   us-east-2c
```

As you can see in the preceding example, the OpenShift Machine API automatically provisioned and added a new compute node to the cluster (`ip-10-0-196-32.us-east-2.compute.internal`) on the desired `us-east-2c` AWS AZ.

Gathering OpenShift Data

When interacting with Red Hat Support to solve any OpenShift issue, administrators are asked to provide cluster debug information. Depending on the OpenShift component to troubleshoot, administrators can use different debug mechanisms.

OpenShift Cluster Data

You can gather cluster debug information using the `oc adm must-gather` CLI command as the `cluster-admin` user. This CLI command collects the information from your cluster, such as:

- Resource definitions
- Audit logs
- Service logs

The execution of the `oc adm must-gather` command creates a new pod on the cluster. That pod collects the cluster data and stores it in a new directory. The new directory name starts with `must-gather.local`. The `oc adm must-gather` command creates this directory in the current working directory.

```
[user@demo ~]$ export KUBECONFIG=${HOME}/ocp4-cluster/auth/kubeconfig
```

```
[user@demo ~]$ cd ${HOME}
[user@demo ~]$ oc adm must-gather
[must-gather      ] OUT Using must-gather plugin-in image: quay.io/openshift-
release-dev/ocp-v4.0-art-dev@sha256:47..86
[must-gather      ] OUT namespace/openshift-must-gather-ndwcz created
[must-gather      ] OUT clusterrolebinding.rbac.authorization.k8s.io/must-
gather-5ptk2 created
[must-gather      ] OUT pod for plug-in image quay.io/openshift-release-dev/ocp-
v4.0-art-dev@sha256:47..86 created
[must-gather-276db] POD Wrote inspect data to must-gather.
...output omitted...
```

```
[user@demo ~]$ ls
install-config.yaml  must-gather.local.1227184995617480385/  ocp4-cluster/
```

```
[user@demo ~]$ find must-gather.local.1227184995617480385/
must-gather.local.1227184995617480385/
must-gather.local.1227184995617480385/timestamp
...output omitted...
```

As you can see, this directory contains the OpenShift resources definitions, services logs, and audit logs.

**Note**

When opening an OpenShift support case in the Red Hat Customer Portal, create a tar file with the output generated by the `oc adm must-gather` execution and attach it to the support case.

```
[user@demo ~]$ tar cvaf must-gather.tar.gz \
> must-gather.local.1227184995617480385/
...output omitted...
```

You can gather debug information about specific features using the `oc adm must-gather` CLI command with the `--image` or `--image-stream` argument.

```
[user@demo ~]$ oc adm must-gather \
> --image-stream=openshift/must-gather \
> --image=registry.redhat.io/container-native-virtualization/\
> cnv-must-gather-rhel8:v2.5.2
...output omitted...
```

For instance, using the `cnv-must-gather-rhel8` image the `oc adm must-gather` command collects OpenShift Virtualization specific data.

Most Commonly Used Must-gather Images

Image	Purpose
registry.redhat.io/container-native-virtualization/cnv-must-gather-rhel8:v2.5.2	Data collection for OpenShift Virtualization
registry.redhat.io/openshift-serverless-1/svls-must-gather-rhel8	Data collection for OpenShift Serverless
registry.redhat.io/openshift-service-mesh/istio-must-gather-rhel7	Data collection for Red Hat OpenShift Service Mesh
registry.redhat.io/rhcam-1-2/openshift-migration-must-gather-rhel8	Data collection for migration-related information
registry.redhat.io/ocs4/ocs-must-gather-rhel8	Data collection for Red Hat OpenShift Container Storage
registry.redhat.io/openshift4/ose-cluster-logging-operator	Data collection for Red Hat OpenShift cluster logging

OpenShift Node Data

In some scenarios, Red Hat Support will ask you to collect a `sosreport` file from a specific OpenShift cluster node. The `sosreport` command is a tool that collects configuration details, system information, and diagnostic data from Red Hat Enterprise Linux (RHEL) and Red Hat Enterprise Linux CoreOS (RHCOS) systems.

Red Hat recommends using a debug pod to generate a `sosreport` from an OpenShift cluster node.

```
[user@demo ~]$ oc debug node/ip-10-0-151-177.us-east-2.compute.internal
...output omitted...
sh-4.4# chroot /host
sh-4.4# toolbox
Trying to pull registry.redhat.io/rhel8/support-tools...
...output omitted...

[root@ip-10-0-151-177 /]# sosreport -k crio.all=on -k crio.logs=on
...output omitted...
Your sosreport has been generated and saved in:
/host/var/tmp/sosreport-ip-10-0-151-177-1234-2021-01-11-crlsmlr.tar.xz

Size   31.07MiB
Owner   root
md5    3ddfb7a774002fc8fb18da9c9c1534bc

Please send this file to your support representative.

[root@ip-10-0-151-177 /]# exit
sh-4.4# exit
sh-4.4# ls -lrt \
> /host/var/tmp/sosreport-ip-10-0-151-177-1234-2021-01-11-crlsmlr.tar.xz
-rw----- 1 root root 32578896 Jan 11 13:42 /host/var/tmp/sosreport-
ip-10-0-151-177-1234-2021-01-11-crlsmlr.tar.xz
```

OpenShift Remote Health Monitoring

OpenShift collects anonymized aggregated information about the health, usage, and size of the clusters. With this information, Red Hat can proactively react to issues that can impact customers.

The OpenShift cluster reports this information to Red Hat using two components:

- Telemetry
- Insights Operator

The Telemetry component sends a chosen subset of the cluster monitoring metrics to Red Hat. These metrics are sent continuously and describe:

- OpenShift cluster size
- OpenShift components health
- OpenShift upgrades health
- Limited OpenShift usage information
- OpenShift alerts summary info

Chapter 1 | Describing the OpenShift Installation Process

The Insights Operator periodically gathers the cluster configuration and component failure status and reports that data to Red Hat. Using this information, the Red Hat OpenShift Cluster Manager proactively identifies potential cluster issues and provides solutions and preventive actions.



Note

For more information, refer to the *Remote health monitoring with connected clusters* guide in the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/support

You can access your cluster information using the Red Hat OpenShift Cluster Manager Console [<https://cloud.redhat.com/openshift>]. If you manage several OpenShift clusters, you will need your `cluster_id` to identify your cluster in the Red Hat OpenShift Cluster Manager Console.

```
[user@demo ~]$ oc get clusterversion \
> -o jsonpath='{.items[].spec.clusterID}{ "\n"}'
9d1c5e73-9deb-452b-b327-376d04315246
```

After retrieving the `cluster_id`, open your web browser and navigate to Red Hat OpenShift Cluster Manager Console [<https://cloud.redhat.com/openshift>] using your Red Hat account. Then click your `cluster_id` link and review your cluster information under the Overview, Monitoring, Insights, and Support navigation tabs.

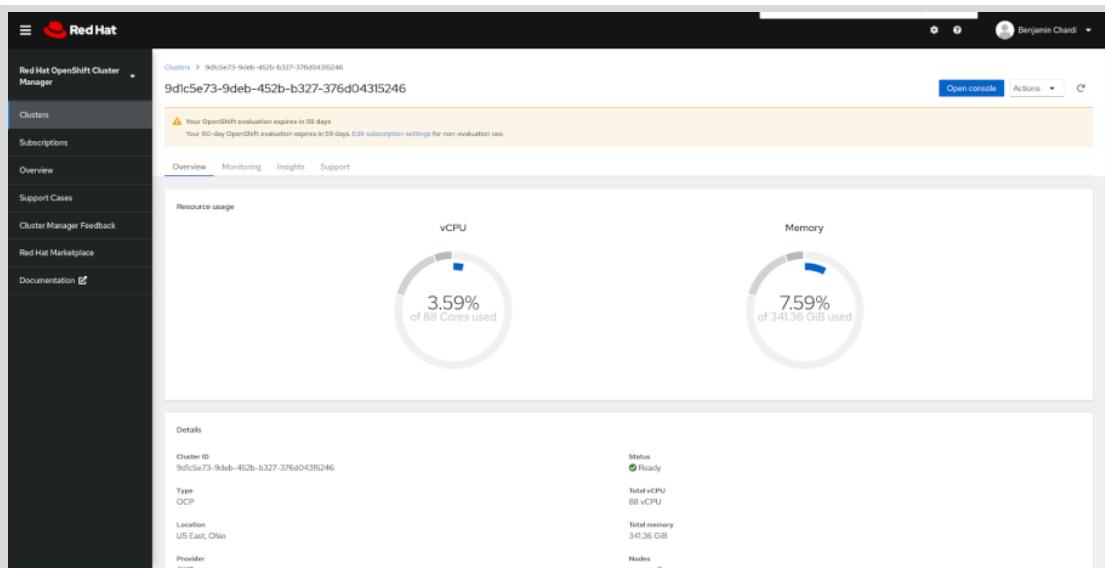


Figure 1.9: Red Hat OpenShift Cluster Manager Console

Deleting an OpenShift Cluster

If you need to remove your OpenShift cluster, then you can run the OpenShift installer using the `destroy cluster` option.

```
[user@demo ~]$ openshift-install destroy cluster \
> --dir=${HOME}/ocp4-cluster
...output omitted...
```



References

- For more information, refer to the *Installation configuration* chapter of the Red Hat OpenShift Container Platform 4.6 documentation at
https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing
- For more information, refer to the *Installing a cluster on AWS with customizations* section of the Red Hat OpenShift Container Platform 4.6 documentation at
https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_aws

► Quiz

Running the OpenShift Installer

Choose the correct answers to the following questions:

- ▶ **1. Choose the correct statement about the step sequence to follow when installing OpenShift with the OpenShift installer.**
 - a. Create ignition configuration files, create Kubernetes manifests, create install-config.yaml, create the cluster.
 - b. Create Kubernetes manifests, create install-config.yaml, create ignition configuration files, create the cluster.
 - c. Create install-config.yaml, create Kubernetes manifests, create ignition configuration files, create the cluster.
 - d. Create install-config.yaml, create ignition configuration files, create Kubernetes manifests, create the cluster.
- ▶ **2. Kubernetes manifests and ignition configuration files are created automatically in the create cluster step, if they do not exist already. (True or False)**
 - a. True
 - b. False
- ▶ **3. Which two of the following practices are recommended when installing OpenShift? (Choose two.)**
 - a. Run the OpenShift installer on a bastion host.
 - b. Setup a minimum of 5 control plane nodes.
 - c. Use your custom Operator container images.
 - d. Ensure that the cluster nodes have access to an NTP server.
- ▶ **4. Which of the following commands can be executed on the bastion host to enable cluster management with the oc command?**
 - a. `export KUBEADMIN=${HOME}/ocp4-aws-cluster/auth/kubeconfig`
 - b. `export KUBECONFIG=${HOME}/ocp4-aws-cluster/auth/kubeconfig`
 - c. `export KUBECONFIG=${HOME}/ocp4-aws-cluster/auth/kubeadmin`

► Solution

Running the OpenShift Installer

Choose the correct answers to the following questions:

► 1. **Choose the correct statement about the step sequence to follow when installing OpenShift with the OpenShift installer.**

- a. Create ignition configuration files, create Kubernetes manifests, create install-config.yaml, create the cluster.
- b. Create Kubernetes manifests, create install-config.yaml, create ignition configuration files, create the cluster.
- c. Create install-config.yaml, create Kubernetes manifests, create ignition configuration files, create the cluster.
- d. Create install-config.yaml, create ignition configuration files, create Kubernetes manifests, create the cluster.

► 2. **Kubernetes manifests and ignition configuration files are created automatically in the create cluster step, if they do not exist already. (True or False)**

- a. True
- b. False

► 3. **Which two of the following practices are recommended when installing OpenShift? (Choose two.)**

- a. Run the OpenShift installer on a bastion host.
- b. Setup a minimum of 5 control plane nodes.
- c. Use your custom Operator container images.
- d. Ensure that the cluster nodes have access to an NTP server.

► 4. **Which of the following commands can be executed on the bastion host to enable cluster management with the oc command?**

- a. `export KUBEADMIN=${HOME}/ocp4-aws-cluster/auth/kubeconfig`
- b. `export KUBECONFIG=${HOME}/ocp4-aws-cluster/auth/kubeconfig`
- c. `export KUBECONFIG=${HOME}/ocp4-aws-cluster/auth/kubeadmin`

► Guided Exercise

Completing the OpenShift Installation Prerequisites

- Prepare the environment for installing OpenShift on pre-existing infrastructure.

Outcomes

You should be able to:

- Download and install the `oc` and `openshift-install` binaries.
- Generate the SSH key for troubleshooting.
- Find out the local registry for your classroom environment region.
- Download and configure the pull secret for using a local registry.
- Verify the pull secret credentials.
- Verify the DNS records for the OpenShift API and APP Ingress load balancers.

Before You Begin

To perform this exercise, ensure that you have completed the *Introducing OpenShift Installation Methods* and *Running the OpenShift Installer* lectures. If you have made any changes to the classroom environment, you must destroy the classroom and recreate it to complete this activity successfully.



Note

The Firefox version in the workstation machine does not render properly the <https://cloud.redhat.com/openshift/> site. To complete this guided exercise, install and use the Chromium web browser in the workstation:

```
[student@workstation ~]$ sudo yum install chromium
```

Instructions

- 1. Install the Chromium web browser from a terminal on the `workstation` machine.

```
[student@workstation ~]$ sudo yum install chromium  
...output omitted...
```

When prompted, type yes and press `Enter` to accept the installation.

- 2. Log in to the `utility` server to download and install the CLI tools.

2.1. Log in to the `utility` server as the `lab` user. Become the `root` user.

```
[student@workstation ~]$ ssh lab@utility  
...output omitted...  
[lab@utility ~]$ sudo -i  
[root@utility ~]#
```

- 2.2. Download the `oc` and `openshift-install` binaries for the 4.6.4 OpenShift version.

```
[root@utility ~]# mirror="https://mirror.openshift.com/pub/openshift-v4/clients"  
[root@utility ~]# wget ${mirror}/ocp/4.6.4/openshift-client-linux-4.6.4.tar.gz  
...output omitted...  
  
[root@utility ~]# wget ${mirror}/ocp/4.6.4/openshift-install-linux-4.6.4.tar.gz  
...output omitted...
```

- 2.3. Extract the `oc` and `openshift-install` binaries and copy them to `/usr/bin`.

```
[root@utility ~]# tar -xvf openshift-client-linux-4.6.4.tar.gz -C /usr/bin/  
README.md  
oc  
kubectl  
  
[root@utility ~]# tar -xvf openshift-install-linux-4.6.4.tar.gz -C /usr/bin/  
README.md  
openshift-install
```

- 2.4. Set the `oc` and `openshift-install` bash completion. Source the bash completion files as the `root` user. Also, source the bash completion files as the `lab` user.

```
[root@utility ~]# oc completion bash > /etc/bash_completion.d/openshift  
  
[root@utility ~]# openshift-install completion \  
> bash > /etc/bash_completion.d/openshift-install  
  
[root@utility ~]# source /etc/bash_completion.d/openshift  
[root@utility ~]# source /etc/bash_completion.d/openshift-install  
[root@utility ~]# exit  
logout  
[lab@utility ~]$ source /etc/bash_completion.d/openshift  
[lab@utility ~]$ source /etc/bash_completion.d/openshift-install
```

- 2.5. As the `lab` user, verify the installation of the `oc` and `openshift-install` binaries.

```
[lab@utility ~]$ oc version  
Client Version: 4.6.4
```

```
[lab@utility ~]$ openshift-install version
openshift-install 4.6.4
built from commit 6e0...6c5
release image quay.io/openshift-release-dev/ocp-release@sha256:66...fc
```

- 3. Create the SSH key for the installation.

```
[lab@utility ~]$ ssh-keygen -t rsa -b 4096 -N '' -f .ssh/ocp4upi
Generating public/private rsa key pair.
Your identification has been saved in .ssh/ocp4upi.
Your public key has been saved in .ssh/ocp4upi.pub.
...output omitted...
```

- 4. Find the local registry FQDN for the region of your classroom environment. You can locate your classroom environment region on the **Lab Environment** tab on the Red Hat OpenShift Installation Lab course web page. Use the Chromium web browser during the rest of this guided exercise.

The following table shows the FQDN of the local registry for each region.

Local Registry FQDN per Classroom Environment Region

Region	Local registry FQDN
northamerica	nexus-registry-int.apps.tools-na150.prod.ole.redhat.com
emea	nexus-registry-int.apps.tools-emea150.prod.ole.redhat.com
apac	nexus-registry-int.apps.tools-apac150.prod.ole.redhat.com

During this course, you will use a local registry containing the OpenShift release images mirrored from `quay.io`. The local registry runs in the same region as the classroom environment, and its fully qualified domain name (FQDN) depends on that region.

- 4.1. Install the Chromium web browser from a terminal on the **workstation** machine.

```
[student@workstation ~]$ sudo yum install chromium
...output omitted...
```

When prompted, type yes and press **Enter** to accept the installation.

- 4.2. Now, find the region of your classroom environment.

From the **workstation** machine, use the Chromium web browser to navigate to the **Lab Environment** tab on the Red Hat OpenShift Installation Lab course web page.

Click **information** to display information about the classroom environment. You can find the classroom environment region in the **Published region** field.

► Lab Controls

Click **CREATE** to build all of the virtual machines needed for the classroom lab environment. This may take several minutes to complete. Once created the environment can then be stopped and restarted to pause your experience.

If you **DELETE** your lab, you will remove all of the virtual machines in your classroom and lose all of your progress.

Lab Information

- Project id: 2ebdcefb42a144ada7e8c1c3fce8b56
- Project name: ole-3030f6f7-9253-47ed-9ea1-73c0aa7e62c9
- Project state: stopped
- Lab definition id: do322ea-4.6
- Published region: northamerica
- openstack region: us-east-1

Actions:

- DELETE**
- START**
- ACTION**
- OPEN CONSOLE**

4.3. Locate the FQDN of the local registry for your region by using the preceding table.

4.4. As the **lab** user on the **utility** server, use the **curl** command to verify that you can communicate with the FQDN of the local registry for your classroom environment.

The following command assumes that the classroom environment region is **northamerica**, which uses the local registry FQDN **nexus-registry-int.apps.tools-na150.prod.ole.redhat.com**.

```
[lab@utility ~]$ curl -s \
> https://nexus-registry-int.apps.tools-na150.prod.ole.redhat.com -o /dev/null;
echo $?
0
```

If you use the wrong local registry FQDN, then the output of the **curl** command will not be **0**.



Note

By default, the local registry of the **northamerica** region **nexus-registry-int.apps.tools-na150.prod.ole.redhat.com** is used for example in all the lectures and guided exercises of this course.

Ensure that you use the local registry FQDN of the region of your classroom environment in all the guided exercises. Using a local registry from other regions will cause the guided exercises to fail.

4.5. Log out from the **utility** server.

```
[lab@utility ~]$ exit
...output omitted...
[student@workstation ~]$
```

► 5. As the **student** user on the **workstation** machine, obtain a pull secret from **cloud.redhat.com**. Replace the credentials for the **quay.io** registry with the credentials for your local registry.

Chapter1 | Describing the OpenShift Installation Process

Assuming that your classroom environment region is `northamerica`, the content of the pull secret for the local registry is:

```
"nexus-registry-int.apps.tools-na150.prod.ole.redhat.com":  
{"auth":"cmVndXNlcjpJbnN0YWxsTTM=", "email":"nobody@example.com"}
```

- 5.1. On the **workstation** machine, use the Chromium web browser to navigate to <https://cloud.redhat.com/openshift/install/metal/user-provisioned>. Log in using your Red Hat account credentials.
- 5.2. Click **Download pull secret**.
- 5.3. Open a terminal on the **workstation** machine and format the `pull-secret.txt` file in the `/home/student/Downloads` folder as JSON:

```
[lab@utility ~]$ exit  
...output omitted...  
[student@workstation ~]$
```

```
[student@workstation ~]$ python3 -m json.tool \  
> Downloads/pull-secret > pull-secret.json
```

```
[student@workstation ~]$ cat pull-secret.json  
{  
    "auths": {  
        "cloud.openshift.com": {  
            "auth":  
"UxUUjYwSTMyb3BlUxUUjYwSTMybnNUxUUjYwSTMyoawZUxUU...YMy3NfNGUxUUjYw==",  
                "email": "student@redhat.com"  
        },  
        "quay.io": {  
  
            "auth": "UxUUjYwSTMyb3BlUxUUjYwSTMybnNUxUUjYwSTMyoawZUxUU...YMy3NfNGUxUUjYw==",  
                "email": "student@redhat.com"  
        },  
        "registry.connect.redhat.com": {  
            "auth":  
"CvmsWaJUROSkhCkEQ71NiM1BsracE9Z0VmBMIDrS3R20KOH8Eq...Tx09phtozeXpqKLJN=",  
                "email": "student@redhat.com"  
        },  
        "registry.redhat.io": {  
            "auth":  
"CvmsWaJUROSkhCkEQ71NiM1BsracE9Z0VmBMIDrS3R20KOH8Eq...Tx09phtozeXpqKLJN=",  
                "email": "student@redhat.com"  
        }  
    }  
}
```

- 5.4. Edit the `pull-secret.json` file to replace the credentials for `quay.io` with the credentials for your local registry.

**Note**

The content of the pull secret for the local registry in the `northamerica` region is:

```
"nexus-registry-int.apps.tools-na150.prod.ole.redhat.com":  
{"auth": "cmVndXNlcjpJbnN0YWxsTTM=", "email": "nobody@example.com"}
```

The values of the `auth` and `email` attributes are the same for all the local registries, regardless of the classroom region.

```
[student@workstation ~]$ vi pull-secret.json  
{  
  "auths": {  
    "cloud.openshift.com": {  
      "auth":  
        "UxUUjYwSTMyb3BlUxUUjYwSTMybnNUxUUjYwSTMyoawZUxUU...YMy3NfNGUXUUjYw==",  
      "email": "student@redhat.com"  
    },  
    "nexus-registry-int.apps.tools-na150.prod.ole.redhat.com": {  
      "auth": "cmVndXNlcjpJbnN0YWxsTTM=",  
      "email": "nobody@example.com"  
    },  
    "registry.connect.redhat.com": {  
      "auth":  
        "CvmsWaJUROSkhCkEQ71NiM1BsracE9Z0VmBMIDrs3R20K0H8Eq...Tx09phTozeXpqKLJN=",  
      "email": "student@redhat.com"  
    },  
    "registry.redhat.io": {  
      "auth":  
        "CvmsWaJUROSkhCkEQ71NiM1BsracE9Z0VmBMIDrs3R20K0H8Eq...Tx09phTozeXpqKLJN=",  
      "email": "student@redhat.com"  
    }  
  }  
}
```

Save the file, and then close the editor.

- 5.5. Use the `jq` tool to generate the compact version of the `pull-secret.json` file in the `pull-secret-oneline.json` file.

```
[student@workstation ~]$ cat pull-secret.json | jq . -c > pull-secret-oneline.json
```

**Note**

The `jq` parsing used in the preceding command also checks that the `pull-secret.json` file is valid. If you do not get any error when running this command, it means that the `pull-secret.json` and `pull-secret-oneline.json` files are valid.

```
[student@workstation ~]$ cat pull-secret-oneline.json
{"auths":{"cloud.openshift.com":
{"auth":"UxUUj...UUjYw==","email":"student@redhat.com"},"nexus-
registry-int.apps.tools-na150.prod.ole.redhat.com":
{"auth":"cmVnd...sTTM=","email":"nobody@example.com"},"registry.connect.redhat.com":
 {"auth":"CvmsW...qKLJN=","email":"student@redhat.com"},"registry.redhat.io":
 {"auth":"CvmsW...qKLJN=","email":"student@redhat.com"}}}
```

5.6. Copy the `pull-secret-oneline.json` file to the utility server.

```
[student@workstation ~]$ scp pull-secret-oneline.json lab@utility:
...output omitted...
pull-secret-oneline.json
```

You must use the `pull-secret-oneline.json` file later on this course to complete the `install-config.yaml` file.

- ▶ 6. As the `lab` user on the `utility` server, verify that the credentials stored in the `~/pull-secret-oneline.json` file are valid to pull images from the local registry and the `registry.redhat.io` registry.
 - 6.1. Install the `podman` tool to pull the OpenShift install release image from the local registry.

```
[student@workstation ~]$ ssh lab@utility
...output omitted...
```

```
[lab@utility ~]$ sudo yum install podman
...output omitted...
Complete!
```

```
[lab@utility ~]$ sudo podman pull --authfile ~/pull-secret-oneline.json \
> nexus-registry-int.apps.tools-na150.prod.ole.redhat.com/openshift/ocp4:4.6.4-x86_64
Trying to pull nexus-registry-int.apps.tools-na150.prod.ole.redhat.com/openshift/
ocp4:4.6.4-x86_64...
Getting image source signatures
Copying blob ec1681b6a383 done
Copying blob c4d668e229cd done
Copying blob 6b8afa2699e1 done
Copying blob 12dcdecc154a done
Copying blob 1cefa0e873d7 done
Copying blob 09bfd7265184 done
Copying config 26f7cd4cf1 done
Writing manifest to image destination
Storing signatures
26f...fc9
```

- 6.2. As the `lab` user on the `utility` server, verify that the credentials stored in the `~/pull-secret-oneline.json` file are valid to pull images from the `registry.redhat.io` registry.

```
[lab@utility ~]$ sudo podman pull --authfile ~/pull-secret-oneline.json \
> registry.redhat.io/ubi8/ubi:latest
Trying to pull registry.redhat.io/ubi8/ubi...
Getting image source signatures
Copying blob d9e72d058dc5 done
Copying blob cca21acb641a done
Copying config 3269c37eae done
Writing manifest to image destination
Storing signatures
326...a5e
```

**Note**

Running the `podman pull` commands with a malformed `pull-secret` file (or with the wrong local registry FQDN) causes the commands to fail. If the `podman pull` command execution fails, review the preceding steps beginning with the step where you find the local registry FQDN for the region of your classroom environment.

6.3. Verify that you have pulled the test images correctly.

```
[lab@utility ~]$ sudo podman images
REPOSITORY                                     TAG
IMAGE ID          CREATED        SIZE
registry.redhat.io/ubi8/ubi                   latest
3269c37eae33    8 weeks ago   208 MB
nexus-registry-int.apps.tools-na150.prod.ole.redhat.com/openshift/ocp4   4.6.4-
x86_64      26f7cd4cf1fb   2 months ago  322 MB
```

- 7. As the `lab` user on the `utility` server, verify the DNS prerequisites for the OpenShift API and APP Ingress load balancers. The OpenShift API load balancer DNS record and the OpenShift APP Ingress load balancer wildcard DNS record must use the load balancer IP address.

In following chapters, you will learn that:

- The classroom environment load balancer runs on the `utility` server using the `192.168.50.254` IP address.
 - Because you will install an OpenShift cluster called `ocp4` in the `example.com` domain, the OpenShift cluster domain will be `ocp4.example.com`.
- 7.1. Verify that the DNS record of the OpenShift API load balancer is configured to use the classroom environment load balancer IP address `192.168.50.254`.

```
[lab@utility ~]$ dig api.ocp4.example.com

; <>> DiG 9.11.13-RedHat-9.11.13-3.el8 <>> api.ocp4.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 47126
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2
```

Chapter1 | Describing the OpenShift Installation Process

```
;; OPT PSEUDOSECTION:  
; EDNS: version: 0, flags:; udp: 4096  
; COOKIE: 386ab7d875e394653ca11a22601bfe40a8ddbd530978d9d7 (good)  
;; QUESTION SECTION:  
;api.ocp4.example.com. IN A  
  
;; ANSWER SECTION:  
api.ocp4.example.com. 86400 IN A 192.168.50.254  
  
;; AUTHORITY SECTION:  
example.com. 86400 IN NS dns.ocp4.example.com.  
  
;; ADDITIONAL SECTION:  
dns.ocp4.example.com. 86400 IN A 192.168.50.254  
  
;; Query time: 2 msec  
;; SERVER: 172.25.250.254#53(172.25.250.254)  
;; WHEN: Thu Feb 04 09:01:36 EST 2021  
;; MSG SIZE rcvd: 127
```

```
[lab@utility ~]$ dig api-int.ocp4.example.com  
  
; <>> DiG 9.11.13-RedHat-9.11.13-3.el8 <>> api-int.ocp4.example.com  
;; global options: +cmd  
;; Got answer:  
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26234  
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2  
  
;; OPT PSEUDOSECTION:  
; EDNS: version: 0, flags:; udp: 4096  
; COOKIE: 010170465441c41bb60331c9601bfe78fe03a7a63159f818 (good)  
;; QUESTION SECTION:  
;api-int.ocp4.example.com. IN A  
  
;; ANSWER SECTION:  
api-int.ocp4.example.com. 86400 IN A 192.168.50.254  
  
;; AUTHORITY SECTION:  
example.com. 86400 IN NS dns.ocp4.example.com.  
  
;; ADDITIONAL SECTION:  
dns.ocp4.example.com. 86400 IN A 192.168.50.254  
  
;; Query time: 2 msec  
;; SERVER: 172.25.250.254#53(172.25.250.254)  
;; WHEN: Thu Feb 04 09:02:32 EST 2021  
;; MSG SIZE rcvd: 131
```

- 7.2. Verify that the wildcard DNS record of the OpenShift APP Ingress load balancer is configured to use the classroom environment load balancer IP address 192.168.50.254.

```
[lab@utility ~]$ dig test.apps.ocp4.example.com

; <>> DiG 9.11.13-RedHat-9.11.13-3.el8 <>> test.apps.ocp4.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 47551
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 1c160cfb1d57ff3f547b7e85601bff32fe98468a22aea3f9 (good)
;; QUESTION SECTION:
;test.apps.ocp4.example.com. IN A

;; ANSWER SECTION:
test.apps.ocp4.example.com. 86400 IN A 192.168.50.254

;; AUTHORITY SECTION:
example.com. 86400 IN NS dns.ocp4.example.com.

;; ADDITIONAL SECTION:
dns.ocp4.example.com. 86400 IN A 192.168.50.254

;; Query time: 3 msec
;; SERVER: 172.25.250.254#53(172.25.250.254)
;; WHEN: Thu Feb 04 09:05:38 EST 2021
;; MSG SIZE rcvd: 133
```

Finish

Do not make any other changes to the lab environment until the next guided exercise. You will continue using this environment in upcoming exercises.

This concludes the guided exercise.

Introducing Hosted OpenShift

Objectives

- Describe the differences between a self-managed OpenShift cluster and hosted OpenShift offerings.

Comparing Self-managed with Hosted OpenShift Offerings

OpenShift 4 offers a variety of installation methods depending on the customer needs, including full stack and pre-existing infrastructure installation methods for both on-premise installations and also installations on IaaS cloud providers. These installation methods, cluster operation, and maintenance require more or less expertise and effort from the cluster administrators.

Customers planning to use OpenShift in production environments, but unwilling to invest the time and resources required, can turn to Red Hat and other cloud providers offering OpenShift hosted services.

- Hosted services allow OpenShift users to outsource the provisioning, installation, and configuration of the required infrastructure for running their clusters.
- Hosted services also remove the burden of maintaining the clusters, including upgrades to the latest, supported versions of OpenShift.
- Furthermore, hosted services enable customers to benefit from Red Hat's experience managing OpenShift in a cloud environment.

In contrast, there are some limitations when using OpenShift hosted services compared to self-managed OpenShift installations:

- Hosted services do not provide user accounts with administrator permissions for managing the underlying infrastructure.
- The number of customizations during the cluster installation is limited compared to a self-managed installation.
- Specific node-level customizations might not be available (like the number of pods per node).
- OpenShift technology preview features might not be available.
- A curated list of operators is available. Administrators can only install operators to specific namespaces.
- The ability to deploy applications requiring special hardware features is reduced.

Introducing OpenShift Hosted Offerings

As previously discussed, Red Hat partners with other vendors to offer a variety of hosted (or managed) OpenShift services as an alternative to self-managed OpenShift installations.

You can find a high-level comparison of the different OpenShift hosted offerings in the following table:

Hosted offering	Cloud hosted	Billed by	Managed by	Supported by
Microsoft Azure Red Hat OpenShift	Azure	Microsoft	Red Hat and Microsoft	Red Hat and Microsoft
Red Hat OpenShift Dedicated	AWS or GCP	Red Hat (OpenShift), AWS or GCP (Infrastructure)	Red Hat	Red Hat
Red Hat OpenShift on IBM Cloud	IBM Cloud	IBM	IBM	Red Hat and IBM
Red Hat OpenShift Service on AWS	AWS	AWS	Red Hat and AWS	Red Hat and AWS

For more information on each hosted service, refer to:

- Microsoft Azure Red Hat OpenShift: <https://www.openshift.com/products/azure-openshift>
- Red Hat OpenShift Dedicated: <https://www.openshift.com/products/dedicated/>
- Red Hat OpenShift on IBM Cloud: <https://www.openshift.com/products/openshift-ibm-cloud/>
- Red Hat OpenShift Service on AWS: <https://www.openshift.com/products/amazon-openshift/>

Comparing OpenShift Hosted Offerings with Third Parties Managed Kubernetes Offerings

OpenShift hosted offerings provide customers with the ability to deploy workloads within minutes after purchasing the service. In addition to the capabilities provided by Kubernetes, OpenShift delivers a complete application development, deployment, and runtime platform. It also includes features such as pipelines, monitoring, logging, security, and the Red Hat Service Mesh.

Because it can run on a wide variety of cloud vendors, OpenShift provides the same consistent platform experience, no matter where it runs. The following table provides a high-level comparison between OpenShift hosted offerings and third-party managed Kubernetes offerings:

	Red Hat hosted services	Third parties managed Kubernetes services
Operating system	RHEL/RHCOS	Varies
Platform	Kubernetes + OCP	Kubernetes
Cross cloud portability	Portable to other OS or Kubernetes	Kubernetes part only, other services are vendor-specific
Identity and Auth	LDAP, Google, OpenID, GitHub, GitLab	Vendor specific (IAM, Azure AD, Google/Cloud IAM)

	Red Hat hosted services	Third parties managed Kubernetes services
Logs	EFK, Log Forwarding (OpenShift Dedicated) or Azure Monitor	Nothing by default
Metrics	Prometheus and Grafana or Azure Monitor	Nothing by default
CLIs and APIs	oc and kubectl	kubectl + vendor-specific for infrastructure
Cluster Network	Managed by Red Hat	Generally self-managed
CI/CD	S2I, Image Streams, Jenkins, Tekton	Nothing by default
Catalog	Operator Hub, Red Hat Marketplace	Vendor specific or community sources
Container Registry	Included	Nothing by default
Support	24x7 Premium Support	Basic support, can purchase higher levels
Updates	Managed by Red Hat	Mix of provider automated + customer manual activities
Snapshots	Managed by Red Hat for platform and PVs	Managed by the customer
Workload notifications	Grafana and Prometheus (OpenShift Dedicated), OCP Web Console	Nothing by default
Platform notifications	Status Portal	Vendor specific dashboards



References

Red Hat OpenShift Managed services

<https://www.openshift.com/learn/topics/managed-services>

OpenShift Products page

<https://www.openshift.com/products>

► Quiz

Introducing Hosted OpenShift

Choose the correct answers to the following questions:

► 1. **Which four of the following cloud providers offer a hosted OpenShift service? (Choose four.)**

- a. Microsoft Azure
- b. Alibaba Cloud
- c. Amazon Web Services
- d. IBM Cloud
- e. Google Cloud
- f. OVHcloud

► 2. **Which of the following sentences regarding hosted OpenShift services is true?**

- a. Red Hat does not participate in supporting the OpenShift Service on AWS.
- b. Red Hat participates in supporting hosted OpenShift services on every cloud provider.
- c. All the hosted OpenShift services are billed by Red Hat.
- d. Red Hat does not partner with Microsoft to provide a hosted OpenShift service.

► 3. **Which two of the following are advantages of using a hosted OpenShift service? (Choose two.)**

- a. Customers receive administrator accounts for managing the underlying infrastructure.
- b. Customers do not need to invest time and resources in cluster upgrades.
- c. Customers do not need to know how to configure the cloud provider infrastructure for running OpenShift.
- d. Customers can perform any customization at the node level.

► Solution

Introducing Hosted OpenShift

Choose the correct answers to the following questions:

- ▶ **1. Which four of the following cloud providers offer a hosted OpenShift service? (Choose four.)**
 - a. Microsoft Azure
 - b. Alibaba Cloud
 - c. Amazon Web Services
 - d. IBM Cloud
 - e. Google Cloud
 - f. OVHcloud

- ▶ **2. Which of the following sentences regarding hosted OpenShift services is true?**
 - a. Red Hat does not participate in supporting the OpenShift Service on AWS.
 - b. Red Hat participates in supporting hosted OpenShift services on every cloud provider.
 - c. All the hosted OpenShift services are billed by Red Hat.
 - d. Red Hat does not partner with Microsoft to provide a hosted OpenShift service.

- ▶ **3. Which two of the following are advantages of using a hosted OpenShift service? (Choose two.)**
 - a. Customers receive administrator accounts for managing the underlying infrastructure.
 - b. Customers do not need to invest time and resources in cluster upgrades.
 - c. Customers do not need to know how to configure the cloud provider infrastructure for running OpenShift.
 - d. Customers can perform any customization at the node level.

► Quiz

Chapter Review: Describing the OpenShift Installation Process

Choose the correct answers to the following questions:

- ▶ 1. **Which of the following actions is performed by the OpenShift installer when using the full-stack automation installation method?**
 - a. Install RHEL 7 on the control plane nodes
 - b. Create the developer user
 - c. Set up load balancers
 - d. Configure an NFS server for persistent storage

- ▶ 2. **Which of the following actions is performed by the OpenShift installer when using the pre-existing infrastructure installation method?**
 - a. Starts the installation of RHCOS on the nodes
 - b. Generates the ignition configuration files
 - c. Builds all the network resources required for the installation
 - d. Sets up a load balancer for the API server

- ▶ 3. **Which OpenShift installation method is recommended for installing OpenShift when the administrator wants full flexibility and customization of the infrastructure?**
 - a. The pre-existing infrastructure installation method
 - b. The disconnected installation mode
 - c. The full-stack automation installation method
 - d. The connected installation mode

- ▶ 4. **Which option is best for an OpenShift user who wants quick access to a cluster, and also needs it to be supported, configured, and maintained by a certified partner?**
 - a. A self-managed OpenShift cluster, using the pre-existing infrastructure installation method.
 - b. A self-managed OpenShift cluster, using the full-stack automation installation method.
 - c. A hosted OpenShift cluster.
 - d. A self-managed OpenShift cluster, using the connected installation mode.

► **5. Which three companies have a hosted OpenShift offering? (Choose three.)**

- a. Red Hat
- b. Microsoft (Azure)
- c. VMware
- d. Amazon Web Services
- e. Oracle

► Solution

Chapter Review: Describing the OpenShift Installation Process

Choose the correct answers to the following questions:

- ▶ 1. **Which of the following actions is performed by the OpenShift installer when using the full-stack automation installation method?**
 - a. Install RHEL 7 on the control plane nodes
 - b. Create the developer user
 - c. Set up load balancers
 - d. Configure an NFS server for persistent storage

- ▶ 2. **Which of the following actions is performed by the OpenShift installer when using the pre-existing infrastructure installation method?**
 - a. Starts the installation of RHCOS on the nodes
 - b. Generates the ignition configuration files
 - c. Builds all the network resources required for the installation
 - d. Sets up a load balancer for the API server

- ▶ 3. **Which OpenShift installation method is recommended for installing OpenShift when the administrator wants full flexibility and customization of the infrastructure?**
 - a. The pre-existing infrastructure installation method
 - b. The disconnected installation mode
 - c. The full-stack automation installation method
 - d. The connected installation mode

- ▶ 4. **Which option is best for an OpenShift user who wants quick access to a cluster, and also needs it to be supported, configured, and maintained by a certified partner?**
 - a. A self-managed OpenShift cluster, using the pre-existing infrastructure installation method.
 - b. A self-managed OpenShift cluster, using the full-stack automation installation method.
 - c. A hosted OpenShift cluster.
 - d. A self-managed OpenShift cluster, using the connected installation mode.

► **5. Which three companies have a hosted OpenShift offering? (Choose three.)**

- a. Red Hat
- b. Microsoft (Azure)
- c. VMware
- d. Amazon Web Services
- e. Oracle

Summary

- The details of the OpenShift installation process.
- The OpenShift installation methods:
 - Use the full-stack automation installation method to deploy the cluster on infrastructure that the installer provisions and the cluster maintains.
 - Use the pre-existing infrastructure installation method to deploy a cluster on infrastructure that administrators prepare and maintain themselves.
- The OpenShift installation modes:
 - Use the connected installation mode when the cluster nodes have access to the Internet.
 - Use the disconnected mode when the cluster nodes are on a restricted network.
- The main assets created by the OpenShift installer:
 - The installation configuration file `install-config.yaml`
 - The Kubernetes manifests
 - The ignition configuration files
- The prerequisites that the infrastructure environment must fulfill to install OpenShift.
- The steps to validate that an OpenShift cluster is healthy and ready for day-2 tasks to onboard users and applications.
- How to edit the `install-config.yaml` configuration file to customize the cluster installation.
- How to run, monitor, and troubleshoot each stage of the OpenShift installation process.
- How to gather information from an OpenShift cluster.
- How Red Hat partners with other vendors to provide OpenShift hosted offerings.

Chapter 2

Installing OpenShift on a Cloud Provider

Goal

Provision OpenShift clusters on IaaS cloud providers, with common customizations, using the full-stack automation installation method.

Objectives

- Describe the architecture and workflow for installing OpenShift on an IaaS Cloud Provider using the full-stack automation method.
- Provide the prerequisites to install OpenShift on Amazon Web Services (AWS) using full-stack automation.
- Install OpenShift on Amazon Web Services (AWS) using full-stack automation with common customizations.
- Assess the success of an installation of OpenShift on AWS.

Sections

- Introducing OpenShift Full-stack Automation Installation on a Cloud Provider (and Quiz)
- Describing How to Install OpenShift on AWS Using Full-stack Automation (and Quiz)
- Demonstrating How to Install OpenShift on AWS Using Full-stack Automation (and Quiz)
- Verifying the Installation of OpenShift on AWS (and Quiz)

Introducing OpenShift Full-stack Automation Installation on a Cloud Provider

Objectives

- Describe the architecture and workflow for installing OpenShift on an IaaS Cloud Provider using the full-stack automation method.

Introducing OpenShift Installations on a Cloud Provider

When using the full-stack automation method to install OpenShift on a supported IaaS cloud provider, the installer automatically provisions the required infrastructure and resources.

- OS images
- VMs or cloud instances
- Load balancers
- Storage
- Networking

Using this method, administrators can install OpenShift with minimal intervention on the following supported IaaS cloud providers:

- Amazon Web Services (AWS)
- Red Hat OpenStack Platform (RHOSP)
- Microsoft Azure (MS Azure)
- Google Cloud Platform (GCP)

Reasons to Install OpenShift on a Cloud Provider Using Full-stack Automation

On supported cloud providers, Red Hat recommends using the full-stack automation installation method for the following reasons:

- Administrators install OpenShift with minimal manual intervention in an opinionated manner.
- The OpenShift installer automatically creates the required resources.
- The OpenShift installer fully integrates the OpenShift Machine API resource with the cloud provider services. The installer creates the OpenShift MachineSets resource, so automatic node provisioning and cluster autoscaling are ready to use after the installation finishes.
- Red Hat supports the OpenShift installation on the most relevant cloud providers.
- Red Hat partners with the most relevant cloud providers to ensure the quality of the full-stack automation installation method.

**Note**

Administrators can also install OpenShift using the pre-existing infrastructure method on a cloud provider. Using this approach, administrators must create manually the required cloud resources before installing OpenShift.

Describing OpenShift Installations on a Cloud Provider

The following list summarizes the main steps to install OpenShift on a cloud provider using the full-stack automation method. This process is described in greater detail in *Introducing OpenShift Installation Methods* section.

1. Fulfill the general and full-stack automation prerequisites listed in the *Describing OpenShift Installation Prerequisites* lecture.
2. Verify the infrastructure prerequisites, cloud account permissions, and quotas.
3. Run the OpenShift installer on the bastion host.
4. The OpenShift installer creates the required cloud resources and starts the cluster installation.
5. Monitor the OpenShift installation process.
6. Verify the cluster health after the OpenShift installation finishes.
7. Perform post-installation tasks (optional).

The cluster installation step has the following stages:

Resource Creation Stage

First, the OpenShift installer creates and configures the following cloud resources:

- Cluster network
- Cluster network services
- Cluster network gateway
- Cluster load balancers
- Cluster storage
- Bootstrap node
- Control plane nodes

The following diagram shows the relationship between OpenShift resources and the cloud provider services used to create them.

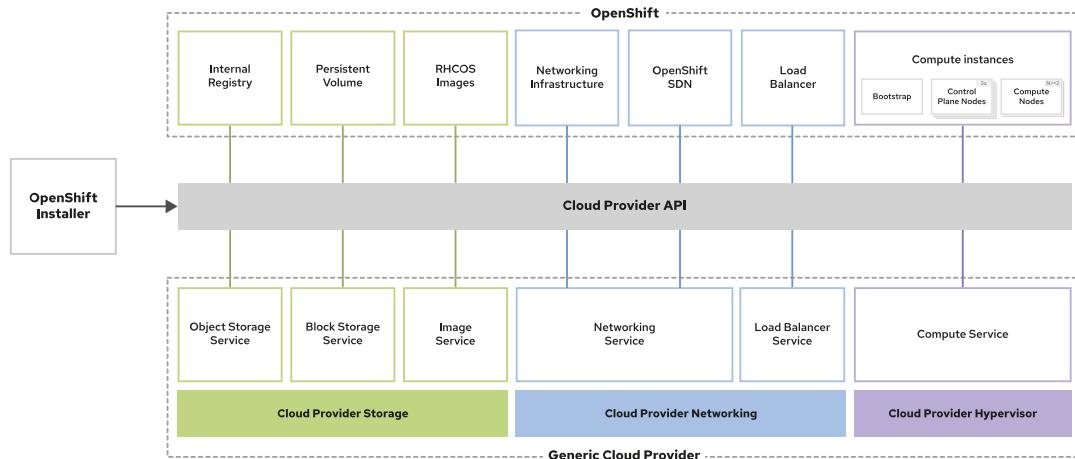


Figure 2.1: OpenShift full-stack automation installation resources on a generic cloud provider

Bootstrap Stage

At this stage:

- The bootstrap node starts the Kubernetes API.
- The control plane nodes try to fetch their ignition configuration files from the Kubernetes API.
- When the Kubernetes API becomes available on the bootstrap node, the control plane nodes fetch their ignition configuration files successfully and finish installing.

The following diagram explains the architecture of an OpenShift deployment on a generic cloud provider at the bootstrap stage.

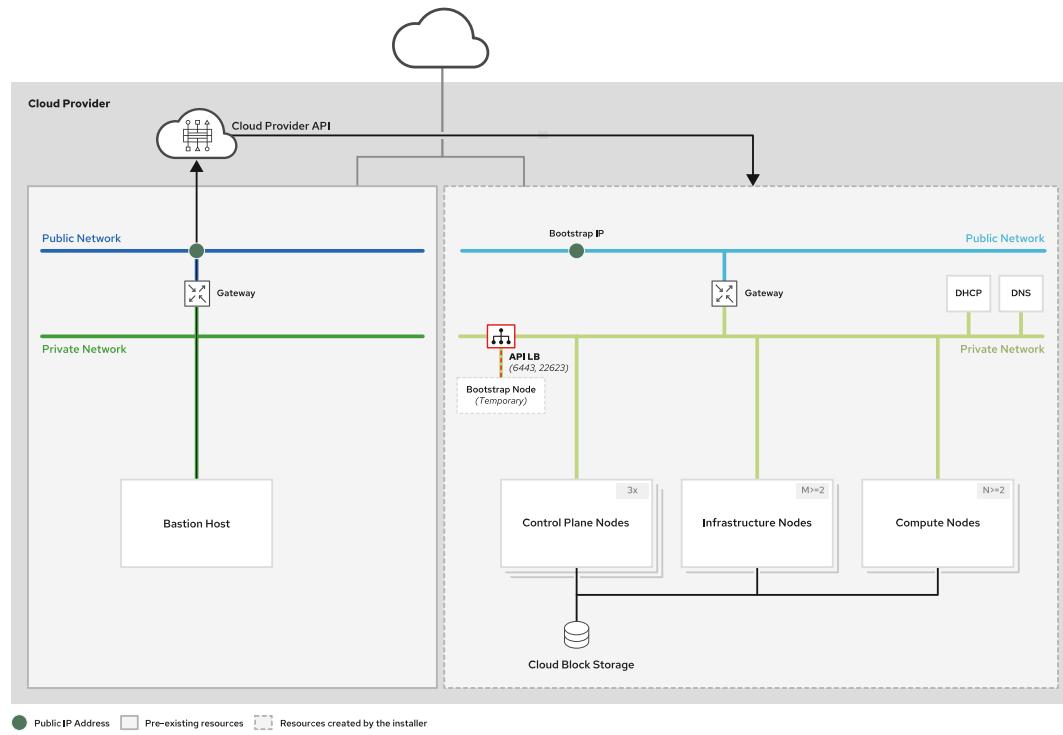


Figure 2.2: OpenShift full-stack automation installation on a generic cloud provider - Bootstrap stage

Production Control Plane Stage

Once the bootstrap node transfers the temporary control plane to the control plane nodes, it is no longer needed.

At this stage:

- The production control plane runs on the control plane nodes.
- The OpenShift API load balancer (OpenShift API LB) uses the control plane nodes as the back-end pool members.
- The installer creates and installs the compute nodes. The compute nodes join the cluster.
- The OpenShift application Ingress load balancer (OpenShift APP Ingress LB) uses the compute nodes as the back-end pool members.
- The installer publishes the internal load balancers IP addresses on the cloud provider public network (optional).
- In parallel, the cluster version operator (CVO), running on the production control plane, installs the operators that build the cluster, and then finishes the installation.
- The installer configures the OpenShift internal registry to use the cloud object storage.
- The installer configures a dynamic storage provider to provide persistent storage for the containerized applications using the cloud storage service.
- If the cloud provider services have OpenShift Machine API integration support, then the installer configures node provisioning and cluster autoscaling.

The following diagram shows the architecture of an OpenShift deployment on a generic cloud provider after the installation has finished.

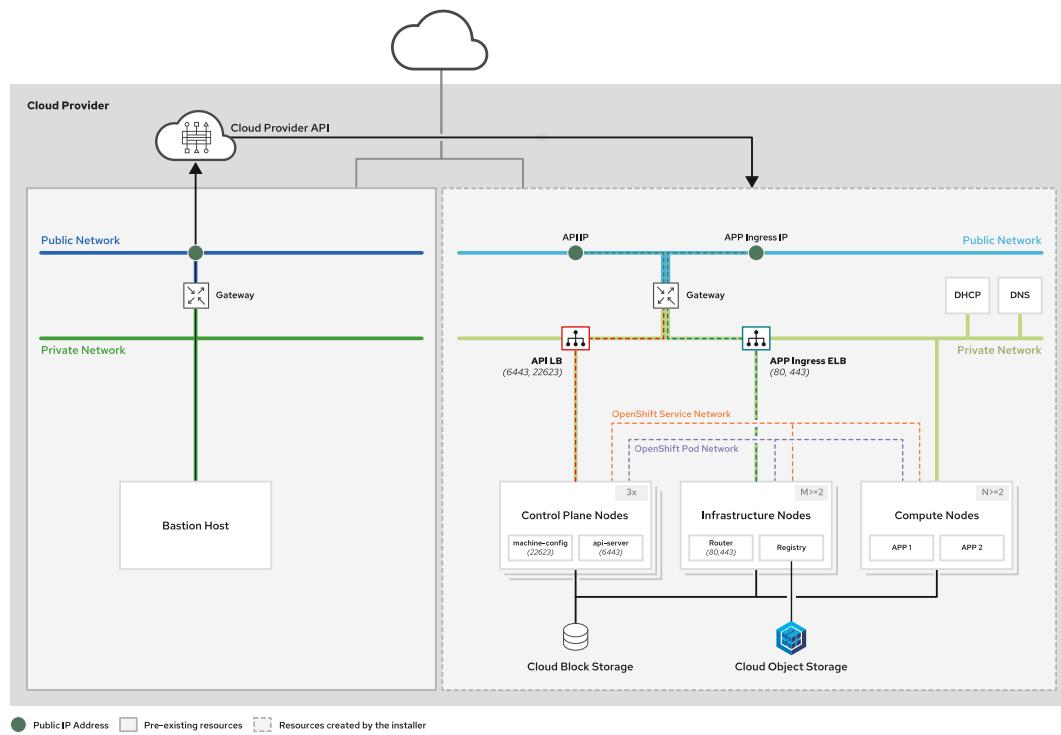


Figure 2.3: OpenShift full-stack automation installation on a generic cloud provider - Final stage

During the OpenShift installation on a generic cloud provider:

- The bastion host requires network connectivity to the cloud provider API to install OpenShift and to the OpenShift API for managing OpenShift after installation. It is not mandatory (but is recommended) to install the bastion host on the same cloud provider used to install OpenShift.
- The OpenShift installation process does not create infrastructure nodes; they are just compute nodes. The infrastructure nodes are compute nodes running router or registry pods.
- The OpenShift installation process creates the APP Ingress LB cloud resource containing all the compute nodes as the back-end pool members for the 80/TCP and 443/TCP ports. Only compute nodes running router pods that are listening on these ports will pass the LB health check. Therefore, the APP Ingress LB sends network traffic only to the compute nodes running the router pods.
- The DHCP configuration also provides the NTP configuration.
- By default, the chronyd systemd service running on each cluster node uses the public NTP pool `rhel.pool.ntp.org` to synchronize the system clock. After installation, administrators can configure the cluster nodes to use a local NTP server using the machine config operator as explained in the *Configuring chrony time service* section at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/post-installation_configuration.

Comparing IaaS Cloud Providers Terminology

Each cloud provider uses specific terminology to refer to different services used during the installation.

Chapter 2 | Installing OpenShift on a Cloud Provider

The following table shows the different cloud terminology used on the most common cloud providers.

Cloud Provider Services Used During an OpenShift Full-stack Automation Installation

OpenShift resources	AWS provider	RHOSP provider	Azure provider	GCP provider
Internal registry storage	Amazon S3	Swift	Azure Blob	Google Cloud Object Storage
Persistent volume	Amazon EBS	Cinder	Azure Disk	Google Cloud Block Storage
RHCOS images	Amazon AMI	Glance	Azure Images (VHD)	Google Compute Engine Images
Networking	Amazon VPC	Neutron	Azure VNet	Google VPC
SDN / OVN	Amazon VPC	Neutron + Kuryr	Azure VNet	Google VPC
Load balancer	Amazon ELB	Static pods/ Octavia LBaaS	Azure LB	Google Networking Suites
DNS	Amazon Route53	External	Azure DNS	Google Cloud DNS
Compute instances	Amazon EC2	Nova	Azure Compute	Google Compute Engine

Installing OpenShift Using Full-stack Automation on Microsoft Azure

Microsoft Azure (MS Azure) is a supported option when installing OpenShift on a cloud provider using the full-stack automation installation method. Using this installation method, administrators can install an OpenShift cluster on MS Azure in an opinionated manner with minimal intervention.

Prerequisites for Installing OpenShift on MS Azure

Before installing OpenShift on MS Azure, administrators must perform the following actions:

- Review the general and full-stack automation prerequisites described in the *Describing OpenShift Installation Prerequisites* lecture.
- Create a public DNS hosted zone for the cluster in the Azure DNS service, if it does not already exist. This zone must be authoritative for the domain.
- Use an Azure Identity and Access Management (IAM) user account to run the installation. This account must have the `User Access Administrator` role permission.
- Ensure that the IAM Azure account fulfills the required quotas and permissions to install OpenShift.

Chapter 2 | Installing OpenShift on a Cloud Provider

- Create a service principal for representing the OpenShift installer in the Azure Resource Manager.
- Verify the supported Azure regions. The OpenShift installer dynamically generates a list of available Azure regions based on the MS Azure account subscription.
- Check the Azure resource name restrictions.

**Note**

- For more information, refer to the *Configuring an Azure account* section of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_azure

Sizing OpenShift Installations on MS Azure

Administrators must verify the MS Azure account resource quotas to ensure the following minimum requirements.

Azure Limits That Can Impact the Ability to Install and Run OpenShift Clusters

Resource	Required	Default Azure limit
vCPU	40	20 per region
VNet	1	1000 per region
Network interfaces	6	65536
Network security groups	2	5000
Network load balancers	3	1000 per region
Public IP addresses (PIP)	3	-
Private IP addresses	7	-

**Note**

The OpenShift resource requirements described in this table are adequate for installing a small OpenShift cluster. The application workloads running on the OpenShift cluster can affect the actual requirements.

**Note**

Because the OpenShift installation requires at least 40 vCPUs and the default Azure limit per region is 20, you will probably need to increase this limit.

- For more information, refer to the *Increasing Azure account limits* guide in the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_azure

Customizing OpenShift Installations on MS Azure

By default, the virtual machine (VM) sizes used on an OpenShift installation on Azure are **Standard_D8s_v3** for control plane nodes and **Standard_D4s_v3** for compute nodes. These default Azure VM sizes are adequate for proof of concept (PoC) and other non-production scenarios.

MS Azure Dsv3-series VM Sizes (Reference)

Size	vCPU	RAM (GiB)	Max SSD IOPS	Max SSD throughput
Standard_D4s_v3	4	16	8000	100 MiB/s
Standard_D8s_v3	8	32	16000	200 MiB/s
Standard_D16s_v3	16	64	32000	400 MiB/s
Standard_D32s_v3	32	128	64000	800 MiB/s

For production clusters, use a larger VM size than the default **Standard_D8s_v3** for the control plane nodes to ensure an adequate storage performance for the etcd cluster running on them.

Also, for production clusters, use a larger VM size than the default **Standard_D4s_v3** for the compute nodes to ensure an adequate performance for the cluster infrastructure services running on them.



Note

For more information, refer to the *Installing a cluster on Azure with customizations* section of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_azure

For more information about OpenShift installations on MS Azure: <https://www.openshift.com/blog/openshift-4-2-on-azure-preview>

Describing OpenShift Installations on MS Azure

The steps required to run an OpenShift installation on a generic cloud provider are discussed in the *Describing OpenShift Installations on a Cloud Provider* section of this lecture. This section describes the particularities of the OpenShift installation process on the MS Azure cloud provider at each stage of the installation.

Resource Creation Stage

At the beginning of the installation, the OpenShift installer creates and configures the required cloud resources.

The OpenShift installer uses:

- Azure virtual network (VNet) service to create the cluster network and the cluster network services.
- Azure NAT Gateway service to create the cluster network gateway.
- Azure load balancer (LB) service to create the cluster load balancers.

Chapter 2 | Installing OpenShift on a Cloud Provider

- Azure Compute service to create the cluster compute nodes.
- Azure Disk and Azure Blob services for the cluster storage.

The following diagram shows the relationship between OpenShift resources and the MS Azure services used to create them.

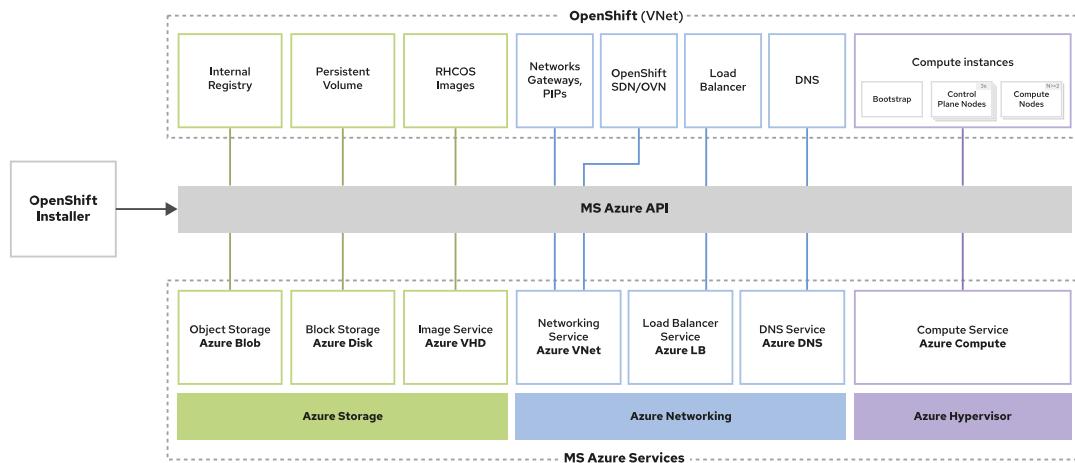


Figure 2.4: OpenShift full-stack automation installation resources on MS Azure

Bootstrap Stage

At this stage, the bootstrap node runs the Kubernetes API and the temporary control plane. The OpenShift installer begins to install the cluster nodes.

The following diagram explains the architecture of an OpenShift deployment running on MS Azure at the bootstrap stage.

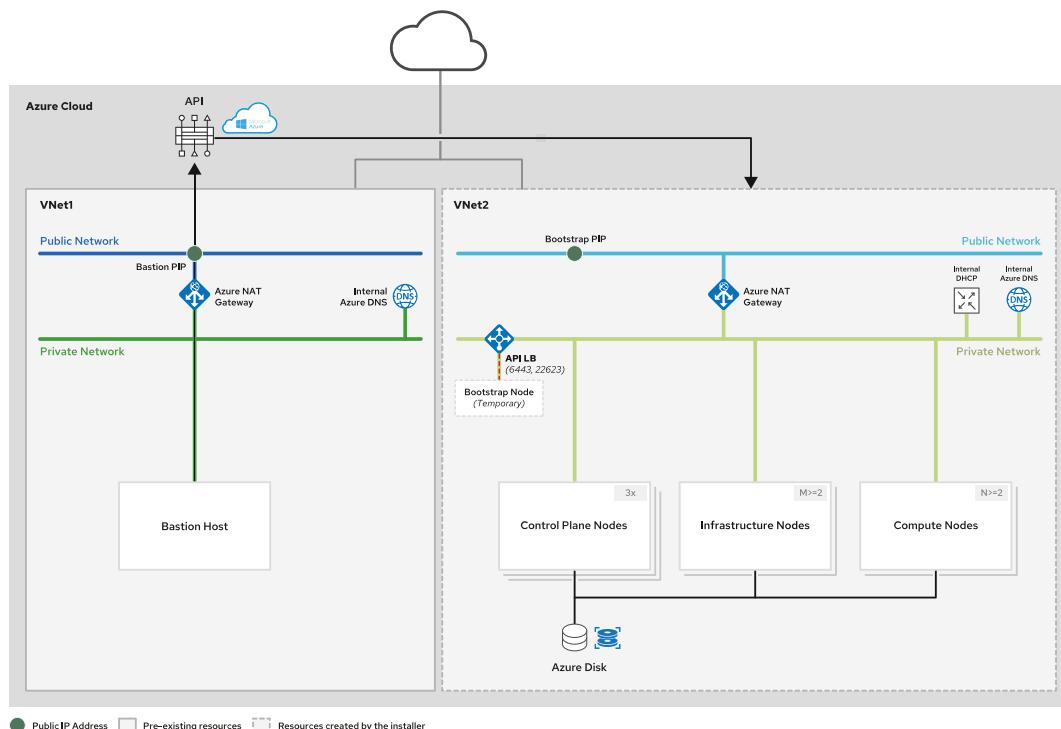


Figure 2.5: OpenShift full-stack automation installation on MS Azure - Bootstrap stage

Production Control Plane Stage

At this stage, the Kubernetes API and the production control plane are running on the control plane nodes. The cluster version operator (CVO), running on the production control plane, installs the operators that build the cluster and then finishes the installation.

The OpenShift installer:

- Publishes the OpenShift API LB and APP Ingress LB IPs on the Azure public network as public IPs.

When installing OpenShift on MS Azure, the installer publishes the OpenShift load balancer endpoints using public IPs. This step is optional when installing OpenShift on other cloud providers, such as RHOSP or AWS, but it is mandatory when installing OpenShift on the MS Azure cloud provider.

To avoid the use of public IPs for the cluster load balancer endpoints when installing OpenShift on MS Azure, administrators must configure the cluster to use *User-defined outbound routing* as explained in the *Installing a private cluster on Azure* section of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_azure

- Configures the OpenShift internal registry to use Azure Blob object storage.
- Configures a dynamic storage provider to provide persistent storage for the containerized applications using the Azure Disk storage service.

The following diagram describes the architecture of an OpenShift deployment on MS Azure once the installation has finished.

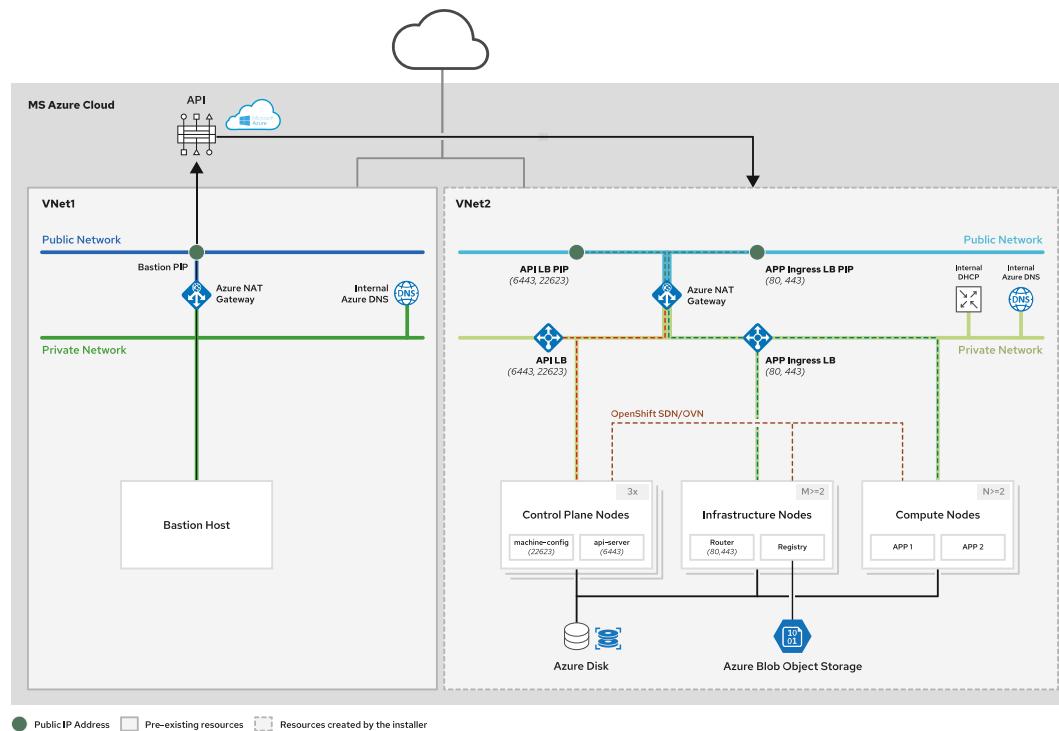


Figure 2.6: OpenShift full-stack automation installation on MS Azure - Final stage

Installing OpenShift Using Full-stack Automation on MS Azure

Video Outcome

Install a minimal OpenShift cluster on MS Azure using the full-stack automation defaults.

Installing OpenShift Using Full-stack Automation on Red Hat OpenStack Platform

When installing OpenShift on a Red Hat OpenStack Platform (RHOSP), Red Hat recommends using the Kuryr SDN. Kuryr is a container network interface (CNI) plug-in solution. It uses the Neutron and Octavia RHOSP services to provide networking for pods and services.

Using the Kuryr SDN for OpenShift deployments on RHOSP has the following advantages:

- Improves the network performance by plugging OpenShift pods into the RHOSP SDN directly.
- Provides internal connectivity between pods and RHOSP virtual instances.
- Avoids network traffic double encapsulation.

Prerequisites for Installing OpenShift on RHOSP Using Kuryr

Before installing OpenShift on RHOSP using Kuryr, administrators must perform the following actions:

- Review the general and full-stack automation prerequisites described in the *Describing OpenShift Installation Prerequisites* lecture.
- Verify the version compatibility between OpenShift and RHOSP from the following documentation page [https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/architecture/index#architecture-installation].
 - For installing OpenShift 4.6 on RHOSP using Kuryr, use the RHOSP 16.1 version.
- Ensure that RHOSP account limits and permissions are sufficient to install OpenShift.
- Ensure that RHOSP has the Cinder block storage and the Swift object storage (optional but recommended) services installed.
- Ensure that RHOSP metadata service is enabled.
- Create the DNS records for the API and Ingress load balancers floating IPs (optional).
- When using Kuryr, administrators must increase the quotas to satisfy the RHOSP resources used by pods, services, namespaces, and network policies.

```
[user@demo ~]$ sudo openstack quota set --secgroups 250 --secgroup-rules 1000 \
> --ports 1500 --subnets 250 --networks 250 <project>
```

Sizing OpenShift Installations on RHOSP Using Kuryr

Administrators need to verify the RHOSP account resource quotas to ensure the following minimum requirements:

Recommended Resources for Installing a Default OpenShift Cluster on RHOSP Using Kuryr

Resource	Value
Floating IP addresses	3 - plus the expected number of Services of LoadBalancer type
Ports	1500 - 1 needed per pod
Routers	1
Subnets	250 - 1 needed per namespace/project
Networks	250 - 1 needed per namespace/project
RAM	112 GB
vCPUs	28
Volume storage	275 GB
Instances	7
Security groups	250 - 1 needed per service and NetworkPolicy
Security group rules	1000
Load balancers	100 - 1 needed per service
Load balancer listeners	500 - 1 needed per service-exposed port
Load balancer pools	500 - 1 needed per service-exposed port



Note

The OpenShift resource requirements described in this table are adequate for installing a small OpenShift cluster. The application workloads running on the OpenShift cluster can affect the actual requirements.

Customizing OpenShift Installations on RHOSP Using Kuryr

By default, the RHOSP instance flavor (4 vCPUs and 8 GiB RAM) used for an OpenShift installation on RHSOP is adequate for proof of concept (PoC) and other non-production scenarios. For production clusters, use an instance flavor with at least with 32 GB of RAM, 8 vCPUs, and 100 GB of storage space for the cluster nodes installation.

Before installing OpenShift on RHOSP, you can also create your custom instance flavors:

```
[user@demo ~]$ sudo openstack flavor create --disk 200 \
> --ram 32384 --vcpu 8 --public m1.master
[user@demo ~]$ sudo openstack flavor create --disk 200 \
> --ram 64384 --vcpu 16 --public m1.worker
```

After they are created, you can include the custom flavors in the `install-config.yaml` file and install the OpenShift cluster.

**Note**

For more information, refer to the *Installing a cluster on OpenStack with customizations* section of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_openstack

Describing OpenShift Installations on RHOSP Using Kuryr

The steps to run an OpenShift installation on RHOSP are explained in the *Describing OpenShift Installations on a Cloud Provider* section of this document. These steps apply to OpenShift installation on RHOSP.

This section describes the particularities of the OpenShift installation process on RHOSP cloud provider at each stage of the installation.

Resource Creation Stage

At the beginning of the installation, the OpenShift installer creates and configures the required cloud resources.

The OpenShift installer uses:

- RHOSP Neutron service to create the cluster network and the cluster network services.
- RHOSP External Router service to create the cluster network gateway.
- Static pods running on cluster nodes to create the cluster load balancers.
- RHOSP Nova service to create the cluster compute nodes.
- RHOSP Cinder and Swift services for the cluster storage.

The following diagram shows the relationship between OpenShift resources and the RHOSP services used to create them.

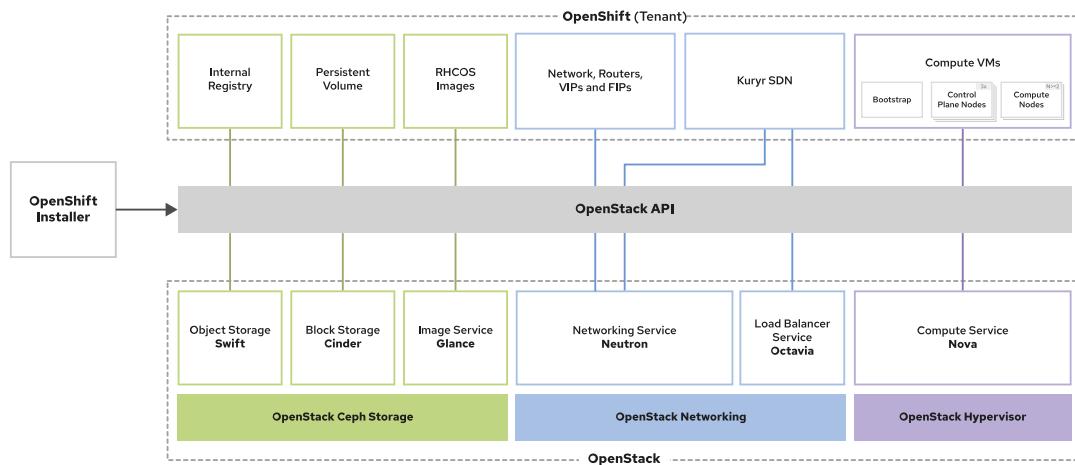


Figure 2.7: OpenShift full-stack automation installation resources on RHOSP

Bootstrap Stage

At this stage, the bootstrap node runs the Kubernetes API and the temporary control plane. The OpenShift installer begins to install the cluster nodes.

Chapter 2 | Installing OpenShift on a Cloud Provider

The bootstrap node hosts the remote resources required for control plane nodes to boot (ignition configuration files) in the Machine Configuration Server (MCS). It also runs a single instance of the etcd cluster.

The following diagram explains the architecture of an OpenShift deployment running on RHOSP at the bootstrap stage.

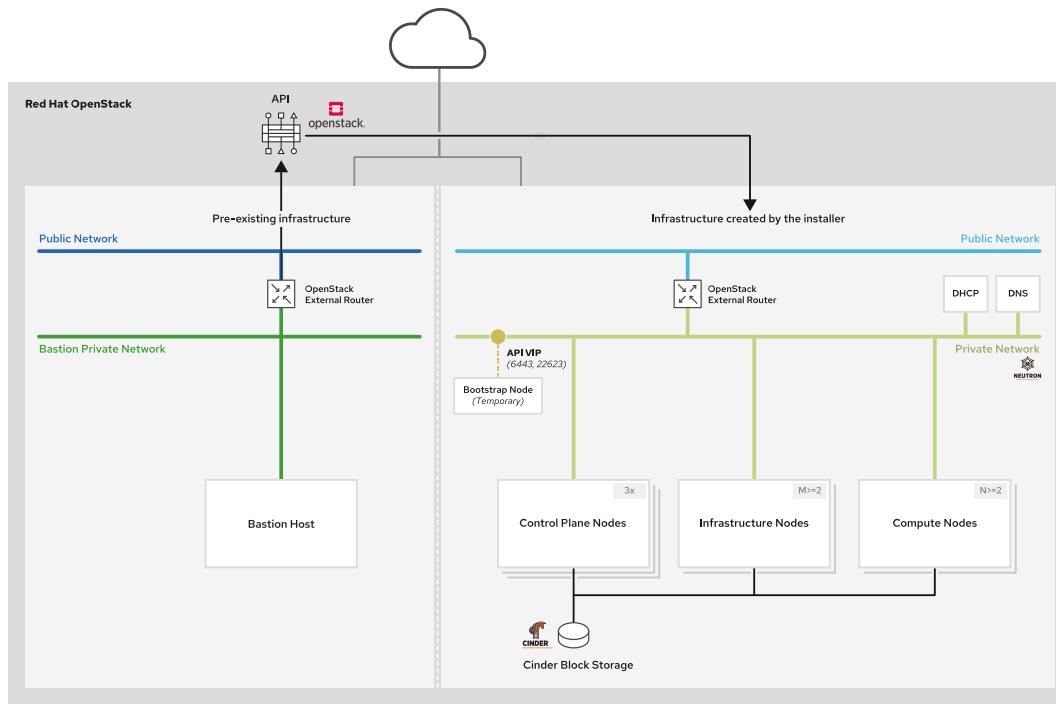


Figure 2.8: OpenShift full-stack automation installation on RHOSP - Bootstrap stage

Production Control Plane Stage

At this stage, the Kubernetes API and the production control plane are running on the control plane nodes. The cluster version operator (CVO), running on the production control plane, installs the operators that build the cluster and then finishes the installation.

The OpenShift installer:

- Publishes the OpenShift API LB and APP Ingress LB IPs on the RHOSP public network as floating IPs (optional).
- Configures the OpenShift internal registry to use RHOSP Swift object storage. If the RHOSP environment is not using the Swift object storage service, then the installer configures the OpenShift internal registry to use RHOSP Cinder block storage.
- Configures a dynamic storage provider to provide persistent storage for the containerized applications using the RHOSP Cinder storage service.

The following diagram describes the architecture of an OpenShift deployment on RHOSP using Kuryr after the installation has finished.

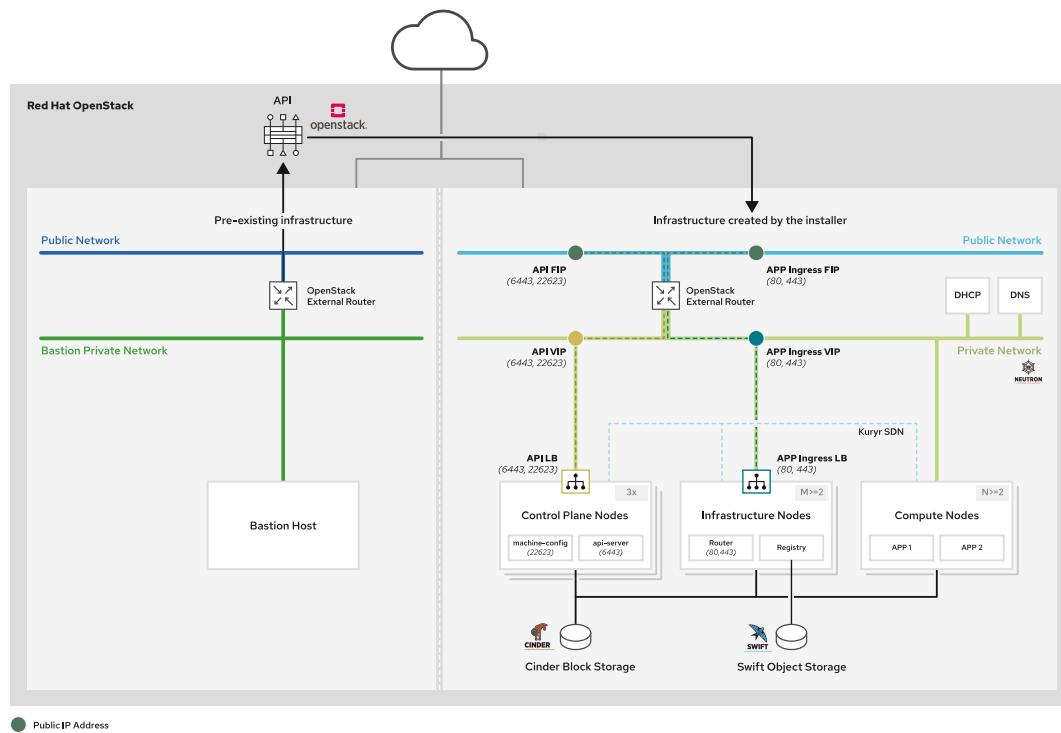


Figure 2.9: OpenShift full-stack automation installation on RHOSP - Final stage

Post-installation Tasks

For large OpenShift cluster deployments on RHOSP with Kuryr, Red Hat recommends using the Octavia LBaaS to scale up the Ingress load balancer.

By default, the OpenShift installer deploys the API and Ingress load balancers on RHOSP using `keepalived`, and `haproxy` static pods. These static pods run on the control plane for the API load balancer and the compute nodes for the Ingress load balancer in an active-passive mode. For each load balancer (API and Ingress), only one `keepalived` static pod at a time provides service.

The active `haproxy` pod forwards the incoming network traffic to the corresponding OpenShift pods. To avoid bottleneck issues on the Ingress load balancer, Red Hat recommends using RHOSP Octavia LBaaS as described in this diagram.

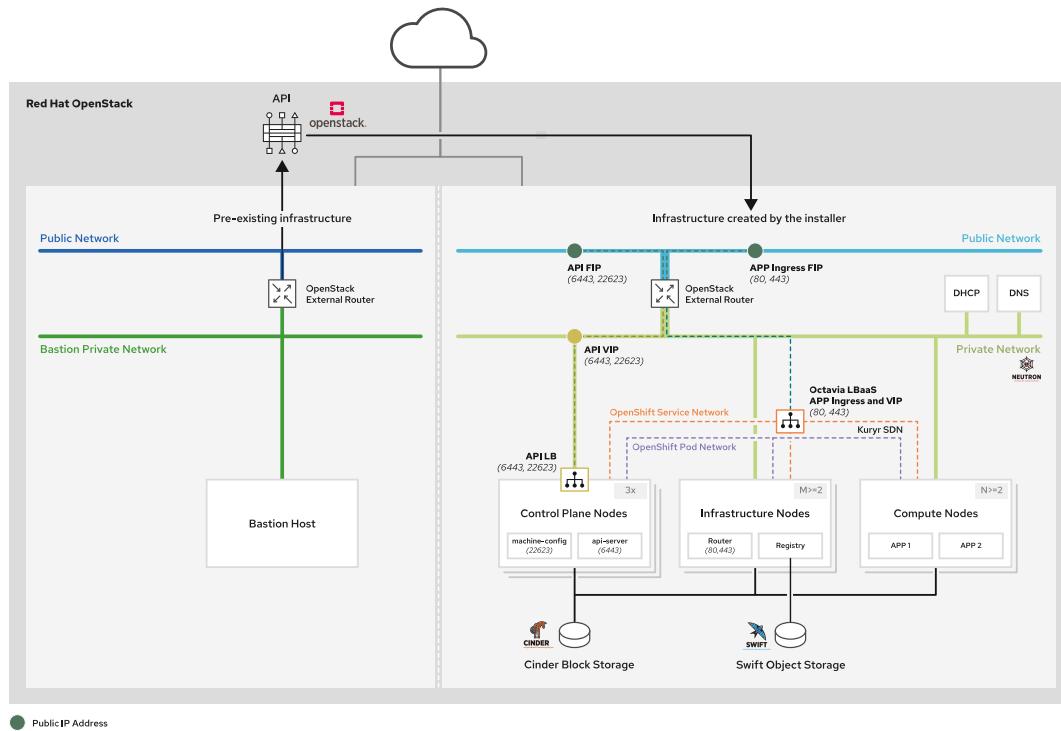


Figure 2.10: OpenShift full-stack automation installation on RHOSP - Scaling Ingress load balancer with Octavia LBaaS



References

- For more information, refer to the *Installing on Azure* guide in the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_azure
- For more information, refer to the *Installing on OpenStack* guide in the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_openstack

► Quiz

Introducing OpenShift Full-stack Automation Installation on a Cloud Provider

Choose the correct answers to the following questions:

- ▶ 1. When installing OpenShift on a generic cloud provider using the full-stack automation method, which three of the following resources are created by the installer? (Choose three.)
 - a. Cluster node instances
 - b. Cluster Network
 - c. Cloud provider account
 - d. Bootstrap node instance
 - e. The bastion host

- ▶ 2. The bootstrap node is used when installing OpenShift on a generic cloud provider using the full-stack automation method. (True or False)
 - a. True
 - b. False

- ▶ 3. On which three of the following IaaS cloud providers does Red Hat support the full-stack automation method for installing OpenShift? (Choose three.)
 - a. Amazon Web Services (AWS)
 - b. Red Hat OpenStack Platform (RHOSP)
 - c. Oracle Cloud
 - d. Google Cloud Platform (GPC)

- ▶ 4. When installing OpenShift on MS Azure cloud using the full-stack automation method, which two of the following are cloud storage resources used by the installer? (Choose two.)
 - a. Azure Blob
 - b. Azure Swift
 - c. Azure EBS
 - d. Azure Disk

- **5. Which two of the following are advantages of using Kuryr SDN for OpenShift deployments on RHOSP? (Choose two.)**
- a. Improves the network performance
 - b. Scales up the Ingress load balancer
 - c. Provides internal connectivity between pods and RHOSP virtual instances
 - d. Enables network traffic double encapsulation

► Solution

Introducing OpenShift Full-stack Automation Installation on a Cloud Provider

Choose the correct answers to the following questions:

- ▶ 1. When installing OpenShift on a generic cloud provider using the full-stack automation method, which three of the following resources are created by the installer? (Choose three.)
 - a. Cluster node instances
 - b. Cluster Network
 - c. Cloud provider account
 - d. Bootstrap node instance
 - e. The bastion host

- ▶ 2. The bootstrap node is used when installing OpenShift on a generic cloud provider using the full-stack automation method. (True or False)
 - a. True
 - b. False

- ▶ 3. On which three of the following IaaS cloud providers does Red Hat support the full-stack automation method for installing OpenShift? (Choose three.)
 - a. Amazon Web Services (AWS)
 - b. Red Hat OpenStack Platform (RHOSP)
 - c. Oracle Cloud
 - d. Google Cloud Platform (GPC)

- ▶ 4. When installing OpenShift on MS Azure cloud using the full-stack automation method, which two of the following are cloud storage resources used by the installer? (Choose two.)
 - a. Azure Blob
 - b. Azure Swift
 - c. Azure EBS
 - d. Azure Disk

- **5. Which two of the following are advantages of using Kuryr SDN for OpenShift deployments on RHOSP? (Choose two.)**
- a. Improves the network performance
 - b. Scales up the Ingress load balancer
 - c. Provides internal connectivity between pods and RHOSP virtual instances
 - d. Enables network traffic double encapsulation

Describing How to Install OpenShift on AWS Using Full-stack Automation

Objectives

- Provide the prerequisites to install OpenShift on Amazon Web Services (AWS) using full-stack automation.

Installing OpenShift Using Full-stack Automation on Amazon Web Services (AWS)

Using the full-stack automation installation method, administrators can smoothly install OpenShift on Amazon Web Services (AWS).

Administrators must fulfill all prerequisites before running an OpenShift installation. Then the OpenShift installer asks for the necessary information and creates the `install-config.yaml` file. Optionally, administrators can customize the installation by modifying the `install-config.yaml` file. Finally, the OpenShift installer creates the required AWS resources and installs an OpenShift Cluster in about 30 minutes.

Prerequisites to Install OpenShift on AWS

Before installing OpenShift on AWS, administrators must perform the following actions:

- Review the general and full-stack automation prerequisites described in the *Describing OpenShift Installation Prerequisites* lecture.
- Create a public DNS hosted zone for the cluster in the AWS Route53 service, if it does not already exist. This zone must be authoritative for the domain.
- Use an AWS Identity and Access Management (IAM) user account to run the installation. Red Hat recommends the creation of a secondary IAM administrative user for this purpose.
- Ensure that the IAM AWS account fulfills the required quotas and permissions to install OpenShift.

Configuring Prerequisites to Install OpenShift on AWS

The following demonstration shows the detailed steps required to fulfill the prerequisites necessary to install OpenShift using the full-stack automation installation method on AWS.

This scenario assumes:

- The bastion host with Red Hat Enterprise Linux 8 operating system is installed.
- The bastion host has access to the AWS API to perform the installation.
- You have an AWS account KEY and SECRET.
- You need to install the OpenShift cluster in the us-east-2 AWS region.
- Your cluster domain is `example.com`.
- You have a user account named `user` on the bastion host.

Chapter 2 | Installing OpenShift on a Cloud Provider

- The user `user` has access to the `root` account using the `sudo` command.

Run all the demonstration steps from the bastion host.

- Install and configure the AWS CLI `aws` tool on the bastion host.

```
[user@bastion ~]$ export AWSKEY=***** && export AWSSECRETKEY=*****
[user@bastion ~]$ export REGION=us-east-2 && mkdir -p $HOME/.aws
[user@bastion ~]$ cat << EOF >> $HOME/.aws/credentials
> [default]
> aws_access_key_id = ${AWSKEY}
> aws_secret_access_key = ${AWSSECRETKEY}
> region = $REGION
> EOF
```

```
[user@bastion ~]$ sudo -i
[root@bastion ~]# curl "https://s3.amazonaws.com/aws-cli/awscli-bundle.zip" \
> -o "awscli-bundle.zip"
[root@bastion ~]# unzip awscli-bundle.zip
[root@bastion ~]# ./awscli-bundle/install -i /usr/local/aws -b /bin/aws
[root@bastion ~]# rm -rf /root/awscli-bundle /root/awscli-bundle.zip
```

- Verify the installation and configuration of the AWS CLI `aws`.

```
[root@bastion ~]# su - user
[user@bastion ~]$ aws sts get-caller-identity
{
    "Account": "*****",
    "UserId": "*****",
    "Arn": "arn:aws:iam::*****user/do322"
}
```

- Install the `openshift-install` and `oc` tools on the bastion host.

```
[user@bastion ~]$ sudo -i
[root@bastion ~]# OCP_VERSION=4.6.4
[root@bastion ~]# MIRROR=mirror.openshift.com/pub/openshift-v4/clients
[root@bastion ~]# wget \
> https://${MIRROR}/ocp/${OCP_VERSION}/openshift-install-linux-
${OCP_VERSION}.tar.gz
[root@bastion ~]# tar zxvf openshift-install-linux-${OCP_VERSION}.tar.gz \
> -C /usr/bin
[root@bastion ~]# rm -f openshift-install-linux-${OCP_VERSION}.tar.gz
[root@bastion ~]# chmod +x /usr/bin/openshift-install
```

```
[root@bastion ~]# wget \
> https://${MIRROR}/ocp/${OCP_VERSION}/openshift-client-linux-
${OCP_VERSION}.tar.gz
[root@bastion ~]# tar zxvf openshift-client-linux-${OCP_VERSION}.tar.gz \
> -C /usr/bin
[root@bastion ~]# rm -f openshift-client-linux-${OCP_VERSION}.tar.gz
[root@bastion ~]# chmod +x /usr/bin/oc
[root@bastion ~]# oc completion bash >/etc/bash_completion.d/openshift
```

- Generate an SSH key on the bastion host.

```
[root@bastion ~]# su - user
[user@bastion ~]$ ssh-keygen -f ${HOME}/.ssh/ocp4-aws-key -N ''
```

- If required, create a public DNS hosted zone in the AWS Route53 service. This zone must be authoritative for the domain.

```
[user@bastion ~]$ aws route53 create-hosted-zone \
> --name example.com --caller-reference 2020-12-24-18:02
{
  "HostedZone": {
    "ResourceRecordSetCount": 2,
    "CallerReference": "2020-12-24-18:02",
    "Config": {
      "PrivateZone": false
    },
    "Id": "/hostedzone/Z0...NE2",
    "Name": "example.com."
  },
  "DelegationSet": {
    "NameServers": [
      "ns-1843.awsdns-38.co.uk",
      "ns-453.awsdns-56.com",
      "ns-1516.awsdns-61.org"
    ]
  },
  "Location": "https://route53.amazonaws.com/...",
  "ChangeInfo": {
    "Status": "PENDING",
    "SubmittedAt": "2021-02-08T10:25:20.795Z",
    "Id": "/change/C09...M4B"
  }
}
```

The 2020-12-24-18:02 string is a unique identifier that you set for this execution. From the command output, you can see that the DNS server ns-1843.awsdns-38.co.uk is authoritative for the newly created hosted domain zone.

If your DNS root domain is not managed by AWS, then you must configure the pertinent delegation records to set the AWS DNS of the hosted domain as authoritative. The propagation of new hosted domain zones on AWS and Internet DNS servers can take up to 48 hours to complete.

Chapter 2 | Installing OpenShift on a Cloud Provider

- Check the DNS hosted domain propagation.

```
[user@bastion ~]$ dig example.com
; <>> DiG 9.11.4-P2-RedHat-9.11.4-9.P2.el7 <>> example.com
;; global options: +cmd
;; Got answer:
;; ->HEADER<- opcode: QUERY, status: NOERROR, id: 52429
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;example.com. IN A

;; AUTHORITY SECTION:
example.com. 300 IN SOA ns-1843.awsdns-38.co.uk. awsdns-hostmaster.amazon.com. 1
    7200 900 1209600 86400

;; Query time: 848 msec
;; SERVER: 192.168.0.2#53(192.168.0.2)
;; WHEN: Tue Dec 29 13:45:44 UTC 2020
;; MSG SIZE rcvd: 134
```

The `dig` command output verifies that the DNS server `ns-1843.awsdns-38.co.uk` is authoritative for the created hosted domain zone.

- Review the new hosted zone domain status in the AWS Route53 service.

```
[user@bastion ~]$ aws route53 list-hosted-zones
{
  "HostedZones": [
    {
      "ResourceRecordSetCount": 3,
      "CallerReference": "example.com.-15...62.53",
      "Config": {
        "PrivateZone": false
      },
      "Id": "/hostedzone/Z24...BD2",
      "Name": "example.com."
    },
    ...
  ...output omitted...
}
```

- Ensure that the AWS account has the required permissions to install OpenShift.

```
[user@bastion ~]$ aws iam get-account-authorization-details
[user@bastion ~]$ aws iam get-account-authorization-details --filter=Role
...output omitted...
```

- Ensure that the AWS account has the required quotas and limits to install OpenShift.

```
[user@bastion ~]$ aws service-quotas list-service-quotas --service-code=ec2
...output omitted...
```

```
[user@bastion ~]$ aws service-quotas list-service-quotas --service-code=vpc
{
  "Quotas": [
    {
      "QuotaName": "Active VPC peering connections per VPC",
      "Adjustable": true,
      "QuotaArn": "arn:aws:servicequotas:us-east-2:743309855567:vpc/
L-7E9ECCDB",
      "Value": 50.0,
      "ServiceName": "Amazon Virtual Private Cloud (Amazon VPC)",
      "GlobalQuota": false,
      "ServiceCode": "vpc",
      "QuotaCode": "L-7E9ECCDB",
      "Unit": "None"
    },
    ...
  ]
}
```

...output omitted...

- Get the registry pull-secret.

Open your web browser and navigate to <https://cloud.redhat.com/openshift/install/aws/installer-provisioned>. Log in using your Red Hat account.

Click the **Download pull secret** button and select **Save file**. The file is saved by default to /home/user/Downloads/pull-secret.

Sizing OpenShift Installations on AWS

Administrators must verify the AWS account resource quotas to ensure the following minimum requirements.

AWS Limits That Can Impact the Ability to Install and Run OpenShift Clusters

Resource	Required	Default AWS limit
Instances	7	Varies
Elastic IPs	1	5 EIPs per account
Virtual Private Clouds (VPCs)	5	5 VPCs per region
Elastic Load Balancing (ELBs)	3	20 per region
NAT Gateways	5	5 per availability zone
Elastic Network Interfaces (ENIs)	At least 12	350 per region
VPC Gateway	20	20 per account
S3 buckets	99	100 buckets per account
Security Groups	250	2500 per account

**Note**

The OpenShift resource requirements described in this table are adequate for installing a small OpenShift cluster. The application workloads running on the OpenShift cluster can affect the actual requirements.

Customizing OpenShift Installations on AWS

The `install-config.yaml` file generated by the OpenShift installer contains the following default values:

```
[user@demo ~]$ cat ${HOME}/ocp4-aws-cluster/install-config.yaml
apiVersion: v1
baseDomain: example.com
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 3
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform: {}
  replicas: 3
metadata:
  creationTimestamp: null
  name: ocp4-aws
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-east-2
publish: External
pullSecret: '{"auths":{"cloud.openshift.com": ...}}
sshKey: |
  ssh-rsa AAAA....
```

Modifying the `install-config.yaml` file, administrators can customize the OpenShift installation on AWS. The following are the most used customizations:

- Increase the EC2 instance sizes for the cluster nodes.
- Increase the EBS disk size for the cluster nodes.
- Use high-performance storage for the etcd cluster.

- Distribute the cluster nodes across different AWS availability zones (AZs).

Customizing EC2 Instance Sizes

The default AWS EC2 instance sizes used on an OpenShift installation on AWS are `m4.xlarge` for the control plane nodes and `m4.large` for the compute nodes. These default AWS EC2 instance sizes are adequate for proof of concept (PoC) and other non-production scenarios.

For production clusters, use larger AWS EC2 instance sizes than the default ones. Use the following AWS EC2 instance sizes guide values as a reference.

AWS Instance Sizes for a Poc OpenShift Cluster

Node	Flavor	vCPU	RAM (GiB)
Control plane node (etcd)	<code>m4.xlarge</code>	4	16
Compute node	<code>m4.large</code>	2	8

AWS Instance Sizes for a Small OpenShift Cluster

Node	Flavor	vCPU	RAM (GiB)
Control plane node (etcd)	<code>m5.2xlarge</code>	8	32
Compute node	<code>m5.4xlarge</code>	16	64

AWS Instance Sizes for a Medium OpenShift Cluster

Node	Flavor	vCPU	RAM (GiB)
Control plane node (etcd)	<code>r5.4xlarge</code>	16	128
Compute node	<code>m5.8xlarge</code>	32	128

AWS Instance Sizes for a Large OpenShift Cluster

Node	Flavor	vCPU	RAM (GiB)
Control plane node (etcd)	<code>r5.8xlarge</code>	32	256
Compute node	<code>m5.12xlarge</code>	48	192

Use the data shown in the previous tables only as a reference. The AWS EC2 instance sizes to use in OpenShift production clusters depend on many variables, such as:

- The size of the OpenShift cluster.
- The infrastructure services that the OpenShift cluster must run.
- The application workload types that the OpenShift cluster must run.

- The maximum application requests that the OpenShift cluster must support.

**Note**

For more information, refer to the *Planning your environment according to object maximums* section of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/scalability_and_performance

The following example demonstrates the customization of the AWS EC2 instance sizes for a small production cluster.

```
[user@demo ~]$ cat ${HOME}/ocp4-aws-cluster/install-config.yaml
apiVersion: v1
baseDomain: example.com
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform:
    aws:
      type: m5.4xlarge
    replicas: 3
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    aws:
      type: m5.2xlarge
    replicas: 3
...output omitted...
platform:
  aws:
    region: us-east-2
publish: External
pullSecret: '{"auths":{"cloud.openshift.com": ...}}
sshKey: |
  ssh-rsa AAAA....
```

**Note**

For more information, refer to the *Amazon EC2 Instance Types* section of the Amazon Web Services documentation at <https://aws.amazon.com/ec2/instance-types>.

Using High-performance Storage for etcd

By default, the OpenShift installer provisions a 120 GiB EBS gp2 volume on each cluster node for installing RHCOS. For a large OpenShift cluster, where the etcd requires a high I/O performance, the gp2 volume used for the etcd storage is not enough. For the best etcd reliability, the lowest consistent latency storage technology is preferable.

The following table describes a comparison between the Amazon EBS volumes types.

Amazon EBS Volume Types

Volume type	gp2	gp3	io1	io2 Block Express
Category	General Purpose SSD	General Purpose SSD	Provisioned IOPS SSD	Provisioned IOPS SSD
Use case	Low-latency interactive apps	Low-latency interactive apps	Sub-millisecond latency with sustained IOPS performance	Sub-millisecond latency with sustained IOPS performance
Volume size	1 GiB - 16 TiB	1 GiB - 16 TiB	4 GiB - 16 TiB	4 GiB - 64 TiB
Max IOPS per volume	16000	16000	64000	256000
Max throughput per volume	250 MiB/s	1000 MiB/s	1000 MiB/s	4000 MiB/s



Note

For more information, refer to the *Amazon EBS volume types* section of the Amazon Web Services documentation at <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-volume-types.html>

To configure faster storage for etcd, you must minimally set the storage type as **io1** and the IOPS to 4000 on control plane nodes. Also, increase the EBS volume size for all cluster nodes.

```
[user@demo ~]$ cat ${HOME}/ocp4-aws-cluster/install-config.yaml
apiVersion: v1
baseDomain: example.com
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 250
        type: io1
      type: m5.4xlarge
    replicas: 3
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    aws:
      rootVolume:
```

```
iops: 4000
size: 250
type: io1
type: m5.2xlarge
replicas: 3
...output omitted...
platform:
aws:
region: us-east-2
publish: External
pullSecret: '{"auths":{"cloud.openshift.com": ...'
sshKey: |
ssh-rsa AAAA....
```

**Note**

The `iops` parameter configures the input and output operations per second (IOPS) that is reserved for the volume.

Distributing Cluster Nodes on AWS Availability Zones

By default, the OpenShift installer distributes the cluster nodes across different AWS availability zones (AZs) in the same AWS region. With this configuration, the installed OpenShift cluster combines the resilience and high-availability mechanisms provided by both Red Hat OpenShift and AWS.

Red Hat recommends spanning all the cluster nodes on multiple AZs to protect against any issue that would impact a single AZ. Administrators can select the AZs where the cluster nodes are installed.

The following `install-config.yaml` configuration file contains the AWS AZs customization described in this lecture.

```
[user@demo ~]$ cat ${HOME}/ocp4-aws-cluster/install-config.yaml
apiVersion: v1
baseDomain: example.com
compute:
- architecture: amd64
hyperthreading: Enabled
name: worker
platform:
aws:
zones:
- us-east-2a
- us-east-2b
- us-east-2c
rootVolume:
iops: 2000
size: 250
type: io1
type: m5.4xlarge
replicas: 3
controlPlane:
architecture: amd64
```

```
hyperthreading: Enabled
name: master
platform:
aws:
  zones:
    - us-east-2a
    - us-east-2b
    - us-east-2c
  rootVolume:
    iops: 4000
    size: 250
    type: io1
  type: m5.2xlarge
replicas: 3
metadata:
  creationTimestamp: null
  name: ocp4-aws
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
    - 172.30.0.0/16
platform:
aws:
  region: us-east-2
publish: External
pullSecret: '{"auths":{"cloud.openshift.com": ...}}
sshKey: |
  ssh-rsa AAAA....
```

Describing OpenShift Installations on AWS

The steps to run an OpenShift installation on AWS are explained in the *Describing OpenShift Installations on a Cloud Provider* section of this document. In that section, you learned the steps to run an OpenShift installation on a generic cloud provider. These steps apply to OpenShift installation on the AWS cloud provider.

This section describes the particularities of the OpenShift installation process on the AWS cloud provider at each stage of the installation.

Resource Creation Stage

At the beginning of the installation, the OpenShift installer creates and configures the required cloud resources.

The OpenShift installer uses:

- Amazon virtual private cloud (VPC) service to create the cluster network and the cluster network services.
- Amazon NAT Gateway service to create the cluster network gateway.
- Amazon elastic load balancer (ELB) service to create the cluster load balancers.

Chapter 2 | Installing OpenShift on a Cloud Provider

- Amazon EC2 service to create the cluster compute nodes.
- Amazon EBS and Amazon S3 services for the cluster storage.

The following diagram shows the relationship between OpenShift resources and the AWS services used to create them.

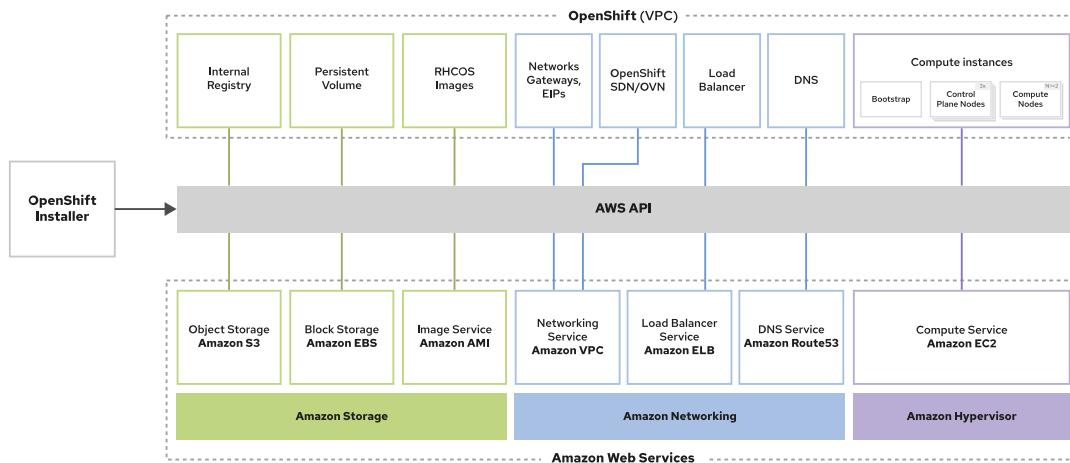


Figure 2.11: OpenShift full-stack automation installation resources on AWS

Bootstrap Stage

At this stage, the bootstrap node runs the Kubernetes API and the temporary control plane. The OpenShift installer begins to install the cluster nodes.

The following diagram explains the architecture of an OpenShift deployment running on AWS at the bootstrap stage.

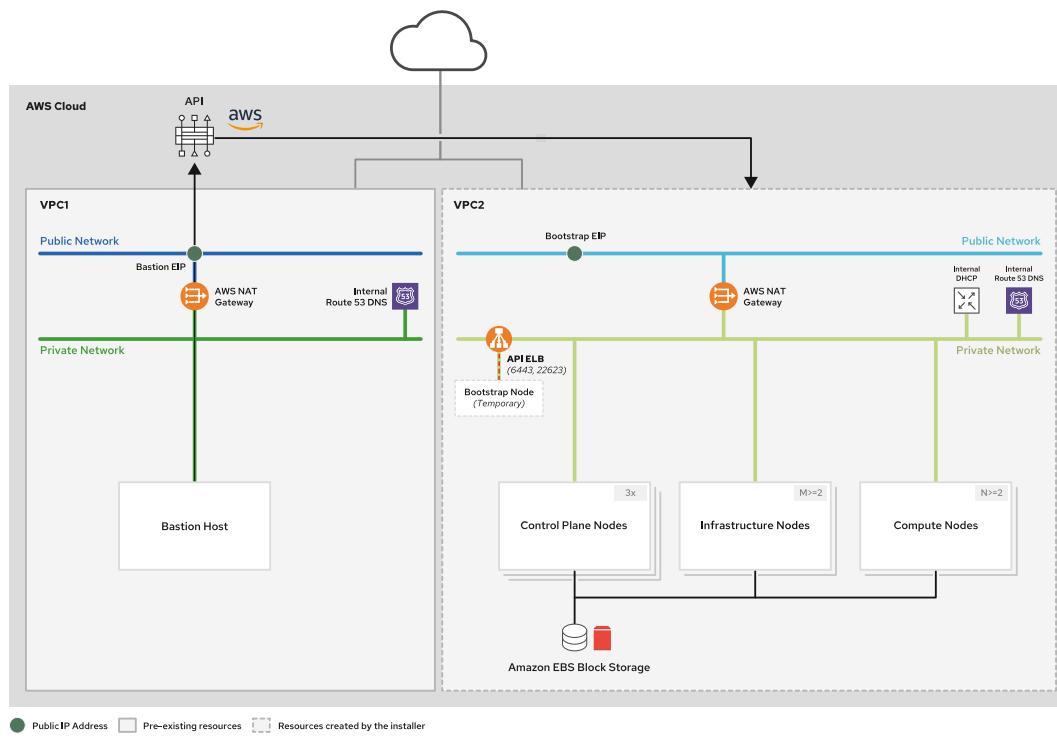


Figure 2.12: OpenShift full-stack automation installation on AWS - Bootstrap stage

Production Control Plane Stage

At this stage, the Kubernetes API and the production control plane are running on the control plane nodes. The cluster version operator (CVO), running on the production control plane, installs the operators that build the cluster and then finishes the installation.

The OpenShift installer:

- Publishes the OpenShift API ELB and APP Ingress ELB IPs on the AWS public network as public IPs (optional).
- Configures the OpenShift internal registry to use AWS S3 object storage.
- Configures a dynamic storage provider to provide persistent storage for the containerized applications using the AWS EBS storage service.

The following diagram describes the architecture of an OpenShift deployment on AWS after the installation has finished.

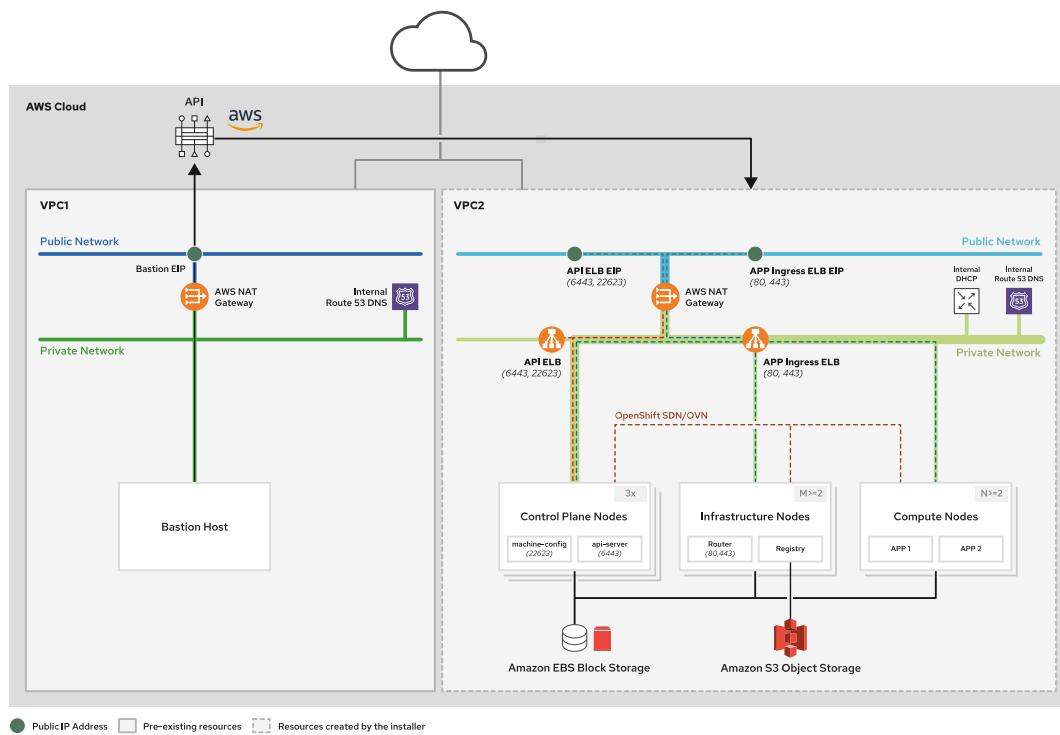


Figure 2.13: OpenShift full-stack automation installation on AWS - Final stage

During the OpenShift installation on AWS:

- The bastion host requires network connectivity to the cloud provider API to install OpenShift and to the OpenShift API for managing OpenShift after installation. It is not mandatory (but is recommended) to install the bastion host on the same cloud provider used to install OpenShift.
- The OpenShift installation process does not create infrastructure nodes; they are just compute nodes. The infrastructure nodes are compute nodes running router or registry pods.
- The OpenShift installation process creates the APP Ingress LB cloud resource containing all the compute nodes as the back-end pool members for the 80/TCP and 443/TCP ports.

Only compute nodes running router pods that are listening on these ports will pass the LB health check. Therefore, the APP Ingress LB sends network traffic only to the compute nodes running the router pods.

- The DHCP configuration also provides the NTP configuration.
- By default, the `chrony` systemd service running on each cluster node uses the public NTP pool `rhel.pool.ntp.org` to synchronize the system clock. After installation, administrators can configure the cluster nodes to use a local NTP server using the machine config operator as explained in the *Configuring chrony time service* section at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/post-installation_configuration.

Configuring Prerequisites to Install OpenShift on AWS

Video Outcome:

Configure prerequisites for installing an OpenShift cluster using the full-stack automation on AWS.



References

- For more information, refer to the *Installing on AWS* guide in the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_aws
- For more information about AWS user accounts, refer to the *Creating an IAM user* section of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_aws

► Quiz

Describing How to Install OpenShift on AWS Using Full-stack Automation

Choose the correct answers to the following questions:

- ▶ 1. Which three of the following resources are created by the installer when using the full-stack automation method on AWS? (Choose three.)
 - a. A dedicated VPC for installing OpenShift.
 - b. A DNS hosted zone in the AWS Route53 service.
 - c. The API and Ingress load balancers.
 - d. The AWS S3 object storage used for the persistent storage of the internal registry.
 - e. A secondary IAM user.

- ▶ 2. Administrators can use the pre-existing infrastructure installation method for installing OpenShift on AWS. (True or False)
 - a. True
 - b. False

- ▶ 3. Which two of the following AWS CLI tool commands can be used to verify AWS account permissions and quotas? (Choose two.)
 - a. aws sts get-caller-identity
 - b. aws iam get-account-authorization-details
 - c. aws iam policy identity
 - d. aws service-quotas list-service-quotas

- ▶ 4. Which two of the following are common customizations for large OpenShift deployments on AWS? (Choose two.)
 - a. Increase EC2 instance sizes.
 - b. Scale up the number of control plane nodes.
 - c. Use high performance storage for the etcd cluster.
 - d. Use EBS gp2 volume for the persistent storage of the internal registry.

► Solution

Describing How to Install OpenShift on AWS Using Full-stack Automation

Choose the correct answers to the following questions:

- ▶ 1. **Which three of the following resources are created by the installer when using the full-stack automation method on AWS? (Choose three.)**
 - a. A dedicated VPC for installing OpenShift.
 - b. A DNS hosted zone in the AWS Route53 service.
 - c. The API and Ingress load balancers.
 - d. The AWS S3 object storage used for the persistent storage of the internal registry.
 - e. A secondary IAM user.

- ▶ 2. **Administrators can use the pre-existing infrastructure installation method for installing OpenShift on AWS. (True or False)**
 - a. True
 - b. False

- ▶ 3. **Which two of the following AWS CLI tool commands can be used to verify AWS account permissions and quotas? (Choose two.)**
 - a. aws sts get-caller-identity
 - b. aws iam get-account-authorization-details
 - c. aws iam policy identity
 - d. aws service-quotas list-service-quotas

- ▶ 4. **Which two of the following are common customizations for large OpenShift deployments on AWS? (Choose two.)**
 - a. Increase EC2 instance sizes.
 - b. Scale up the number of control plane nodes.
 - c. Use high performance storage for the etcd cluster.
 - d. Use EBS gp2 volume for the persistent storage of the internal registry.

Demonstrating How to Install OpenShift on AWS Using Full-stack Automation

Objectives

- Install OpenShift on Amazon Web Services (AWS) using full-stack automation with common customizations.

Running OpenShift Installation Using Full-stack Automation on Amazon Web Services (AWS)

In the previous section, you learned how to prepare an OpenShift installation on AWS using the full-stack automation installation method. Additionally, you learned about the prerequisites that you must fulfill before installing OpenShift on AWS.

Following the OpenShift installation workflow explained in *Introducing OpenShift Installation Methods* section, the next step after the prerequisites fulfillment is to run the OpenShift installation from the bastion host using the `openshift-installer` binary.

For more information about the supported configuration fields for installing OpenShift on AWS, you can use the OpenShift installer `explain` option:

```
[user@demo ~]$ openshift-install explain installconfig.platform.aws
KIND:     InstallConfig
VERSION:  v1

RESOURCE: <object>
AWS is the configuration used when installing on AWS.

FIELDS:
amIID <string>
AMIID is the AMI that should be used to boot machines for the cluster. If set, the AMI should belong to the same region as the cluster.

defaultMachinePlatform <object>
DefaultMachinePlatform is the default configuration used when installing on AWS for machine pools which do not define their own platform configuration.

region <string> -required-
Region specifies the AWS region where the cluster will be created.

serviceEndpoints <[]object>
ServiceEndpoints list contains custom endpoints which override the default service endpoint of AWS Services. A service can have only one ServiceEndpoint.
ServiceEndpoint stores the configuration for services to override existing AWS Services defaults.

subnets <[]string>
Subnets specifies existing subnets (by ID) where cluster resources will be created. Leave unset to have the installer create subnets in a new VPC on your behalf.
```

```
userTags <object>
  UserTags additional keys and values that the installer will add as tags to
  all resources that it creates. Resources created by the cluster itself may not
  include these tags.
```

These are the supported fields that you can use in the `platform.aws` section of the installation configuration file `install-config.yaml`.

```
[user@demo ~]$ cat ${HOME}/ocp4-aws-cluster/install-config.yaml
apiVersion: v1
baseDomain: example.com
...output omitted...
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
    - 172.30.0.0/16
platform:
  aws:
    region: us-east-2
  publish: External
  pullSecret: '{"auths":...}'
  sshKey: |
    ssh-rsa AA...
```

Creating OpenShift Installation Assets

As explained elsewhere in this course, the main assets created by the OpenShift installer are the installation configuration file, the Kubernetes manifests, and the ignition configuration files.

The OpenShift installer transforms the installation configuration file into Kubernetes manifests, and then wraps the manifests into ignition configuration files. The OpenShift installer uses these ignition configuration files to create the OpenShift cluster.

Creating Installation Directory

Before running the OpenShift installer, create an installation directory to store all the installation assets.

```
[user@demo ~]$ mkdir ${HOME}/ocp4-aws-cluster
```

Generating Installation Configuration File

After creating the installation directory, run the OpenShift installer binary to generate the first installation asset: the installation configuration file `install-config.yaml`. The installer prompts you for the necessary cluster information and then creates the `install-config.yaml` file accordingly.

```
[user@demo ~]$ openshift-install create install-config \
> --dir=${HOME}/ocp4-aws-cluster
? SSH Public Key /home/user/.ssh/ocp4-aws-key.pub
? Platform aws
INFO Credentials loaded from the "default" profile in file "/home/user/.aws/
credentials"
? Region us-east-2
? Base Domain example.com
? Cluster Name ocp4-aws
? Pull Secret [? for help] ++++++
INFO Install-Config created in: /home/user/ocp4-aws-cluster
```

```
[user@demo ~]$ find ${HOME}/ocp4-aws-cluster
/home/user/ocp4-aws-cluster
/home/user/ocp4-aws-cluster/.openshift_install.log
/home/user/ocp4-aws-cluster/.openshift_install_state.json
/home/user/ocp4-aws-cluster/install-config.yaml
...output omitted...
```

The installer creates the `install-config.yaml` file, including the information provided and some cluster default values.

```
[user@demo ~]$ cat ${HOME}/ocp4-aws-cluster/install-config.yaml
apiVersion: v1
baseDomain: example.com
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 3
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform: {}
  replicas: 3
metadata:
  creationTimestamp: null
  name: ocp4-aws
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
    networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
```

```
platform:  
  aws:  
    region: us-east-2  
  publish: External  
  pullSecret: '{"auths":...}'  
  sshKey: |  
    ssh-rsa AA...
```

Customizing the Installation Configuration File

You can customize your installation by modifying the `install-config.yaml` file. The following are common customizations for installing OpenShift production clusters on AWS:

- Increase the EC2 instance sizes for the cluster nodes.
- Increase the EBS disk size for the cluster nodes.
- Use high-performance storage for the etcd cluster.
- Distribute the cluster nodes across different AWS availability zones (AZs).

The following `install-config.yaml` file highlights some customizations:

```
[user@demo ~]$ cat ${HOME}/ocp4-aws-cluster/install-config.yaml  
apiVersion: v1  
baseDomain: example.com  
compute:  
  - architecture: amd64  
    hyperthreading: Enabled  
    name: worker  
    platform:  
      aws:  
        zones:  
          - us-east-2a  
          - us-east-2b  
          - us-east-2c  
        rootVolume:  
          iops: 2000  
          size: 250  
          type: io1  
        type: m5.4xlarge  
    replicas: 3  
  controlPlane:  
    architecture: amd64  
    hyperthreading: Enabled  
    name: master  
    platform:  
      aws:  
        zones:  
          - us-east-2a  
          - us-east-2b  
          - us-east-2c  
        rootVolume:  
          iops: 4000  
          size: 250  
          type: io1
```

```
type: m5.2xlarge
replicas: 3
metadata:
  creationTimestamp: null
  name: ocp4-aws
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/12
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
    - 172.30.0.0/16
platform:
  aws:
    region: us-east-2
publish: External
pullSecret: '{"auths":{"cloud.openshift.com": ...}}
sshKey: |
  ssh-rsa AAAA....
```

You can also customize the OpenShift cluster networking. The following is the default cluster networking configuration.

- The cluster network is the 10.128.0.0/14 network (10.128.0.0-10.131.255.255).
- The cluster nodes are allocated in /23 cluster network subnets (10.128.0.0/23, 10.128.2.0/23,...).

This means:

- The cluster network has (23-14 = 9 bits = $2^9 =$) 512 subnets available to assign to cluster nodes.
- Each cluster node is assigned a /23 subnet, which can host up to (32-23 = 9 bits = $2^9 - 2 =$) 510 IP addresses for the pods running on it.

Using the default cluster network configuration, you can create an OpenShift cluster with a maximum of 512 nodes, and each node can host up to 510 pods.



Note

In this case, each cluster node can host up to 510 IP addresses because you must subtract the network and broadcast addresses from the subnet.

The custom cluster networking configuration used in the `install-config.yaml` file shown previously is the following:

- The cluster network is the 10.128.0.0/12 network (10.128.0.0-10.143.255.255).
- The cluster nodes are allocated in /23 cluster network subnets (10.128.0.0/23, 10.128.2.0/23, ...).

Using this custom cluster network configuration, you can create an OpenShift cluster with a maximum of 2048 nodes, and each node can host up to 510 pods. Red Hat has tested a maximum

of 2000 compute nodes on a Red Hat OpenShift Container Platform 4.6 cluster. Before installing a larger cluster, contact Red Hat Support.

Generating Kubernetes Manifests

At this point, you have included all your cluster installation customizations in the `install-config.yaml` file. Before running the OpenShift installer to create the Kubernetes manifests from the `install-config.yaml` file, it is recommended to backup the `install-config.yaml` file. The OpenShift installer will automatically remove it once created the Kubernetes manifests.

```
[user@demo ~]$ cp ${HOME}/ocp4-aws-cluster/install-config.yaml ${HOME}/
```

```
[user@demo ~]$ openshift-install create manifests \
> --dir=${HOME}/ocp4-aws-cluster
INFO Credentials loaded from the "default" profile in file "/home/user/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: /home/user/ocp4-aws-cluster/manifests and /home/user/ocp4-aws-cluster/openshift
```

```
[user@demo ~]$ find ${HOME}/ocp4-aws-cluster/manifests
/home/user/ocp4-aws-cluster/manifests
/home/user/ocp4-aws-cluster/manifests/04-openshift-machine-config-operator.yaml
/home/user/ocp4-aws-cluster/manifests/cluster-config.yaml
...output omitted...
```

For installing OpenShift on AWS using the full-stack automation method, you have to perform all the cluster customizations using the `install-config.yaml` file. You must not modify the Kubernetes manifests once created.



Warning

Modifying Kubernetes manifests is only supported if you follow Red Hat documented procedures or Red Hat support instructions. Otherwise, this is not supported.

Generating Ignition Configuration Files

After the installer creates the Kubernetes manifests from the custom `install-config.yaml` file, the installer wraps the manifests into ignition configuration files. The ignition configuration files are the last installation assets created by the installer.

Finally, the OpenShift installer uses the ignition configuration files to create the OpenShift cluster.

```
[user@demo ~]$ openshift-install create ignition-configs \
> --dir=${HOME}/ocp4-aws-cluster
INFO Consuming Master Machines from target directory
INFO Consuming OpenShift Install (Manifests) from target directory
INFO Consuming Openshift Manifests from target directory
INFO Consuming Worker Machines from target directory
INFO Consuming Common Manifests from target directory
INFO Ignition-Configs created in: /home/user/ocp4-aws-cluster and /home/user/ocp4-aws-cluster/auth
```

```
[user@demo ~]$ find ${HOME}/ocp4-aws-cluster -name '*.ign' | xargs ls -lrt
-rw-r----- 1 user user 1732 Dec 27 19:35 /home/user/ocp4-aws-cluster/master.ign
-rw-r----- 1 user user 1732 Dec 27 19:35 /home/user/ocp4-aws-cluster/worker.ign
-rw-r----- 1 user user 307594 Dec 27 19:35 /home/user/ocp4-aws-cluster/
bootstrap.ign
```

As explained elsewhere in this course:

- The `bootstrap.ign` ignition file contains all the necessary instructions to install the bootstrap node and start the temporary control plane. The bootstrap node fetches its `bootstrap.ign` ignition file from an AWS S3 bucket, applies the configuration from that ignition file, and then runs the temporary control plane.
- During their first boot, the control plane nodes and the compute nodes also fetch their ignition files from an AWS S3 bucket. These ignition files (`master.ign` and `worker.ign`) only contain a redirect instruction to get the corresponding ignition files from the Machine Config Server (MCS) running on the OpenShift (temporary) control plane.



Warning

Modifying ignition configuration files is only supported if you follow Red Hat documented procedures or Red Hat support instructions. Otherwise, this is not supported.

Running OpenShift Installation Process

After retrieving the ignition configuration files, you can start the cluster provisioning using the OpenShift installer. You can find the installation logs in the `${install_dir}/.openshift_install.log` file. From the OpenShift installation process logs, you can differentiate the following stages:

```
[user@demo ~]$ openshift-install create cluster \
> --dir=${HOME}/ocp4-aws-cluster --log-level=debug
DEBUG OpenShift Installer 4.6.4
DEBUG Built from commit 6e02d049701437fa81521fe981405745a62c86c5
...output omitted...
INFO Consuming Bootstrap Ignition Config from target directory
INFO Consuming Worker Ignition Config from target directory
INFO Consuming Master Ignition Config from target directory
...output omitted...
INFO Creating infrastructure resources...①
DEBUG module.vpc.aws_vpc.new_vpc[0]: Creating...
DEBUG module.bootstrap.aws_iam_role.bootstrap: Creating...
```

```

DEBUG module.masters.aws_iam_role.master_role: Creating...
DEBUG module.bootstrap.aws_s3_bucket.ignition: Creating...
DEBUG module.vpc.aws_vpc_dhcp_options.main[0]: Creating...
...output omitted...
DEBUG Apply complete! Resources: 122 added, 0 changed, 0 destroyed.
DEBUG OpenShift Installer 4.6.4
DEBUG Built from commit 6e02d049701437fa81521fe981405745a62c86c5
INFO Waiting up to 20m0s for the Kubernetes API at https://api.ocp4-aws.example.com:6443... 2
INFO API v1.19.0+9f84db3 up
INFO Waiting up to 30m0s for bootstrapping to complete... 3
...output omitted...
DEBUG Bootstrap status: complete
INFO Destroying the bootstrap resources...
...output omitted...
INFO Waiting up to 40m0s for the cluster at https://api.ocp4-aws.example.com:6443 to initialize... 4
DEBUG Still waiting for the cluster to initialize: Working towards 4.6.4: 82% complete
...output omitted...
DEBUG Cluster is initialized
INFO Waiting up to 10m0s for the openshift-console route to be created...
...output omitted...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/user/ocp4-aws-cluster/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.ocp4-aws.example.com
INFO Login to the console with user: "kubeadmin", and password: "xxxx"
...output omitted...
INFO Time elapsed: 34m11s

```

- 1** The AWS resources creation stage
- 2** The bootstrap (bootkube) stage
- 3** The bootstrap (temporary control plane) stage
- 4** The production control plane stage

Monitoring OpenShift Installation Process

To monitor a OpenShift installation, you must follow the procedure explained in the *Describing the OpenShift Installation Process* chapter.

The following is a useful sequence of oc commands for monitoring the installation process:

```
[user@demo ~]$ export KUBECONFIG=${HOME}/ocp4-aws-cluster/auth/kubeconfig
```

```
[user@demo ~]$ watch 'oc get clustervers; oc get clusteroperators; \
> oc get pods --all-namespaces | grep -v -E "Running|Completed"; oc get nodes'
NAME      VERSION AVAILABLE PROGRESSING SINCE STATUS
version    False     True      13m   Working towards 4.6.4: 98% complete
```

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.6.4	False	False	True	7m59s
cloud-credential		True	False	False	11m
cluster-autoscaler	4.6.4	True	False	False	6m34s
config-operator	4.6.4	True	False	False	8m4s
console	4.6.4	False	True	False	60s
csi-snapshot-controller	4.6.4	True	False	False	7m55s
dns	4.6.4	True	False	False	7m18s
etcd	4.6.4	True	True	False	6m39s
<i>...output omitted...</i>					
NAMESPACE	NAME	READY	STATUS	RESTARTS	AGE
openshift-apiserver	apiserver-5c4b794d87	0/2	Pending	0	2m13s
openshift-authentication	oauth Openshift-7bfc	0/1	Terminating	0	58s
openshift-kube-apiserver	kube-apiserver-ip-10	0/5	Init:0/1	0	61s
NAME	STATUS	ROLES	AGE	VERSION	
ip-10-0-143-122.us-east-2	Ready	master	8m53s	v1.19.0+9f84db3	
ip-10-0-148-117.us-east-2	Ready	worker	99s	v1.19.0+9f84db3	
ip-10-0-167-115.us-east-2	Ready	worker	98s	v1.19.0+9f84db3	
ip-10-0-186-189.us-east-2	Ready	master	8m59s	v1.19.0+9f84db3	
ip-10-0-208-252.us-east-2	Ready	master	8m58s	v1.19.0+9f84db3	
ip-10-0-220-24.us-east-2	Ready	worker	98s	v1.19.0+9f84db3	

The `oc get events` command prints the OpenShift event messages in real-time, so it is useful for monitoring installation progress and background events in detail.

```
[user@demo ~]$ oc get events -A -w
...output omitted...
openshift-infra      0s      Warning NetworkNotReady
```

Troubleshooting OpenShift Installation Process

The troubleshooting process depends on the stage in which the OpenShift installation fails.

Troubleshooting AWS Resources Creation Stage

At this stage, the OpenShift installation process creates AWS resources. The following are the relevant installation logs at this stage.

```
[user@demo ~]$ openshift-install create cluster \
> --dir=${HOME}/ocp4-aws-cluster --log-level=debug
DEBUG OpenShift Installer 4.6.4
DEBUG Built from commit 6e02d049701437fa81521fe981405745a62c86c5
...output omitted...
INFO Consuming Bootstrap Ignition Config from target directory
INFO Consuming Worker Ignition Config from target directory
INFO Consuming Master Ignition Config from target directory
...output omitted...
INFO Creating infrastructure resources...
①
...output omitted...
```

- ❶ Error message location on the `Creating infrastructure resources` log section.

You can use the AWS CLI tool `aws` to review the resources created.

```
[user@demo ~]$ aws ec2 describe-instances \
> --filters Name=tag:Name,Values="*ocp4-aws*" --output table
```

You can also use the AWS web console to review the resources created.

Errors at this stage are typically related to the AWS prerequisites required before starting the OpenShift installation process:

- Router53 DNS domain
- AWS account credentials
- AWS account limits and permissions

Troubleshooting Bootstrap (Bootkube) Stage

At this stage:

- The OpenShift installation process has deployed the bootstrap node.
- The `release-image` systemd service running on the bootstrap node downloads the container images required to start the temporary control plane.
- The `bootkube` systemd service, running on the bootstrap node, starts the temporary control plane.
- The OpenShift installation process waits for the Kubernetes API, running on the bootstrap node, to become available.

The following are the relevant installation logs at this stage.

```
[user@demo ~]$ openshift-install create cluster \
> --dir=${HOME}/ocp4-aws-cluster --log-level=debug
DEBUG OpenShift Installer 4.6.4
DEBUG Built from commit 6e02d049701437fa81521fe981405745a62c86c5
...output omitted...
INFO Consuming Bootstrap Ignition Config from target directory
INFO Consuming Worker Ignition Config from target directory
INFO Consuming Master Ignition Config from target directory
...output omitted...
INFO Creating infrastructure resources...
DEBUG module.vpc.aws_vpc.new_vpc[0]: Creating...
DEBUG module.bootstrap.aws_iam_role.bootstrap: Creating...
DEBUG module.masters.aws_iam_role.master_role: Creating...
DEBUG module.bootstrap.aws_s3_bucket.ignition: Creating...
DEBUG module.vpc.aws_vpc_dhcp_options.main[0]: Creating...
...output omitted...
DEBUG Apply complete! Resources: 122 added, 0 changed, 0 destroyed.
DEBUG OpenShift Installer 4.6.4
DEBUG Built from commit 6e02d049701437fa81521fe981405745a62c86c5
```

```
INFO Waiting up to 20m0s for the Kubernetes API at https://api.ocp4-aws.example.com:6443...
❶
...output omitted...
```

- ❶ Error message location on the Waiting for the Kubernetes API log section.

For troubleshooting at this stage, you must get the logs from the bootstrap node. You can log in to the bootstrap node from the bastion host using SSH to inspect the journal and container logs. You can get the bootstrap node public IP using the AWS CLI tool aws. Then, using the SSH key generated for the installation process, you can connect to the bootstrap node.

```
[user@demo ~]$ aws ec2 describe-instances \
> --filters Name>tag:Name,Values="*ocp4-aws*" --output table \
> | grep -B30 -i bootstrap | grep PublicDnsName
PublicDnsName      |  ec2-3-21-104-144.us-east-2.compute.amazonaws.com
```

```
[user@demo ~]$ ssh -i ${HOME}/.ssh/ocp4-aws-key
core@ec2-3-17-184-44.us-east-2.compute.amazonaws.com
```



Note

The installation process deploys the bootstrap node with an AWS public IP (EIP). Once the OpenShift installation finishes, the installer removes the bootstrap node.

After you log in to the bootstrap node, run the following command to debug the `release-image` and `bootkube` services:

```
sh-4.2# journalctl -b -f -u release-image.service -u bootkube.service
```

Also, run the `cricctl` command on the bootstrap node to look for failed containers and print their logs:

```
sh-4.2# sudo bash
sh-4.2# crictl ps -a
sh-4.2# crictl logs <container_id>
```

Troubleshooting Bootstrap (Temporary Control Plane) Stage

At this stage:

- The Kubernetes API and the temporary control plane are running on the bootstrap node.
- The control plane nodes finish their installation using the ignition files fetched from the MCS running on the temporary control plane.
- The bootstrap node transfers the temporary control plane to the production control plane running on the control plane nodes.
- The OpenShift installation process waits for the bootstrap process to complete.

The following are the relevant installation logs at this stage.

```
[user@demo ~]$ openshift-install create cluster \
> --dir=${HOME}/ocp4-aws-cluster --log-level=debug
DEBUG OpenShift Installer 4.6.4
DEBUG Built from commit 6e02d049701437fa81521fe981405745a62c86c5
...output omitted...
INFO Consuming Bootstrap Ignition Config from target directory
INFO Consuming Worker Ignition Config from target directory
INFO Consuming Master Ignition Config from target directory
...output omitted...
INFO Creating infrastructure resources...
DEBUG module.vpc.aws_vpc.new_vpc[0]: Creating...
DEBUG module.bootstrap.aws_iam_role.bootstrap: Creating...
DEBUG module.masters.aws_iam_role.master_role: Creating...
DEBUG module.bootstrap.aws_s3_bucket.ignition: Creating...
DEBUG module.vpc.aws_vpc_dhcp_options.main[0]: Creating...
...output omitted...
DEBUG Apply complete! Resources: 122 added, 0 changed, 0 destroyed.
DEBUG OpenShift Installer 4.6.4
DEBUG Built from commit 6e02d049701437fa81521fe981405745a62c86c5
INFO Waiting up to 20m0s for the Kubernetes API at https://api.ocp4-
aws.example.com:6443...
INFO API v1.19.0+9f84db3 up
INFO Waiting up to 30m0s for bootstrapping to complete...
①
...output omitted...
```

- ① Error message location on the Waiting for bootstrapping to complete log section.

As in the previous section, you must get the logs from the bootstrap node for troubleshooting. At this stage, you can also use the OpenShift installer to obtain the installation logs from the bootstrap and control plane nodes.

```
[user@demo ~]$ export KUBECONFIG=${HOME}/ocp4-aws-cluster/auth/kubeconfig
```

```
[user@demo ~]$ openshift-install gather bootstrap \
> --dir=${HOME}/ocp4-aws-cluster
INFO Pulling debug logs from the bootstrap machine
INFO Bootstrap gather logs captured here "/home/user/ocp4-aws-cluster/log-
bundle-20210107135825.tar.gz"
```

```
[user@demo ~]$ cd ocp4-aws-cluster
[user@demo ~]$ tar -xvzf log-bundle-20210107135825.tar.gz
[user@demo ~]$ cd log-bundle-20210107135825
[user@demo ~]$ ls
bootstrap/ control-plane/ failed-units.txt rendered-assets/ resources/ unit-
status/
```

You can also access the Kubernetes API running on the bootstrap node to monitor the cluster installation progress. Use the standard `oc` commands from the bastion host for troubleshooting.

```
[user@demo ~]$ export KUBECONFIG=${HOME}/ocp4-aws-cluster/auth/kubeconfig
```

Chapter 2 | Installing OpenShift on a Cloud Provider

```
[user@demo ~]$ oc get nodes
ip-10-0-143-122.us-east-2.compute.internal  Ready  master  7h43m
  v1.19.0+9f84db3
ip-10-0-186-189.us-east-2.compute.internal  Ready  master  7h44m
  v1.19.0+9f84db3
ip-10-0-208-252.us-east-2.compute.internal  Ready  master  7h44m
  v1.19.0+9f84db3
```

```
[user@demo ~]$ oc get clusterversion
NAME      VERSION AVAILABLE PROGRESSING SINCE STATUS
version    False     True      8m32s Unable to apply 4.6.4: an unknown
          error has occurred: MultipleErrors
```

```
[user@demo ~]$ oc get clusteroperators
NAME                  VERSION AVAILABLE PROGRESSING DEGRADED SINCE
authentication
cloud-credential       True      False     False   10m
cluster-autoscaler
config-operator
console
csi-snapshot-controller
dns
etcd                 4.6.4  Unknown  Unknown  False   1s
...output omitted...
```

You can follow the installation process in detail by watching the cluster events.

```
[user@demo ~]$ oc get events -A -w
...output omitted...
openshift-infra        0s      Warning  NetworkNotReady
```

At this stage, you can also use the `oc adm node-logs` command to gather the cluster node logs.

```
[user@demo ~]$ oc adm node-logs ip-10-0-131-192.us-east-2.compute.internal
...output omitted...
Jan 06 22:31:58.320538 ip-10-0-131-192 hyperkube[1566]: I0106 22:31:58.320538
  1566 prober.go:126] Liveness probe for "aws-ebs-csi-driver-node-hzcb8_openshift-
cluster-csi-drivers(96c559fe-67c7-4072-9311-450dfd2f8ddb):csi-driver" succeeded
```

As an alternative, use the `oc debug node` command for troubleshooting the control plane nodes. After logging in, you can use the same commands that you used to troubleshoot the bootstrap node.

```
[user@demo ~]$ oc get nodes
ip-10-0-143-122.us-east-2.compute.internal  Ready  master  7h43m
  v1.19.0+9f84db3
ip-10-0-186-189.us-east-2.compute.internal  NotReady  master  7h44m
  v1.19.0+9f84db3
ip-10-0-208-252.us-east-2.compute.internal  Ready  master  7h44m
  v1.19.0+9f84db3
```

```
[user@demo ~]$ oc debug node/ip-10-0-186-189.us-east-2.compute.internal  
...output omitted...  
sh-4.2# chroot /host  
sh-4.2# journalctl -b -f -u kubelet.service -u crio.service  
sh-4.2# sudo bash  
sh-4.2# crictl ps -a  
sh-4.2# crictl logs <container_id>
```

Troubleshooting the Production Control Plane Stage

At this stage:

- The OpenShift installation process has transferred the temporary control plane to the production control plane running on the control plane nodes.
- The OpenShift installation process has destroyed the bootstrap node, and the installation proceeds using the production control plane.
- The cluster version operator (CVO), running on the production control plane, installs all the operators that build the OpenShift cluster and then finishes the installation.
- The OpenShift installation process waits for the CVO to report that the cluster installation is finished.

The following are the relevant installation logs at this stage.

```
[user@demo ~]$ openshift-install create cluster \  
> --dir=${HOME}/ocp4-aws-cluster --log-level=debug  
DEBUG OpenShift Installer 4.6.4  
DEBUG Built from commit 6e02d049701437fa81521fe981405745a62c86c5  
...output omitted...  
INFO Consuming Bootstrap Ignition Config from target directory  
INFO Consuming Worker Ignition Config from target directory  
INFO Consuming Master Ignition Config from target directory  
...output omitted...  
INFO Creating infrastructure resources...  
DEBUG module.vpc.aws_vpc.new_vpc[0]: Creating...  
DEBUG module.bootstrap.aws_iam_role.bootstrap: Creating...  
DEBUG module.masters.aws_iam_role.master_role: Creating...  
DEBUG module.bootstrap.aws_s3_bucket.ignition: Creating...  
DEBUG module.vpc.aws_vpc_dhcp_options.main[0]: Creating...  
...output omitted...  
DEBUG Apply complete! Resources: 122 added, 0 changed, 0 destroyed.  
DEBUG OpenShift Installer 4.6.4  
DEBUG Built from commit 6e02d049701437fa81521fe981405745a62c86c5  
INFO Waiting up to 20m0s for the Kubernetes API at https://api.ocp4-  
aws.example.com:6443...  
INFO API v1.19.0+9f84db3 up  
INFO Waiting up to 30m0s for bootstrapping to complete...  
...output omitted...  
DEBUG Bootstrap status: complete  
INFO Destroying the bootstrap resources...  
...output omitted...  
INFO Waiting up to 40m0s for the cluster at https://api.ocp4-aws.example.com:6443  
to initialize...
```

```
DEBUG Still waiting for the cluster to initialize: Working towards 4.6.4: 82%
complete
...output omitted...
①
...output omitted...
```

- ① Error message location on the Waiting for the cluster to initialize log section.

At this stage, you can apply the same troubleshooting techniques that you used in the previous section. In this case, the cluster control plane and Kubernetes API are running on control plane nodes.

**Note**

For more information, refer to the *Troubleshooting installation issues* section of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing

Installing OpenShift Using Full-stack Automation on AWS.

Video Outcome:

Install OpenShift with customizations using full-stack automation on AWS.

**References**

For more information, refer to the *Installing a cluster on AWS with customizations* section of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_aws

► Quiz

Demonstrating How to Install OpenShift on AWS Using Full-stack Automation

Choose the correct answers to the following questions:

- ▶ 1. When generating the installation configuration file `install-config.yaml` for an OpenShift installation on AWS using the `openshift-install` command, which three of the following cluster information items must you provide? (Choose three.)
 - a. The bastion host IP address
 - b. The cluster AWS region
 - c. The cluster AWS AZ
 - d. The cluster base domain
 - e. The cluster name
- ▶ 2. Which three of the following are stages when installing OpenShift on AWS? (Choose three.)
 - a. AWS resources creation stage
 - b. Bootstrap stage
 - c. AWS account permission stage
 - d. Production control plane stage
 - e. AWS account authentication stage
- ▶ 3. Which two of the following are useful commands for monitoring the OpenShift installation process? (Choose two.)
 - a. `watch 'oc get clusterversion; oc get clusteroperators; oc get pods --all-namespaces | grep -v -E "Running|Completed"; oc get nodes'`
 - b. `aws ec2 describe-instances --output table`
 - c. `oc get events -A -w`
 - d. `oc monitor cluster`
- ▶ 4. When troubleshooting an OpenShift installation, which of the following commands is useful for logging in to the cluster nodes?
 - a. `openshift-install gather bootstrap`
 - b. `oc adm node-logs`
 - c. `oc debug node`

- 5. When installing OpenShift using the full-stack automation method on AWS, which two of the following are default configurations used by the installer? (Choose two.)
- a. The cluster node instances use AWS EBS gp2 volumes for the system disks.
 - b. The cluster node instances use AWS EBS io2 volumes for the system disks.
 - c. The cluster network hosts up to 512 nodes.
 - d. Cluster network hosts up to 2048 nodes.

► Solution

Demonstrating How to Install OpenShift on AWS Using Full-stack Automation

Choose the correct answers to the following questions:

- ▶ 1. When generating the installation configuration file `install-config.yaml` for an OpenShift installation on AWS using the `openshift-install` command, which three of the following cluster information items must you provide? (Choose three.)
 - a. The bastion host IP address
 - b. The cluster AWS region
 - c. The cluster AWS AZ
 - d. The cluster base domain
 - e. The cluster name
- ▶ 2. Which three of the following are stages when installing OpenShift on AWS? (Choose three.)
 - a. AWS resources creation stage
 - b. Bootstrap stage
 - c. AWS account permission stage
 - d. Production control plane stage
 - e. AWS account authentication stage
- ▶ 3. Which two of the following are useful commands for monitoring the OpenShift installation process? (Choose two.)
 - a. `watch 'oc get clusterversion; oc get clusteroperators; oc get pods --all-namespaces | grep -v -E "Running|Completed"; oc get nodes'`
 - b. `aws ec2 describe-instances --output table`
 - c. `oc get events -A -w`
 - d. `oc monitor cluster`
- ▶ 4. When troubleshooting an OpenShift installation, which of the following commands is useful for logging in to the cluster nodes?
 - a. `openshift-install gather bootstrap`
 - b. `oc adm node-logs`
 - c. `oc debug node`

- 5. When installing OpenShift using the full-stack automation method on AWS, which two of the following are default configurations used by the installer? (Choose two.)
- a. The cluster node instances use AWS EBS gp2 volumes for the system disks.
 - b. The cluster node instances use AWS EBS io2 volumes for the system disks.
 - c. The cluster network hosts up to 512 nodes.
 - d. Cluster network hosts up to 2048 nodes.

Verifying the Installation of OpenShift on AWS

Objectives

- Assess the success of an installation of OpenShift on AWS.

Verifying OpenShift Installations on AWS

After the OpenShift installation on AWS finishes successfully, administrators must ensure that the installed cluster is healthy and ready for day-2 tasks to onboard users and applications. Administrators must follow the cluster verification steps explained in the *Describing the OpenShift Installation Process* chapter.

OpenShift Cluster Health

From the bastion host, you can perform a basic health check using the `oc` command.

- Configure the `KUBECONFIG` environment variable to authenticate against the OpenShift API with `cluster-admin` permissions.

```
[user@demo ~]$ export KUBECONFIG=${HOME}/ocp4-aws-cluster/auth/kubeconfig
```

- Verify that all the cluster nodes have the system clock synchronized with a Network Time Protocol (NTP) server.

```
[user@demo ~]$ oc debug node/ip-10-0-151-66.us-east-2.compute.internal
...output omitted...
sh-4.4# chroot /host
sh-4.4# cat /etc/chrony.conf
# Use public servers from the pool.ntp.org project.
# Please consider joining the pool (http://www.pool.ntp.org/join.html).
pool 2.rhel.pool.ntp.org iburst
...output omitted...

sh-4.4# sudo chronyc tracking
Reference ID      : 2D574C03 (ntp.devrandom.be)
Stratum          : 3
Ref time (UTC)   : Sun Feb  7 18:36:05 2021
System time      : 0.000073610 seconds slow of NTP time
Last offset      : -0.000169116 seconds
RMS offset       : 0.000708059 seconds
Frequency        : 12.811 ppm fast
Residual freq    : +0.025 ppm
Skew             : 0.708 ppm
Root delay       : 0.091646813 seconds
Root dispersion  : 0.001100251 seconds
Update interval  : 65.1 seconds
Leap status      : Normal
```

Chapter 2 | Installing OpenShift on a Cloud Provider

The chronyd systemd service running on the cluster node uses the NTP pool 2.rhel.pool.ntp.org. The system clock is synchronized with the NTP server ntp.devrandom.be.

Repeat this procedure on all the cluster nodes.

- Verify that all the cluster nodes are in a Ready status.

If a cluster node is not in a Ready status, it cannot communicate with the OpenShift control plane and is unavailable to the cluster.

```
[user@demo ~]$ oc get nodes
NAME STATUS ROLES AGE VERSION
ip-10-0-153-185.us-east-2.compute.internal Ready worker 22h v1.19.0+9f84db3
ip-10-0-156-69.us-east-2.compute.internal Ready master 22h v1.19.0+9f84db3
ip-10-0-175-189.us-east-2.compute.internal Ready master 22h v1.19.0+9f84db3
ip-10-0-184-156.us-east-2.compute.internal Ready worker 22h v1.19.0+9f84db3
ip-10-0-195-237.us-east-2.compute.internal Ready worker 22h v1.19.0+9f84db3
ip-10-0-220-19.us-east-2.compute.internal Ready master 22h v1.19.0+9f84db3
```

- Check that all the cluster nodes are reporting usage metrics.

```
[user@demo ~]$ oc adm top node
NAME CPU(cores) CPU% MEMORY(bytes) MEMORY%
ip-10-0-153-185.us-east-2.compute.internal 130m 0% 1982Mi 3%
ip-10-0-156-69.us-east-2.compute.internal 742m 9% 5457Mi 17%
ip-10-0-175-189.us-east-2.compute.internal 597m 7% 4004Mi 13%
ip-10-0-184-156.us-east-2.compute.internal 146m 0% 2574Mi 4%
ip-10-0-195-237.us-east-2.compute.internal 851m 5% 6135Mi 9%
ip-10-0-220-19.us-east-2.compute.internal 618m 8% 4445Mi 14%
```

- Ensure that there are not certificate signing requests (CSRs) pending for approval.

```
[user@demo ~]$ oc get csr | grep Pending
```

- Review that the OpenShift cluster is available and ready from the cluster version operator report.

```
[user@demo ~]$ oc get clusterversion
NAME VERSION AVAILABLE PROGRESSING SINCE STATUS
version 4.6.4 True False 22h Cluster version is 4.6.4
```

- Check that all the cluster operators are available and ready.

If the cluster is healthy, all the cluster operators should be available and not progressing unless one or more operators are still applying its configuration.

```
[user@demo ~]$ oc get clusteroperators
NAME VERSION AVAILABLE PROGRESSING DEGRADED SINCE
authentication 4.6.4 True False False 22h
cloud-credential 4.6.4 True False False 22h
cluster-autoscaler 4.6.4 True False False 22h
```

```

config-operator      4.6.4  True   False  False  22h
console            4.6.4  True   False  False  22h
csi-snapshot-controller 4.6.4  True   False  False  22h
dns                4.6.4  True   False  False  22h
etcd               4.6.4  True   False  False  22h
...output omitted...

```

- Verify that there are not any pods with scheduling or execution issues in the cluster.

```
[user@demo ~]$ oc get pods --all-namespaces | grep -v -E 'Running|Completed'
NAMESPACE  NAME    READY  STATUS    RESTARTS  AGE
```

OpenShift Etcd Health

- Ensure that all the etcd cluster members are healthy.

```
[user@demo ~]$ oc get pods -n openshift-etcd | grep etcd-ip
etcd-ip-10-0-156-69.us-east-2.compute.internal 3/3 Running  0  22h
etcd-ip-10-0-175-189.us-east-2.compute.internal 3/3 Running  0  22h
etcd-ip-10-0-220-19.us-east-2.compute.internal 3/3 Running  0  22h
```

```
[user@demo ~]$ oc rsh -n openshift-etcd \
> etcd-ip-10-0-156-69.us-east-2.compute.internal
sh-4.4# etcdctl endpoint health --cluster
https://10.0.156.69:2379 is healthy: successfully committed proposal: took=10.8ms
https://10.0.175.189:2379 is healthy: successfully committed proposal: took=11.8ms
https://10.0.220.19:2379 is healthy: successfully committed proposal: took=12.1ms
```

OpenShift API and Console Health

- Verify that the OpenShift API DNS record api.ocp4-aws.example.com is configured to use the OpenShift API ELB public IPs.

```
[user@demo ~]$ dig api.ocp4-aws.example.com
...output omitted...
;; ANSWER SECTION:
api.ocp4-aws.example.com. 60 IN A 3.139.205.90
api.ocp4-aws.example.com. 60 IN A 18.216.12.234
api.ocp4-aws.example.com. 60 IN A 3.131.198.170
...output omitted...
```

```
[user@demo ~]$ aws elbv2 describe-load-balancers | grep DNSName
"DNSName": "ocp4-aws-n6768-ext-09c36c5873bc4193.elb.us-east-2.amazonaws.com",
"DNSName": "ocp4-aws-n6768-int-dc74cad870d449a1.elb.us-east-2.amazonaws.com",
```

```
[user@demo ~]$ host \
> ocp4-aws-n6768-ext-09c36c5873bc4193.elb.us-east-2.amazonaws.com
ocp4-aws-n6768-ext-09c36c5873bc4193.elb.us-east-2.amazonaws.com has
address 18.216.12.234
ocp4-aws-n6768-ext-09c36c5873bc4193.elb.us-east-2.amazonaws.com has
address 3.131.198.170
ocp4-aws-n6768-ext-09c36c5873bc4193.elb.us-east-2.amazonaws.com has
address 3.139.205.90
```

- Ensure that the OpenShift API is available by requesting the Kubernetes version.

```
[user@demo ~]$ curl -k https://api.ocp4-aws.example.com:6443/version
...output omitted...
"gitVersion": "v1.19.0+9f84db3",
...output omitted...
```

- Check that you can connect to the OpenShift Console.

```
[user@demo ~]$ curl -kIs \
> https://console-openshift-console.apps.ocp4-aws.example.com
...output omitted...
HTTP/1.1 200 OK
...output omitted...
```

```
[user@demo ~]$ firefox https://console-openshift-console.apps.ocp4-aws.example.com
```

OpenShift Registry Health

- Ensure that the number of internal registry pods running on the OpenShift cluster matches its deployment configuration.

```
[user@demo ~]$ oc -n openshift-image-registry get deployment.apps/image-registry
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
image-registry 2/2      2           2           24h
```

- If there are multiple compute nodes, verify that each registry pod is running on a different compute node.

```
[user@demo ~]$ oc -n openshift-image-registry get pods -o wide | grep -i
^image-registry
image-registry-64889f9585-ngmkv           1/1     Running    0
  6d20h  10.132.0.13  ip-10-0-215-243.us-east-2
image-registry-64889f9585-w6445           1/1     Running    0
  6d20h  10.131.0.6  ip-10-0-133-26.us-east-2
```

- From any cluster node, check the internal registry health.

```
[user@demo ~]$ oc debug node/ip-10-0-184-156.us-east-2.compute.internal  
sh-4.4# chroot /host  
sh-4.4# curl -kIs \  
> https://image-registry.openshift-image-registry.svc:5000/healthz  
...output omitted...  
HTTP/2 200  
...output omitted...
```

- Verify that the internal registry deployment is using AWS S3 persistent storage. Also, ensure that the image registry operator management state is Managed.

```
[user@demo ~]$ oc get configs.imageregistry.operator.openshift.io cluster -o yaml  
...output omitted...  
spec:  
  managementState: Managed  
  ...output omitted...  
  storage:  
    s3:  
      bucket: ocp4-aws-n6768-image-registry-us-east-2-uklufirxkteleqjuxyjjiuae  
      encrypt: true  
      region: us-east-2  
      virtualHostedStyle: false  
  ...output omitted...
```

OpenShift Ingress Health

- Verify that the wildcard DNS record for applications *.apps.ocp4-aws.example.com is configured to use the Ingress ELB public IPs.

```
[user@demo ~]$ dig test.apps.ocp4-aws.example.com  
...output omitted...  
;; ANSWER SECTION:  
test.apps.ocp4-aws.example.com. 60 IN A 3.129.28.60  
test.apps.ocp4-aws.example.com. 60 IN A 18.221.163.32  
...output omitted...
```

```
[user@demo ~]$ aws elb describe-load-balancers | grep DNSName  
"DNSName":  
  "ad3c569731b15486a8f31214c5a2b6be-1651370221.us-east-2.elb.amazonaws.com",  
  
[user@demo ~]$ host \  
> ad3c569731b15486a8f31214c5a2b6be-1651370221.us-east-2.elb.amazonaws.com  
ad3c569731b15486a8f31214c5a2b6be-1651370221.us-east-2.elb.amazonaws.com has  
address 3.129.28.60  
ad3c569731b15486a8f31214c5a2b6be-1651370221.us-east-2.elb.amazonaws.com has  
address 18.221.163.32
```

- Check that you can access an application exposed by an OpenShift Ingress route.

```
[user@demo ~]$ oc get routes -A | grep downloads
openshift-console downloads downloads-openshift-console.apps.ocp4-aws.example.com
```

```
[user@demo ~]$ curl -kIs \
> https://downloads-openshift-console.apps.ocp4-aws.example.com
...output omitted...
HTTP/1.0 200 OK
...output omitted...
```

- Ensure that the number of router pods running on the OpenShift cluster matches its deployment configuration.

```
[user@demo ~]$ oc -n openshift-ingress get deployment.apps/router-default
NAME           READY   UP-TO-DATE   AVAILABLE   AGE
router-default  2/2     2            2           24h
```

- If there are multiple compute nodes, verify that each router pod is running on a different compute node.

```
[user@demo ~]$ oc -n openshift-ingress get pods -o wide
NAME                           READY   STATUS    RESTARTS   AGE   IP
  NODE
router-default-6b5dcc7d88-5sjdp  1/1     Running   0          6d20h  10.131.0.4
  ip-10-0-133-26.us-east-2
router-default-6b5dcc7d88-bn62l  1/1     Running   0          6d20h  10.132.0.4
  ip-10-0-215-243.us-east-2
```

OpenShift Dynamic Storage Provider Health

- Check the AWS EBS gp2 storage class status.

```
[user@demo ~]$ oc get sc
NAME          PROVISIONER          RECLAIMPOLICY  BINDINGMODE      EXPANSION AGE
gp2 (default) kubernetes.io/aws-ebs Delete        WaitForFirstConsumer true      32m
gp2-csi       ebs.csi.aws.com     Delete        WaitForFirstConsumer true      32m
```

The gp2 storage class uses the `WaitForFirstConsumer` volume binding mode. This volume binding mode delays the binding and provisioning of a `PersistentVolume` until a pod using the `PersistentVolumeClaim` is created. This configuration is immutable for this storage class.

- Verify that the gp2 storage class works as expected.

Create a simple httpd application that uses persistent storage for its `DocumentRoot` directory at `/var/www/html`.

```
[user@demo ~]$ oc new-project httpd-persistent
[user@demo ~]$ cat /tmp/httpd-persistent.yaml
---
apiVersion: v1
kind: PersistentVolumeClaim
```

```
metadata:
  name: httpd-claim
  namespace: httpd-persistent
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 3Gi
---
apiVersion: v1
kind: Pod
metadata:
  name: httpd
  namespace: httpd-persistent
spec:
  containers:
    - image: registry.redhat.io/rhel8/httpd-24:latest
      name: httpd
      ports:
        - containerPort: 8080
          name: http
          protocol: TCP
      volumeMounts:
        - mountPath: /var/www/html
          name: httpd-claim
  volumes:
    - name: httpd-claim
      persistentVolumeClaim:
        claimName: httpd-claim

[user@demo ~]$ oc create -f /tmp/httpd-persistent.yaml
persistentvolumeclaim/httpd-claim created
pod/httpd created
```

Verify that the PVC creation automatically triggers the PV provisioning and binding through the gp2 default storage class.

```
[user@demo ~]$ oc get pvc
NAME           STATUS    VOLUME      CAPACITY   ACCESS MODES  STORAGECLASS  AGE
httpd-claim    Bound     pvc-d965cb1f  3Gi        RWO          gp2          29s
```

```
[user@demo ~]$ oc rsh httpd
sh-4.4$ df -h
Filesystem      Size  Used Avail Use% Mounted on
...output omitted...
/dev/nvme2n1    2.9G  9.0M  2.9G   1% /var/www/html
...output omitted...
```

OpenShift Application Build and Deployment Test

- Build and deploy a test application to verify the OpenShift application build cycle.

```
[user@demo ~]$ oc new-project validate
[user@demo ~]$ oc new-app django-psql-example
[user@demo ~]$ oc get pods -n validate
NAME          READY   STATUS    RESTARTS   AGE
django-psql-example-1-build  0/1     Completed  0          11m
django-psql-example-1-deploy 0/1     Completed  0          10m
django-psql-example-1-vfb5l  1/1     Running   0          10m
postgresql-1-bdgkk           1/1     Running   0          11m
postgresql-1-deploy          0/1     Completed  0          11m
```

```
[user@demo ~]$ oc logs -f django-psql-example-1-build
...output omitted...
Successfully pushed image-registry.openshift-image-registry.svc:5000/validate/
django-psql-example@sha256:b97b...ff82
Push successful
```

```
[user@demo ~]$ oc logs -f django-psql-example-1-deploy
...output omitted...
--> Scaling django-psql-example-1 to 1
--> Success
```

```
[user@demo ~]$ oc get routes -n validate
NAME          HOST/PORT          PATH
SERVICES      PORT  TERMINATION WILDCARD
django-psql-example  django-psql-example-validate.apps.ocp4-aws.example.com
django-psql-example <all>        None
```

```
[user@demo ~]$ curl -Is \
> http://django-psql-example-validate.apps.ocp4-aws.example.com
...output omitted...
HTTP/1.1 200 OK
...output omitted...
```

```
[user@demo ~]$ firefox http://django-psql-example-validate.apps.ocp4-
aws.example.com
```

Verifying OpenShift Installation Customizations on AWS

After performing a basic OpenShift cluster health check, you must verify that the customizations applied during the OpenShift installation are in place.

OpenShift Cluster Nodes Distribution

- Verify that the OpenShift installation process has spanned the cluster nodes across different AZs.

```
[user@demo ~]$ oc get nodes --label-columns \
> failure-domain.beta.kubernetes.io/region,failure-domain.beta.kubernetes.io/zone
NAME                               STATUS   ROLES    AGE     VERSION
REGION      ZONE
ip-10-0-131-192.us-east-2.compute.internal Ready    master   6d20h
v1.19.0+9f84db3 us-east-2  us-east-2a
ip-10-0-133-26.us-east-2.compute.internal Ready    worker   6d20h
v1.19.0+9f84db3 us-east-2  us-east-2a
ip-10-0-161-55.us-east-2.compute.internal Ready    master   6d20h
v1.19.0+9f84db3 us-east-2  us-east-2b
ip-10-0-168-121.us-east-2.compute.internal Ready    worker   6d20h
v1.19.0+9f84db3 us-east-2  us-east-2b
ip-10-0-213-142.us-east-2.compute.internal Ready    master   6d20h
v1.19.0+9f84db3 us-east-2  us-east-2c
ip-10-0-215-243.us-east-2.compute.internal Ready    worker   6d20h
v1.19.0+9f84db3 us-east-2  us-east-2c
```

OpenShift Cluster Nodes Instance Sizes

- Verify the AWS instance sizes of the cluster nodes.

```
[user@demo ~]$ oc get nodes --label-columns \
> beta.kubernetes.io/instance-type
NAME                               STATUS   ROLES    AGE     VERSION
INSTANCE-TYPE
ip-10-0-131-192.us-east-2.compute.internal Ready    master   6d20h
v1.19.0+9f84db3  m5.2xlarge
ip-10-0-133-26.us-east-2.compute.internal Ready    worker   6d20h
v1.19.0+9f84db3  m5.4xlarge
ip-10-0-161-55.us-east-2.compute.internal Ready    master   6d20h
v1.19.0+9f84db3  m5.2xlarge
ip-10-0-168-121.us-east-2.compute.internal Ready    worker   6d20h
v1.19.0+9f84db3  m5.4xlarge
ip-10-0-213-142.us-east-2.compute.internal Ready    master   6d20h
v1.19.0+9f84db3  m5.2xlarge
ip-10-0-215-243.us-east-2.compute.internal Ready    worker   6d20h
v1.19.0+9f84db3  m5.4xlarge
```

OpenShift Cluster Network

- Verify the OpenShift cluster network configuration.

```
[user@demo ~]$ oc get network.config/cluster -o yaml
apiVersion: config.openshift.io/v1
kind: Network
metadata:
...output omitted...
spec:
```

```
clusterNetwork:
- cidr: 10.128.0.0/12
  hostPrefix: 23
externalIP:
  policy: {}
networkType: OpenShiftSDN
serviceNetwork:
- 172.30.0.0/16
status:
  clusterNetwork:
- cidr: 10.128.0.0/12
  hostPrefix: 23
  clusterNetworkMTU: 8951
  networkType: OpenShiftSDN
  serviceNetwork:
- 172.30.0.0/16
```

OpenShift Etcd Storage Performance

- Verify the etcd storage performance.

```
[user@demo ~]$ oc get pods -n openshift-etcd | grep etcd-ip
etcd-ip-10-0-151-122.us-east-2.compute.internal          3/3     Running    0
  4h52m
etcd-ip-10-0-176-145.us-east-2.compute.internal          3/3     Running
  0           4h54m
etcd-ip-10-0-203-117.us-east-2.compute.internal          3/3     Running
  0           4h55m
```

```
[user@demo ~]$ oc rsh -n openshift-etcd \
> etcd-ip-10-0-151-122.us-east-2.compute.internal
...output omitted...

sh-4.4# etcdctl check perf --load="m"
 60 / 60 Boooooooooooooooooooooooo! 100.00% 1m0s
PASS: Throughput is 993 writes/s
PASS: Slowest request took 0.192837s
PASS: Stddev is 0.012513s
PASS

sh-4.4# etcdctl check perf --load="l"
 60 / 60 Boooooooooooooooo! 100.00% 1m0s
FAIL: Throughput too low: 6506 writes/s
PASS: Slowest request took 0.211567s
PASS: Stddev is 0.021653s
FAIL
```

The test results show that the etcd cluster performs well for a medium cluster (`--load="m"`) and fails for a large cluster (`--load="l"`). For more information about etcd performance, visit the etcd documentation page at <https://etcd.io/docs/current/op-guide/hardware/>

For more detailed etcd storage performance information, use the `fio` tool from the `etcd-perf` container to run a performance test on the control plane nodes. The performance test output

reports whether the disk is fast enough to host etcd by comparing the 99th percentile of the fsync metric captured from the run to see if it is less than **10 ms**.

```
[user@demo ~]$ oc debug node/ip-10-0-151-122.us-east-2.compute.internal
...output omitted...
sh-4.4# chroot /host
sh-4.4# podman run --volume /var/lib/etcd:/var/lib/etcd:Z
quay.io/openshift-scale/etcd-perf
{
    "fio version" : "fio-3.7",
...output omitted...
    "global options" : {
        "rw" : "write",
        "ioengine" : "sync",
        "fdatasync" : "1",
        "directory" : "/var/lib/etcd",
        "size" : "22m",
        "bs" : "2300"
    },
...output omitted...
    "write" : {
...output omitted...
        "iops" : 1273.362113,
...output omitted...
    }
}
-----
99th percentile of fsync is 970752 ns
99th percentile of the fsync is within the recommended threshold - 10 ms, the disk can be used to host etcd
```

The fio performance test result is the following:

1. This test writes 22 MiB of data in blocks of 2300 bytes on the /var/lib/etcd directory.
2. The 99th percentile of the fsync is 970752 ns, which is equivalent to **1ms** of write latency.
3. The operating system has achieved an average of **1273 IOPS** during the test.

OpenShift Machine API

- Verify the compute node provisioning and autoscaling.

When installing OpenShift on AWS using the full-stack automation method, the OpenShift installer creates and configures a MachineSet for each availability zone in the selected region.

[user@demo ~]\$ oc get machines -n openshift-machine-api					
NAME	AGE	PHASE	TYPE	REGION	ZONE
ocp4-aws-9r678-master-0	16h	Running	m5.2xlarge	us-east-2	us-east-2a
ocp4-aws-9r678-master-1	16h	Running	m5.2xlarge	us-east-2	us-east-2b
ocp4-aws-9r678-master-2	16h	Running	m5.2xlarge	us-east-2	us-east-2c
ocp4-aws-9r678-worker-us-east-2a-gq2ps	16h	Running	m5.4xlarge	us-east-2	us-east-2a
ocp4-aws-9r678-worker-us-east-2b-slp7l	16h	Running	m5.4xlarge	us-east-2	us-east-2b
ocp4-aws-9r678-worker-us-east-2c-vj7pj	16h	Running	m5.4xlarge	us-east-2	us-east-2c

[user@demo ~]\$ oc get machinesets -n openshift-machine-api					
NAME	DESIRED	CURRENT	READY	AVAILABLE	AGE
ocp4-aws-9r678-worker-us-east-2a	1	1	1	1	16h
ocp4-aws-9r678-worker-us-east-2b	1	1	1	1	16h
ocp4-aws-9r678-worker-us-east-2c	1	1	1	1	16h

[user@demo ~]\$ oc get nodes --label-columns \> failure-domain.beta.kubernetes.io/region,failure-domain.beta.kubernetes.io/zone					
NAME	REGION	ZONE	STATUS	ROLES	AGE
ip-10-0-151-177.us-east-2.compute.internal	v1.19.0+9f84db3	us-east-2	Ready	master	16h
ip-10-0-157-4.us-east-2.compute.internal	v1.19.0+9f84db3	us-east-2	Ready	worker	16h

Chapter 2 | Installing OpenShift on a Cloud Provider

```
ip-10-0-166-182.us-east-2.compute.internal  Ready  worker  16h
  v1.19.0+9f84db3  us-east-2  us-east-2b
ip-10-0-180-27.us-east-2.compute.internal  Ready  master   17h
  v1.19.0+9f84db3  us-east-2  us-east-2b
ip-10-0-205-233.us-east-2.compute.internal  Ready  master   17h
  v1.19.0+9f84db3  us-east-2  us-east-2c
ip-10-0-217-153.us-east-2.compute.internal  Ready  worker  16h
  v1.19.0+9f84db3  us-east-2  us-east-2c
```

Using the worker MachineSets, you can scale up (or down) the number of compute nodes running on the cluster.

```
[user@demo ~]$ oc scale machineset ocp4-aws-9r678-worker-us-east-2c \
> --replicas=2 -n openshift-machine-api
machineset.machine.openshift.io/ocp4-aws-9r678-worker-us-east-2c scaled
```

NAME	DESIRED	CURRENT	READY	AVAILABLE	AGE
ocp4-aws-9r678-worker-us-east-2a	1	1	1	1	17h
ocp4-aws-9r678-worker-us-east-2b	1	1	1	1	17h
ocp4-aws-9r678-worker-us-east-2c	2	2	1	1	17h

After a few minutes, the new compute node must be in a Ready status.

NAME	DESIRED	CURRENT	READY	AVAILABLE	AGE
ocp4-aws-9r678-worker-us-east-2a	1	1	1	1	17h
ocp4-aws-9r678-worker-us-east-2b	1	1	1	1	17h
ocp4-aws-9r678-worker-us-east-2c	2	2	2	2	17h

NAME	PHASE	TYPE	REGION	ZONE
ocp4-aws-9r678-master-0	Running	m5.2xlarge	us-east-2	us-east-2a 17h
ocp4-aws-9r678-master-1	Running	m5.2xlarge	us-east-2	us-east-2b 17h
ocp4-aws-9r678-master-2	Running	m5.2xlarge	us-east-2	us-east-2c 17h

```
ocp4-aws-9r678-worker-us-east-2a-gq2ps    Running   m5.4xlarge  us-east-2   us-
east-2a  17h
ocp4-aws-9r678-worker-us-east-2b-slp7l    Running   m5.4xlarge  us-east-2   us-
east-2b  17h
ocp4-aws-9r678-worker-us-east-2c-65hln  Running   m5.4xlarge  us-east-2
us-east-2c  3m41s
ocp4-aws-9r678-worker-us-east-2c-vj7pj    Running   m5.4xlarge  us-east-2   us-
east-2c  17h
```

```
[user@demo ~]$ oc get nodes --label-columns \
> failure-domain.beta.kubernetes.io/region,failure-domain.beta.kubernetes.io/zone
NAME                           STATUS  ROLES   AGE    VERSION
      REGION      ZONE
ip-10-0-151-177.us-east-2.compute.internal  Ready   master   17h
v1.19.0+9f84db3  us-east-2  us-east-2a
ip-10-0-157-4.us-east-2.compute.internal  Ready   worker   17h
v1.19.0+9f84db3  us-east-2  us-east-2a
ip-10-0-166-182.us-east-2.compute.internal  Ready   worker   17h
v1.19.0+9f84db3  us-east-2  us-east-2b
ip-10-0-180-27.us-east-2.compute.internal  Ready   master   17h
v1.19.0+9f84db3  us-east-2  us-east-2b
ip-10-0-196-32.us-east-2.compute.internal  Ready   worker   2m31s
v1.19.0+9f84db3  us-east-2  us-east-2c
ip-10-0-205-233.us-east-2.compute.internal  Ready   master   17h
v1.19.0+9f84db3  us-east-2  us-east-2c
ip-10-0-217-153.us-east-2.compute.internal  Ready   worker   17h
v1.19.0+9f84db3  us-east-2  us-east-2c
```

The OpenShift Machine API has automatically provisioned and added to the cluster a new compute node (`ip-10-0-196-32.us-east-2.compute.internal`) on the desired `us-east-2c` AWS AZ.

Video Outcome:

Verify that the OpenShift cluster is ready for day-2 tasks to onboard users and applications.



References

- For more information, refer to the *Installing on AWS* guide of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_aws

► Quiz

Verifying the Installation of OpenShift on AWS

Choose the correct answers to the following questions:

- ▶ 1. **Which three of the following are health checks that administrators must perform after an OpenShift installation on AWS? (Choose three.)**
 - a. Verify that all the cluster nodes are in a Ready status.
 - b. Check that the bootstrap node is running.
 - c. Ensure that the cluster version operator reports that the cluster is available and ready.
 - d. Check that all the cluster operators are available and ready.
 - e. Verify that the cluster has been installed using a secondary IAM user.
- ▶ 2. **When verifying an OpenShift installation, which of the following commands must be used for checking the system clock synchronization of the cluster nodes?**
 - a. ntpstat
 - b. chronyc tracking
 - c. ntpq -p
- ▶ 3. **Red Hat recommends using a debug pod to generate a sosreport from an OpenShift cluster node. (True or False)**
 - a. True
 - b. False
- ▶ 4. **When measuring the etcd storage performance, which of the following commands must be used?**
 - a. etcdctl endpoint health
 - b. etcdctl check perf
 - c. etcdctl perf etcd
- ▶ 5. **After installing OpenShift on AWS, which of the following commands must be used to verify the OpenShift Machine API integration with the AWS services?**
 - a. oc scale machines
 - b. oc edit autoscaling -o yaml
 - c. oc scale machineset

► Solution

Verifying the Installation of OpenShift on AWS

Choose the correct answers to the following questions:

- ▶ **1. Which three of the following are health checks that administrators must perform after an OpenShift installation on AWS? (Choose three.)**
 - a. Verify that all the cluster nodes are in a Ready status.
 - b. Check that the bootstrap node is running.
 - c. Ensure that the cluster version operator reports that the cluster is available and ready.
 - d. Check that all the cluster operators are available and ready.
 - e. Verify that the cluster has been installed using a secondary IAM user.

- ▶ **2. When verifying an OpenShift installation, which of the following commands must be used for checking the system clock synchronization of the cluster nodes?**
 - a. ntpstat
 - b. chronyc tracking
 - c. ntpq -p

- ▶ **3. Red Hat recommends using a debug pod to generate a sosreport from an OpenShift cluster node. (True or False)**
 - a. True
 - b. False

- ▶ **4. When measuring the etcd storage performance, which of the following commands must be used?**
 - a. etcdctl endpoint health
 - b. etcdctl check perf
 - c. etcdctl perf etcd

- ▶ **5. After installing OpenShift on AWS, which of the following commands must be used to verify the OpenShift Machine API integration with the AWS services?**
 - a. oc scale machines
 - b. oc edit autoscaling -o yaml
 - c. oc scale machineset

► Quiz

Chapter Review: Installing OpenShift on a Cloud Provider

Choose the correct answers to the following questions:

- ▶ 1. When installing OpenShift on a supported cloud provider, which three of the following are advantages of using the full-stack automation installation method? (Choose three.)
 - a. Administrators install OpenShift with minimal manual intervention.
 - b. The full-stack automation installation method is more customizable than the pre-existing infrastructure installation method.
 - c. The OpenShift installer automatically creates the required cloud resources.
 - d. The OpenShift installer automatically configures the OpenShift Machine API for cluster node autoscaling.
- ▶ 2. During the entire OpenShift installation process, the bootstrap node must be up and running. (True or False)
 - a. True
 - b. False
- ▶ 3. Which two of the following are prerequisites for installing OpenShift on the MS Azure cloud using the full-stack automation installation method? (Choose two.)
 - a. Ensure that the IAM Azure account fulfills the required quotas and permissions to install OpenShift.
 - b. Provision the bastion host on the MS Azure cloud.
 - c. Configure an HTTPD server to store the ignition configuration files.
 - d. Verify the supported MS Azure regions for installing OpenShift.
- ▶ 4. Which of the following AWS EBS volume types provides the higher maximum number of IOPS per volume?
 - a. gp2
 - b. gp3
 - c. io1
 - d. io2

► **5. When scaling up a worker machineset on an OpenShift cluster deployed on AWS, which three of the following are automatic actions performed by the OpenShift cluster? (Choose three.)**

- a. The OpenShift Machine API automatically provisions and starts an AWS EC2 instance for the new compute node.
- b. The new compute node gets its ignition configuration file and installs RHCOS.
- c. The new compute node joins the OpenShift cluster.
- d. The new compute node is configured as an infrastructure compute node.
- e. The new compute node is shut down.

► Solution

Chapter Review: Installing OpenShift on a Cloud Provider

Choose the correct answers to the following questions:

- ▶ 1. When installing OpenShift on a supported cloud provider, which three of the following are advantages of using the full-stack automation installation method? (Choose three.)
 - a. Administrators install OpenShift with minimal manual intervention.
 - b. The full-stack automation installation method is more customizable than the pre-existing infrastructure installation method.
 - c. The OpenShift installer automatically creates the required cloud resources.
 - d. The OpenShift installer automatically configures the OpenShift Machine API for cluster node autoscaling.
- ▶ 2. During the entire OpenShift installation process, the bootstrap node must be up and running. (True or False)
 - a. True
 - b. False
- ▶ 3. Which two of the following are prerequisites for installing OpenShift on the MS Azure cloud using the full-stack automation installation method? (Choose two.)
 - a. Ensure that the IAM Azure account fulfills the required quotas and permissions to install OpenShift.
 - b. Provision the bastion host on the MS Azure cloud.
 - c. Configure an HTTPD server to store the ignition configuration files.
 - d. Verify the supported MS Azure regions for installing OpenShift.
- ▶ 4. Which of the following AWS EBS volume types provides the higher maximum number of IOPS per volume?
 - a. gp2
 - b. gp3
 - c. io1
 - d. io2

► **5. When scaling up a worker machineset on an OpenShift cluster deployed on AWS, which three of the following are automatic actions performed by the OpenShift cluster? (Choose three.)**

- a. The OpenShift Machine API automatically provisions and starts an AWS EC2 instance for the new compute node.
- b. The new compute node gets its ignition configuration file and installs RHCOS.
- c. The new compute node joins the OpenShift cluster.
- d. The new compute node is configured as an infrastructure compute node.
- e. The new compute node is shut down.

Summary

- The benefits of using the full-stack automation installation method when installing OpenShift on a supported IaaS cloud provider.
- The OpenShift installation process using the full-stack automation method on a supported IaaS cloud provider.
- How to install OpenShift using the full-stack automation method on Microsoft Azure.
- How to install OpenShift using the full-stack automation method on Red Hat OpenStack Platform.
- The detailed steps for installing OpenShift using the full-stack automation method on Amazon Web Services:
 - Prepare the prerequisites for installing OpenShift.
 - Size the OpenShift installation.
 - Customize the OpenShift installation.
 - Run the OpenShift installation.
 - Monitor the OpenShift installation.
 - Troubleshoot the OpenShift installation.
 - Verify the OpenShift installation running a detailed OpenShift health check process.

Chapter 3

Installing OpenShift on a Virtualized Environment

Goal

Provision OpenShift clusters on hypervisors, with common customizations, using the full-stack automation and the pre-existing infrastructure installation methods.

Objectives

- Describe the architecture and workflow for installing OpenShift on hypervisors using the full-stack automation and pre-existing infrastructure methods.
- Describe the installation of OpenShift on vSphere using full-stack automation with common customizations.
- Describe the installation of OpenShift on vSphere using pre-existing infrastructure with common customizations.

Sections

- Introducing OpenShift Installation on Hypervisors (and Quiz)
- Describing the Installation of OpenShift on vSphere Using Full-stack Automation (and Quiz)
- Describing the Installation of OpenShift on vSphere Using Pre-existing Infrastructure (and Quiz)

Introducing OpenShift Installation on Hypervisors

Objectives

- Describe the architecture and workflow for installing OpenShift on hypervisors using the full-stack automation and pre-existing infrastructure methods.

Introducing OpenShift Installation on Hypervisors

Prior to the adoption of hypervisors, physical computers only leveraged one operating system. Although this resulted in a stable physical server that could effectively and efficiently handle requests from a single operating system, unused resources were wasted. The hypervisor provides a software layer that enables multiple operating systems to run in parallel.

A hypervisor is a specific layer of software that builds and runs virtual machines. A hypervisor is sometimes called a virtual machine (VM) monitor due to its operating system isolation features.

VM resources are logically isolated from one another. Resources such as memory and CPU are allocated and scheduled by the hypervisor to existing and new virtual machines based on VM requirements and the pool of resources available from the physical host.

The primary benefit of virtualization is enabling multiple operating systems that share the same virtualized hardware resources managed by a hypervisor. A successful deployment of OpenShift Container Platform also requires other services besides the hypervisor, such as a traffic load balancer or DNS zone management.

Introducing Hypervisors

Virtualization hypervisors are categorized as either a bare metal hypervisor or hosted hypervisor.

A native hypervisor is commonly defined as a bare metal hypervisor, due to its ability to replace the host operating system, run directly on the host's hardware, manage the guest operating systems, and schedule VM resources directly to the hardware. The following bare metal hypervisors are commonly deployed in enterprise data centers and server-based environments.

- RHV
- Microsoft Hyper-V
- vSphere ESXi

A software layer hypervisor is considered a hosted hypervisor because it can run on a standard operating system as a software layer or application.

In contrast to bare metal hypervisors, software layer hypervisors schedule VM resources against the host operating system and are executed against the hardware. They are typically suitable for personal computer users who require access to multiple operating systems. The following are examples software layer hypervisors:

- VMware Workstation
- Oracle VirtualBox

Reasons for Installing OpenShift on Hypervisors Using Full-stack Automation

On supported hypervisors, Red Hat recommends using the Full-stack installation method for the following reasons:

- Cluster administrators install OpenShift with minimal manual intervention in an opinionated installer wizard.
- The OpenShift installer creates the virtual machines, installs RHCOS, and then starts the OpenShift install process.
- The OpenShift installer uses the credentials provided to connect to the virtualization management software, upload a template, then clone that template to be used for the bootstrap and control plane nodes.
- The OpenShift installer fully integrates the OpenShift Machine API resource with the virtualization management services. The installer creates the OpenShift MachineSets resource, so automatic node provisioning and cluster autoscaling are ready to use after the installation completes.
- The Only DNS requirement are two entries, Ingress and API.

Reasons for Installing OpenShift on Hypervisors Using Pre-existing Infrastructure

Deploying OpenShift in a pre-existing infrastructure can be advantageous for the following reasons:

- The cluster administrator maintains and manages the infrastructure, allowing for detailed customizations.
- OpenShift clusters can be deployed on infrastructure specifically tailored to their needs.
- Cluster administrators have more flexibility and can more easily respond to resource requirements.

Comparing Full-stack Automation and Pre-existing Infrastructure Installation Methods

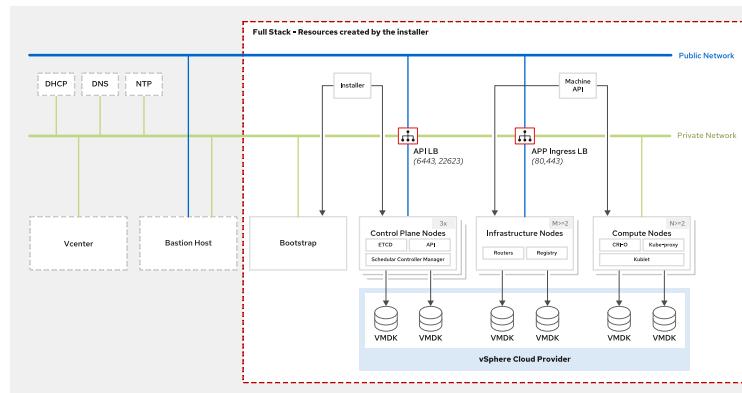
The following table displays the common deployment differences and requirements between Full-stack Automation and Pre-existing Infrastructure.

Full-stack Automation & Pre-existing Infrastructure Comparison

Action	Full-stack Automation	Pre-existing Infrastructure
Build network	Installer	User
Setup load balancer	Installer	User
Configure DNS	Installer	User
Hardware or VM provisioning	Installer	User
OS installation	Installer	User

Action	Full-stack Automation	Pre-existing Infrastructure
Generate ignition configs	Installer	Installer
OS support	Installer: RHCOS	User: RHCOS
Configure persistent storage for the internal registry	Installer	User
Configure dynamic storage provider	Installer	User
Configure node provisioning and autoscaling	Yes	Only for providers with OpenShift Machine API support.

Describing the General Architecture of a Full-stack Cluster in a Hypervisor.



A Full Stack cluster architecture consists of the following:

Bastion host (Optional)

The bastion host requires network connectivity to the cloud provider API to install OpenShift and to the OpenShift API for managing OpenShift after installation.

Control plane nodes

Manages workloads for the compute nodes and runs services required to control the cluster.

Compute nodes

Location where the actual workloads requested by Kubernetes users run and are managed.

Infrastructure nodes

Run core infrastructure components such as service brokers and logging.

Temporary bootstrap node

A temporary node that runs a minimal Kubernetes used to deploy the OpenShift control plane. It is deleted at the end of the installation.

Ingress load balancer

The virtual IP (VIP) is managed by Keepalived and is only hosted on nodes that have a router instance. Traffic destined for the *.apps Ingress VIP are passed directly to the router instance.

API load balancer

When a client creates a new request the API, HAProxy on the node hosting the API IP load balances across nodes using round robin.

Describing OpenShift Network Plug-ins for vSphere

OpenShift software defined networking (SDN) is available for layer 2 VM connectivity. However, there are vSphere plug-ins that can integrate or replace features provided by OpenShift SDN.

Networking (NSX-T)

Networking (NSX-T) is the Software Defined Network (SDN) integrated into vSphere. NSX-T creates overlay networks for VM connectivity with additional features, such as micro-segmentation, load balancers, and granular security policies. NSX-T enables any traffic to or from the VM to be firewalled at the network layer.

NSX Container Plug-in (NCP)

The NSX Container plug-in integrates NSX-T and OpenShift. It replaces OpenShift SDN and creates load balancer objects inside the cluster. NSX manager provides visibility into which pods are connected to which networks. NCP is deployed during the install process via previously created manifests.

Describing Cluster Storage

VMware developed the vSphere Cloud Provider to support the persistent storage requirements of containers. The vSphere Cloud Provider is a storage provider that provides volume plug-ins that are accessed by the OpenShift platform, backed by VMware vSAN or any supported vSphere Datastore. Virtual Machine File System (VMFS), Network File System (NFS) or Virtual Storage Area Network (vSAN) datastores, are available storage offerings.

Persistent Volumes (PV)

Persistent Volumes, or PVs, allow a cluster administrator to provision persistent storage for a cluster. PVs are resources that are not scoped to a particular project and have a life cycle that is independent of any individual pod that uses the PV. After a PV is bound to a PVC, that PV cannot then be bound to any additional PVCs. The following table displays the storage access modes available for a targeted volume plug-in.

Storage Access Modes

Volume Plug-in	ReadWriteOnce	ReadOnlyMany	ReadWriteMany
VMware vSphere	X		
Cinder	X		
NFS	X	X	X

The accessibility of storage components and features can vary depending on the type of virtualization solution.

Virtual Storage Area Network (vSAN)

A virtual storage area network, or VSAN, is a hyperconverged, software-defined, storage solution. Local resources on the physical servers are pooled, or combined together. Hybrid and all-flash versions contain a flash-based cache tier for storing and retrieving the initial set of data. Each node contains a group of disks, hard drives, or solid state drives (SSDs) that defines the capacity tier.

A flash-based cache device such as SATA or SSD are pooled together across the nodes to form the vSAN data store. Attempts to write to the VSAN data store will traverse the cache first and then are flushed to the capacity tier. The data is cached as it is read into the cache tier. For example, if the same set of blocks are repeatedly accessed for the VM, then you can conclude that they are staying in cache.

VSAN version 7 and above supports file and block storage. This enables the cluster administrator to create RWO and RWX PVCs from the VSAN.

vSphere ESXi Plug-ins

vSphere storage for Kubernetes, also referred to as vSphere Cloud Provider, is a cloud provisioner. By default, an OpenShift deployment on vSphere enables In-tree Storage.

In-tree storage integration is supported by Red Hat in vSphere 6.5 and later. OpenShift Container Platform allows the use of VMware vSphere Virtual Machine Disk (VMDK) volumes. Both VSAN and VMFS are supported volume types, and VMware vSphere volumes can be provisioned dynamically.

The vSphere disk is created by OpenShift and attached to the correct image. The deployment storage is block only and supports ReadWriteOnce. The cluster administrator can specify any data store whether block, file, or VSAN, and the provisioner creates the Virtual Machine Disk (VMDK) for that data store.

- Storage In-tree drivers
 - Formerly maintained by VMware, this is the default storage deployed when using both full-stack and pre-existing infrastructure installation methods. The plug-in creates vSphere storage using the in-tree storage drivers for vSphere included in OpenShift Container Platform, and is used when vSphere CSI drivers are not available. These drivers are supported by Red Hat on vSphere 6.5 and later.
- vSphere CSI driver
 - The Container Storage Integration (CSI) provisioner is not deployed by Red Hat. It is a Day 2 operation deployed by the customer. The plug-in creates vSphere storage using the standard Container Storage Interface. The vSphere CSI driver is provided and supported by VMware.



Note

RHEL NFS does not fully support all POSIX locking and caching semantics. Testing uncovered issues with certain apps using shared storage, such as the internal registry and HA databases. Therefore, using RHEL NFS to back PVs used by core services is not recommended. However, other NFS implementations on the marketplace might not have these issues.

Contact the NFS implementation vendor regarding completed testing against these OpenShift Container Platform core components.



References

- For more information, refer to the *Installation and update* chapter of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/architecture

► Quiz

Introducing OpenShift Installation on Hypervisors

Choose the correct answers to the following questions:

► 1. **Which of the following best describes a hypervisor?**

- a. A ready-to-run container package, including everything needed to run an application
- b. An opensource application layer that stores images
- c. A specific layer of software that builds and runs virtual machines

► 2. **Which statement describes a reason to deploy OpenShift using full-stack installation on a hypervisor?**

- a. The cluster administrator maintains the infrastructure.
- b. The cluster administrator can tailor the environment to their needs.
- c. The cluster administrator creates the initial load balancer.
- d. The cluster administrator installs OpenShift with minimal manual intervention in an opinionated manner.

► 3. **Which two of the following hypervisors are software layer hypervisors? (Choose two.)**

- a. VMware Workstation
- b. RHV
- c. Oracle VirtualBox
- d. vSphere ESXi

► 4. **Which two of the following DNS entries are required when performing a full-stack installation of OpenShift on hypervisors? (Choose two.)**

- a. API VIP
- b. ETCD VIP
- c. Ingress VIP
- d. SSL VIP

► Solution

Introducing OpenShift Installation on Hypervisors

Choose the correct answers to the following questions:

- ▶ **1. Which of the following best describes a hypervisor?**
 - a. A ready-to-run container package, including everything needed to run an application
 - b. An opensource application layer that stores images
 - c. A specific layer of software that builds and runs virtual machines

- ▶ **2. Which statement describes a reason to deploy OpenShift using full-stack installation on a hypervisor?**
 - a. The cluster administrator maintains the infrastructure.
 - b. The cluster administrator can tailor the environment to their needs.
 - c. The cluster administrator creates the initial load balancer.
 - d. The cluster administrator installs OpenShift with minimal manual intervention in an opinionated manner.

- ▶ **3. Which two of the following hypervisors are software layer hypervisors? (Choose two.)**
 - a. VMware Workstation
 - b. RHV
 - c. Oracle VirtualBox
 - d. vSphere ESXi

- ▶ **4. Which two of the following DNS entries are required when performing a full-stack installation of OpenShift on hypervisors? (Choose two.)**
 - a. API VIP
 - b. ETCD VIP
 - c. Ingress VIP
 - d. SSL VIP

Describing the Installation of OpenShift on vSphere Using Full-stack Automation

Objectives

- Describe the installation of OpenShift on vSphere using full-stack automation with common customizations.

Installing OpenShift on Using Full-stack Automation on vSphere with Common Customizations

The cluster administrator must ensure that the specific prerequisites for vSphere are completed before deploying the cluster.

The OpenShift installer prompts the cluster administrator for values, and then uses the input from those values to configure the install-config.yaml file. The default automated install-config.yaml might not be completely suitable for your environment, however, the cluster administrator can customize the install-config.yaml file. Parameters for customization are required before deploying the cluster. Finally, the OpenShift installer creates the required vSphere resources and installs an OpenShift Cluster.

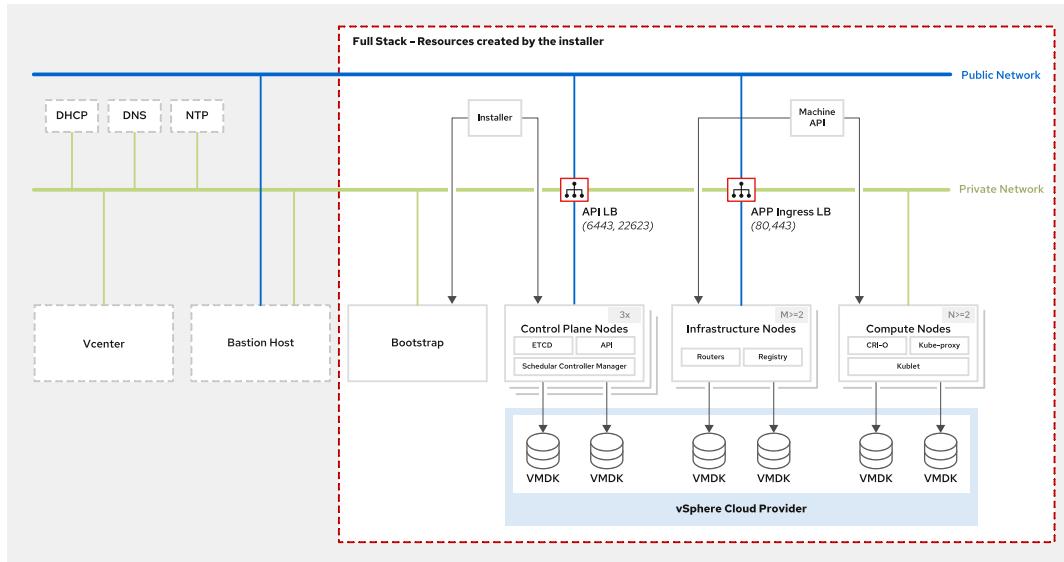
Verifying Prerequisites for OpenShift Full-stack Installation on vSphere

Before to deploying an OpenShift full-stack cluster on vSphere, cluster administrators are required to verify the following prerequisites:

- Use persistent storage with ReadWriteMany access mode for the OpenShift internal registry as a Day 2 task.
- Verify that the vSphere server has only one data center and one cluster. If there is more than one resource pool, then worker nodes will not provision during installation.
- Review the general and full-stack automation prerequisites detailed in the [Describing OpenShift Installation Prerequisites](#) lecture.
- Use VMware vSphere, version 6 or 7, to deploy an OpenShift Container Platform cluster.
- Configure vCenter for the infrastructure before deploying an OpenShift cluster.

Describing OpenShift Full-stack Architecture on vSphere

The following diagram displays the architecture of a full-stack deployment on vSphere:



This workflow description assumes that you have configured the vSphere ESXi host.

- Verify that the two DNS records, API and .apps are valid.
- Use the OpenShift installer binary interactive mode to prompt for the targeted values that the installer requires to configure the `install-config.yaml` file.
 - After invoking the `openshift-install create cluster` command, provide the SSH key when the interactive installer prompts you for the key. The SSH key is essential for authenticating nodes during the installation.
 - Provide the information required by the target platform, vSphere, including the vCenter URL, username, and password.
 - Provide values for the base domain, cluster name, and the pull secret when prompted by the installer. If you do not specify answers to the interactive queries, then the installer relies on default selections. For example, if there is only one data center or cluster, the installer will automatically use those values. You must select the appropriate data store and network for the deployment, and also ensure that the virtual IP address for API and for the `*apps` wildcard matches the entries in DNS.

The bootstrap and control plane virtual machines are configured by invoking the values specified during the OpenShift installer interactive prompts.

- The Red Hat Enterprise Linux CoreOS image is downloaded and used to create the virtual machines. After the bootstrap and control plane virtual machines are cloned, they are automatically powered on and begin the deployment.
- When bootstrapping completes, the virtual machine is automatically destroyed. Then, the control plane nodes create the worker nodes using the same template downloaded previously.
- After the worker nodes are created and the deployment completes, use the `kubeadm` password to log in to the cluster.

The Ingress and API addresses require static IP addresses.

- Communication with the cluster API requires the Ingress virtual IP.

Chapter 3 | Installing OpenShift on a Virtualized Environment

- The API requires a virtual IP for processing Ingress traffic.

Configuring DNS records for the Ingress and API static IPs is mandatory for the vCenter instance that hosts your OpenShift Container Platform cluster.

- The DNS records use the following form: <component>.<cluster_name>.<base_domain>.

A DHCP server is necessary for the network and provides persistent IP addresses to the cluster machines.

Required vCenter Account Privileges

vCenter requires specific access to an account configured with privileges to both read and create resources, such as the VMs.

- You can give administrator privileges to an account instead of granting specific privileges. However, unintended security consequences might result from granting more access than necessary.

Review the appropriate documentation prior to deployment.

Installing OpenShift Using Full-stack on vSphere Mandatory Requirement Checklist

The following lists the mandatory requirements that the cluster administrator should consider before deploying an OpenShift cluster using the full-stack automation method on vSphere.

vSphere version

The following vSphere versions are supported:

- VMware vSphere 6.5
- VMware vSphere 6.7
- VMware vSphere 7.0

vSphere data center

- vCenter has only one data center.
- vSphere data center has only one cluster.

vSphere data store

Create a data store for persistent volumes that is accessible by all the machine nodes in the data center for Day 1 installation.

vSphere VM folder

Create a new folder to contain all the cluster VMs.

Resource pool

The installer binary creates the machine in the default resource pool.

- Configuring another resource pool is **NOT** possible and must be considered when planning an OpenShift cluster deployment.

Physical hosts

Use separate physical hosts for the cluster machines to maintain high availability of the cluster.

VM / VM anti affinity

Configure anti affinity rules for control plane and infra VMs when they do not reside in the same physical host.

DRS disabled

VMware vMotion is intended to provide live migration of virtual machines between hosts while preventing downtime. VMware Distributed Resource Scheduler (DRS) is disabled for control plane and infra worker nodes.

- A node draining procedure is implemented before migrating compute worker nodes.
- In VMware DRS, virtual machines are migrated when triggered by the level of utilization on specific hosts.
- A VMware DRS migration might, in turn, trigger another VMware DRS migration if the OpenShift node being migrated becomes `NotReady` and the workloads on that node are scheduled on other nodes.



Note

Currently, vMotion is not extensively tested with OpenShift. However, there is an feature enhancement request under consideration to address formally testing vMotion with OpenShift. Red Hat does not recommend the use of vMotion at this time.

Network

Configure DHCP, DHCP IP reservations, or static IP addresses for the cluster VMs.

Image registry

File storage is installed as a storage technology for high available cluster internal registry.

- NFS on RHEL is not supported.
- Others NFS implementations are supported (NetApp, HPE, DELL, etc).
- OCS using CephFS is supported.

Monitoring & logging

Install block storage for both monitoring and logging.

Storage provisioning strategy

Choose either a static or a dynamic storage provisioning strategy.

vSphere account

Create a service account in vSphere with the roles and permissions annotated on the vSphere Storage for Kubernetes Permission web page: <https://vmware.github.io/vsphere-storage-for-kubernetes/documentation/vcp-roles.html>

Restart policy

Configure the restart policy in the following order:

1. Stop compute nodes
2. Stop infra nodes
3. Stop master nodes
4. Start master nodes

5. Start infra nodes
6. Start compute nodes

Installing OpenShift Using Full-stack Automation on vSphere Recommendation Checklist

This section describes the recommendations for a full-stack installation of OpenShift on vSphere.

- Three additional worker machines (CPU: 4 cores, RAM: 24 GB, Storage: 120 GB) dedicated for infra (registry, haproxy, etc.) are provisioned.
- The other worker machines are dedicated for computing (compute machines).

Installing OpenShift Using Full-stack Automation on vSphere

- Download the installer binary, oc tools, and pull secret from the Red Hat OpenShift Cluster Manager site.
- Download the root CA certificates from vCenter and add them to your bastion VM.
- Run the openshift-install binary to create the install-config.yaml file.
- Input values when prompted by the installer binary.
- Run the installer binary to create the cluster.



Note

At the time this course was written, there is an error on try.openshift.com/cloud.redhat.com that displays only UPI as available for vSphere. However, the pull-secret and tools on the pre-existing infrastructure web page are suitable for full-stack deployments as well.

- Install the openshift-install binary and oc tools on the bastion host.

```
[user@bastion ~]$ sudo -i
[root@bastion ~]# OCP_VERSION=4.6.4
[root@bastion ~]# MIRROR=mirror.openshift.com/pub/openshift-v4/clients
[root@bastion ~]# wget \
> https://$MIRROR/ocp/${OCP_VERSION}/openshift-install-linux-
${OCP_VERSION}.tar.gz
[root@bastion ~]# tar zxvf openshift-install-linux-${OCP_VERSION}.tar.gz \
> -C /usr/bin
[root@bastion ~]# rm -f openshift-install-linux-${OCP_VERSION}.tar.gz
[root@bastion ~]# chmod +x /usr/bin/openshift-install
[root@bastion ~]# openshift-install version
openshift-install 4.6.4
built from commit 6e02d049701437fa81521fe981405745a62c86c5
release image quay.io/openshift-release-dev/ocp-release@sha256:668...6fc
```

```
[root@bastion ~]# wget \
> https://$MIRROR/ocp/${OCP_VERSION}/openshift-client-linux-${OCP_VERSION}.tar.gz
[root@bastion ~]# tar zxvf openshift-client-linux-${OCP_VERSION}.tar.gz \
> -C /usr/bin
```

Chapter 3 | Installing OpenShift on a Virtualized Environment

```
[root@bastion ~]# rm -f openshift-client-linux-${OCP_VERSION}.tar.gz
[root@bastion ~]# chmod +x /usr/bin/oc
[root@bastion ~]# oc completion bash >/etc/bash_completion.d/openshift
[root@bastion ~]# oc version
Client Version: 4.6.4
```

- Generate an SSH key on the bastion host.

```
[root@bastion ~]# su - user
[user@bastion ~]$ ssh-keygen -f ${HOME}/.ssh/ocp46-key -N ''
```

- Download and extract the root CA certificates from vCenter.

```
[user@bastion ~]$ sudo wget \
> vcenter.sddc.vmwaremc.com/downloads/certs/download.zip
...output omitted...
Saving to: `download.zip'
100% [=====] 5,708 ---K/s in 0s
2020-10-09 22:08:15 (781 MB/s) - `download.zip` saved [6708/6708]
```

```
[user@bastion ~]$ sudo unzip download.zip
Archive: download.zip
  inflating: download/certs//lin/000cec1a.0
  inflating: download/certs//mac/000cec1a.0
  inflating: download/certs//win/000cec1a.0.crt
  inflating: download/certs//lin/000cec1a.r0
  inflating: download/certs//mac/000cec1a.r0
  inflating: download/certs//win/000cec1a.r0.cr1
```

- Update the Linux certificates to enable vCenter authentication. vCenter requires an established trust relationship with the configuration host. The installer will not connect to vCenter without this trust relationship.

```
[user@bastion ~]$ sudo cp certs/lin/* /etc/pki/ca-trust/source/anchors
[user@bastion ~]$ sudo update-ca-trust extract
```

- Create an install directory named ocp46.

```
[user@bastion ~]# mkdir ocp46
```

- Run the openshift-install binary to create the install-config.yaml file.

```
[user@demo ~]$ ./openshift-install create install-config --dir=ocp46 \
> --log-level=info
? SSH Public Key /user/.ssh/ocp4ipi.pub
? Platform vsphere
? vCenter vcenter.vmwarecloud.com
? Username clouddadmin@vmcloud.local.com
Info connecting to vcenter vcenter.vmwarecloud.com
Info Defaulting to only available datacenter: SDDC
Info Defaulting to only available cluster: cluster1
```

```
? Default Datastore: WorkloadData
? Network network-segment1
? Virtual IP Address for API 192.168.1.100
? Virtual IP Address for Ingress 192.168.1.110
? Base Domain example.com
? Cluster Name ocp4
? Pull Secret ...output omitted...
```

- Run the openshift-install binary to deploy the cluster.

```
[user@bastion ~]$ ./openshift-install create cluster --dir=ocp46 \
> --log-level=debug
```

Video



References

- For more information, refer to the *Installing on vSphere* chapter of the Red Hat OpenShift Container Platform 4.6 documentation at
https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_vsphere/index

► Quiz

Describing the Installation of OpenShift on vSphere Using Full-stack Automation

Choose the correct answers to the following questions:

- ▶ 1. In full-stack automation the cluster administrator is **not** responsible for which of the following tasks?
 - a. Verifying DNS is properly configured
 - b. Ensuring DHCP is properly configured
 - c. Creating the cluster virtual machines
- ▶ 2. In full-stack automation, how many data centers can be used with vCenter?
 - a. Only two
 - b. Two or more
 - c. Only one
- ▶ 3. In full-stack automation, which certificates are required on the bastion virtual machine before beginning the installation configuration?
 - a. Terraform Certificates
 - b. Data store Certificates
 - c. vCenter root CA certificates
- ▶ 4. In full-stack automation in vSphere, what is the recommended restart policy?
 - a. Stop infra nodes → Stop master nodes → Stop compute nodes → Start master nodes → Start infra nodes → Start compute nodes
 - b. Stop master nodes → Stop infra nodes → Stop compute nodes → Start compute nodes → Start infra nodes → Start master nodes
 - c. Stop compute nodes → Stop infra nodes → Stop master nodes → Start master nodes → Start infra nodes → Start compute nodes.

► Solution

Describing the Installation of OpenShift on vSphere Using Full-stack Automation

Choose the correct answers to the following questions:

- ▶ **1. In full-stack automation the cluster administrator is *not* responsible for which of the following tasks?**
 - a. Verifying DNS is properly configured
 - b. Ensuring DHCP is properly configured
 - c. Creating the cluster virtual machines
- ▶ **2. In full-stack automation, how many data centers can be used with vCenter?**
 - a. Only two
 - b. Two or more
 - c. Only one
- ▶ **3. In full-stack automation, which certificates are required on the bastion virtual machine before beginning the installation configuration?**
 - a. Terraform Certificates
 - b. Data store Certificates
 - c. vCenter root CA certificates
- ▶ **4. In full-stack automation in vSphere, what is the recommended restart policy?**
 - a. Stop infra nodes → Stop master nodes → Stop compute nodes → Start master nodes → Start infra nodes → Start compute nodes
 - b. Stop master nodes → Stop infra nodes → Stop compute nodes → Start compute nodes → Start infra nodes → Start master nodes
 - c. Stop compute nodes → Stop infra nodes → Stop master nodes → Start master nodes → Start infra nodes → Start compute nodes.

Describing the Installation of OpenShift on vSphere Using Pre-existing Infrastructure

Objectives

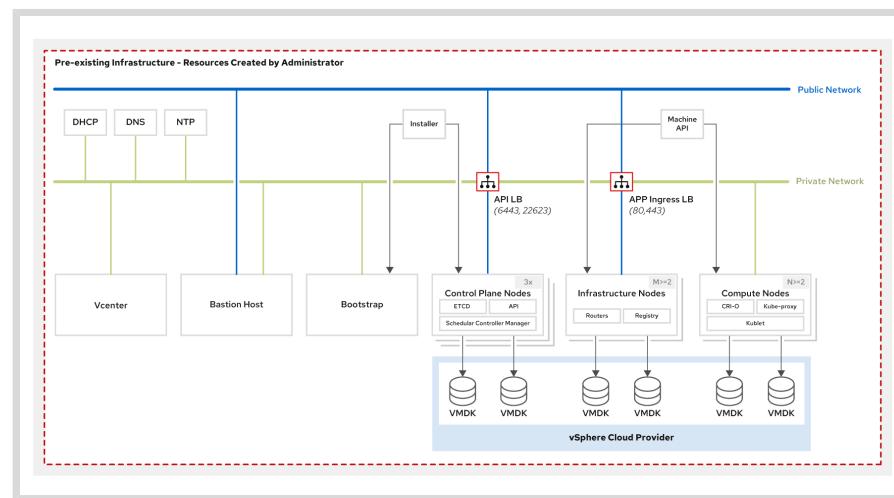
- Describe the installation of OpenShift on vSphere using pre-existing infrastructure with common customizations.

Installing OpenShift Using Pre-existing Infrastructure on vSphere

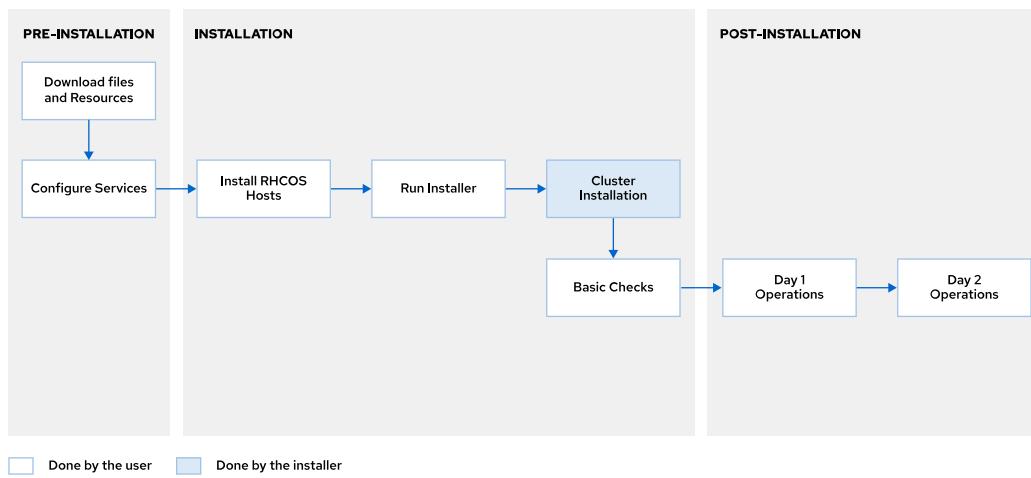
The administrator must ensure that the specific prerequisites for the vSphere infrastructure are completed before deploying the cluster.

Describing OpenShift Architecture Using Pre-existing Infrastructure on vSphere

The administrator is responsible for the architecture resources displayed in the following diagram of an OpenShift deployment using pre-existing infrastructure on vSphere:

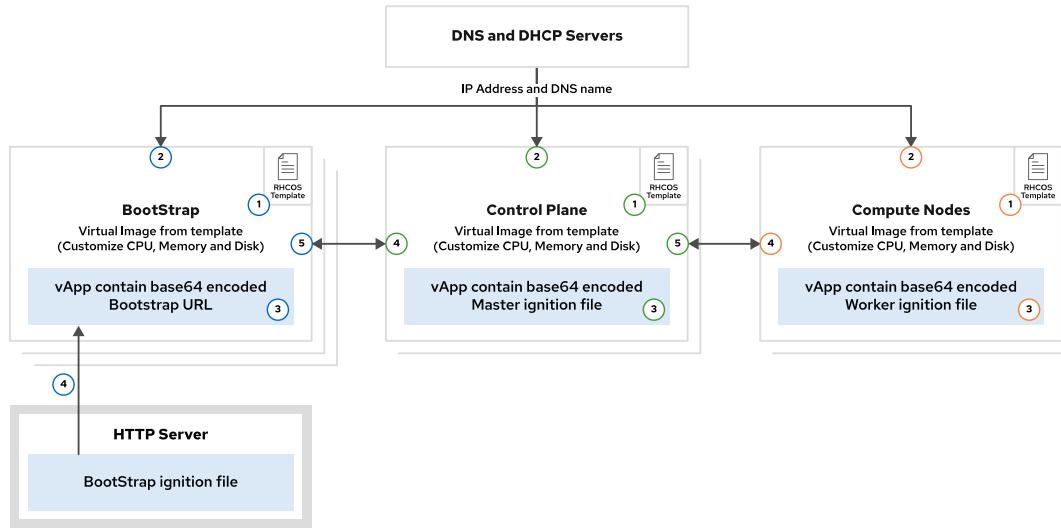


The following diagram displays the different phases of the installation process using pre-existing infrastructure.



Describing the Deployment Workflow Using Pre-existing Infrastructure on vSphere

The following diagram illustrates the workflow for deploying an OpenShift architecture using pre-existing infrastructure on vSphere.



This section provides an overview of the process steps for installing OpenShift using pre-existing infrastructure on vSphere.

Notice that the numbers on the preceding workflow diagram align to the following numbered steps.

1. Power on the bootstrap node to initialize the RHCOS template image.
2. The bootstrap node receives an IP address from the DHCP server. The DNS server contains an entry mapping the bootstrap host name to the appropriate IP address.
3. The vApp specifies the base64 encoded URL to the bootstrap ignition configuration file located on the HTTPD server.
4. The bootstrap node downloads the ignition file located at the URL specified in the vApp.

5. Using the information specified in the ignition configuration file, the control plane nodes begin installing RHCOS and receive the ignition files from the Kubernetes API MCS running on the bootstrap node.

On each of the control plane VM images:

1. The RHCOS template image completes and the control plane node boots.
2. The control plane node receives an IP address from the DHCP server.
3. The vApp specifies the base64 encoded ignition configuration information used to install the control plane nodes.
4. The control plane nodes fetch their ignition configuration files from the bootstrap node.
5. The compute nodes fetch their ignition configuration files from the Kubernetes API MCS, running on the control plane nodes after the control plane is transferred from the bootstrap node to the control plane.

Installing OpenShift Using Pre-existing Infrastructure in vSphere Mandatory Requirement Checklist

This section describes the requirements for a pre-existing infrastructure installation of OpenShift on vSphere.

Machines

The following machines are provisioned:

- One bootstrap machine (CPU: 4 cores, RAM: 16 GB, Storage: 120 GB) - This machine is removed after installation.
- Three control plane machines (CPU: 4 cores, RAM: 16 GB, Storage: 120 GB)
- Two compute machines (CPU: 2 cores, RAM: 8 GB, Storage: 120 GB)

vSphere version

The following vSphere versions are supported:

- VMware vSphere 6.5
- VMware vSphere 6.7
- VMware vSphere 7.0

Network

Each VM must have DHCP access, DHCP IP reservations, or static IP addresses configured.

Network ports

For inbound traffic, the following ports are opened:

- All machines:
 - ICMP
 - TCP: 22
- All machines except bastion:
 - TCP: 9000-9999, 10249-10259, 30000-32767

Chapter 3 | Installing OpenShift on a Virtualized Environment

- UDP: 4789, 6081, 9000-9999, 30000-32767
 - Control plane and bootstrap:
 - TCP: 2379-2380, 6443, 22623
 - Workers:
 - TCP: 443. If optional infra worker machine configuration chosen, open 443 only to infra worker
- For outbound traffic, all ports are opened.

API load balancer

- One highly-available logical layer-4 load balancer is configured with 2 virtual IPs (VIPs):
 - VIP1 (External API): Dispatch the traffic to all control plane nodes IPs and the bootstrap node IP with the following port mapping: 6443 → 6443
 - VIP2 (Internal API): Dispatch the traffic to all control plane nodes IPs and the bootstrap node IP with the following port mapping: 6443 → 6443, 22623 → 22623
- Redirection to the bootstrap node IP is removed during installation.
- Layer-7 configured load balancers are not supported.
- Both VIP health checks are configured for the following
 - The endpoint for the API server is ready.
 - The timeout is set to 30 seconds before the removal of the API server instance.
 - Probing occurs every 10 seconds, requiring two successful requests to become healthy and three unsuccessful requests to become unhealthy.
- The load balancing algorithm is stateless.

Apps load balancer

- One highly available logical layer-4 load balancer is configured with one VIP.
 - The applications VIP dispatches the traffic to all compute nodes IP with the following port mapping: 443 → 443
- Redirection to non-infra nodes is removed after the infra node configuration.
- Layer-7 configured load balancers are not supported.
- The basic VIP health check is configured.
- The load balancing algorithm is stateless.

External DNS records

The following external DNS records are required:

- API:
 - api.<cluster_name>.<base_domain>: VIP1
 - *.apps.<cluster_name>.<base_domain>: VIP2

The DNS records must be resolvable by both clients external to the cluster and from all the nodes within the cluster.

Internal DNS records

The following are the required internal DNS records:

- API:
 - api-int.<cluster_name>.<base_domain>: VIP3

The DNS records must be resolvable from all the nodes within the cluster.

HTTP server

The HTTP server must be accessible from both the bastion machine running the installation and all the cluster machines that are installed.

- If the HTTP server is not available, use the bastion machine as an HTTP server.
- Ensure that port 80 is opened in the bastion machine for inbound traffic.

Red Hat Enterprise Linux CoreOS

Provision the Red Hat Enterprise Linux CoreOS VMs from the Red Hat Enterprise Linux CoreOS OVA file or ISO with iPXE files provided by Red Hat.

Latency sensitivity

Set Latency Sensitivity to **High** for all VMs, except for the bastion machine.

Image registry storage

If file storage is installed as a storage technology for the internal registry on a highly-available cluster, then:

- NFS on RHEL is not supported.
- Other NFS implementations are supported, for example, NetApp, HPE, and DELL.
- OCS using CephFS is supported.

Storage provisioning strategy

Select either static or dynamic for the storage provisioning strategy.

vSphere data store

Create a data store for persistent volumes that are accessible by all machine nodes in the data center for Day 1 installation.

vSphere VM folder

Create a folder that contains all the VMs that are created for the deployment.

Physical hosts

Use separate physical hosts for the cluster machines to maintain high availability of the cluster.

VM / VM anti affinity

Configure anti affinity rules for control plane and infra VMs when they do not reside in the same physical host.

DRS disabled

VMware vMotion is intended to provide live migration of virtual machines between hosts while preventing downtime. VMware Distributed Resource Scheduler (DRS) is disabled for control plane and infra worker nodes.

- A node draining procedure is implemented before migrating compute worker nodes.
- In VMware DRS, virtual machines are migrated when triggered by the level of utilization on specific hosts.
- A VMware DRS migration might, in turn, trigger another VMware DRS migration if the OpenShift node being migrated becomes `NotReady` and the workloads on that node are scheduled on other nodes.

**Note**

Currently, vMotion is not extensively tested with OpenShift. However, there is an feature enhancement request under consideration to address formally testing vMotion with OpenShift. Red Hat does not recommend the use of vMotion at this time.

vSphere account

Create a service account in vSphere with the roles/permissions annotated on vSphere Storage for Kubernetes Permission web page: <https://vmware.github.io/vsphere-storage-for-kubernetes/documentation/vcp-roles.html>

Restart policy

Configure the restart policy in the following order:

1. Stop compute nodes
2. Stop infra nodes
3. Stop control plane nodes
4. Start control plane nodes
5. Start infra nodes
6. Start compute nodes

Installing an OpenShift Cluster on vSphere with Pre-existing Infrastructure.

- Download the installer binary, oc tools, and pull secret from the Red Hat OpenShift Cluster Manager site.
- Download the root CA certificates for vCenter and add them to your bastion VM.
- Download the bare metal install image.
- Verify DNS, HAProxy, and Apache Web server configurations.
- Manually create the `install-config.yaml` file.
- Create the manifest files.
- Modify the `manifest/cluster-schedular-02` file to reflect uppercase `False`.
- Create the ignition files.
- Make the ignition files available via HTTPD.
- Create `append-bootstrap.ign` to point to the URL for `bootstrap.ign`.

- Convert the ignition files using base64 encoding.
- Download the RHCOS .ova file and deploy the template.
- Clone the template to create cluster VMs.
- Add base64 encoding of ignition files to the corresponding VMs.
- Power on the VM and pass kernel line arguments to customize the cluster VMs.
- Use the openshift-install binary to monitor the bootstrapping process.
- Remove the bootstrap entry from the load balancer after bootstrapping completes.
- Use the openshift-install binary to monitor the deployment progress until completion.

Video



References

- For more information, refer to the *Installing vSphere* guide of the Red Hat OpenShift Container Platform 4.6 documentation at
https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_vsphere/index

► Quiz

Describing the Installation of OpenShift on vSphere Using Pre-existing Infrastructure

Choose the correct answers to the following questions:

- ▶ **1. When installing OpenShift on pre-existing infrastructure, which of the following tasks is the responsibility of the cluster administrator?**
 - a. Creating cluster virtual machine
 - b. Configuring a load balancer
 - c. Configuring an Apache server to distribute ignition files
 - d. Creating the ignition files
 - e. Configuring NTP
 - f. All of the above
- ▶ **2. In a pre-existing infrastructure deployment, which nodes receive traffic from the apps load balancer application VIP?**
 - a. Control-plane nodes
 - b. Compute nodes
 - c. Operator nodes
 - d. Firewall nodes
- ▶ **3. Which internal DNS record is required when using pre-existing infrastructure?**
 - a. api.<cluster_name>.<base_domain>
 - b. *apps.<cluster_name>.<base_domain>
 - c. api-int.<cluster_name>.<base_domain>
 - d. api.<base-domain>.<cluster_name>
- ▶ **4. In pre-existing infrastructure, the load balancer internal API dispatches traffic to which nodes and ports?**
 - a. To all control plane nodes IPs and the bootstrap node IP with the following port mapping:
6443 → 6443
 - b. To all control plane nodes IPs and the bootstrap node IP with the following port mapping:
6443 → 6443, 22623 → 22623
 - c. To the bastion node IP and compute nodes IPs with the following port mapping: 6443 → 6443
 - d. To the web server nodes IPs and the bootstrap node IP with the following port mapping:
80 → 8080

► Solution

Describing the Installation of OpenShift on vSphere Using Pre-existing Infrastructure

Choose the correct answers to the following questions:

- ▶ 1. When installing OpenShift on pre-existing infrastructure, which of the following tasks is the responsibility of the cluster administrator?
 - a. Creating cluster virtual machine
 - b. Configuring a load balancer
 - c. Configuring an Apache server to distribute ignition files
 - d. Creating the ignition files
 - e. Configuring NTP
 - f. All of the above

- ▶ 2. In a pre-existing infrastructure deployment, which nodes receive traffic from the apps load balancer application VIP?
 - a. Control-plane nodes
 - b. Compute nodes
 - c. Operator nodes
 - d. Firewall nodes

- ▶ 3. Which internal DNS record is required when using pre-existing infrastructure?
 - a. api.<cluster_name>.<base_domain>
 - b. *apps.<cluster_name>.<base_domain>
 - c. api-int.<cluster_name>.<base_domain>
 - d. api.<base-domain>.<cluster_name>

- ▶ 4. In pre-existing infrastructure, the load balancer internal API dispatches traffic to which nodes and ports?
 - a. To all control plane nodes IPs and the bootstrap node IP with the following port mapping: 6443 → 6443
 - b. To all control plane nodes IPs and the bootstrap node IP with the following port mapping: 6443 → 6443, 22623 → 22623
 - c. To the bastion node IP and compute nodes IPs with the following port mapping: 6443 → 6443
 - d. To the web server nodes IPs and the bootstrap node IP with the following port mapping: 80 → 8080

► Quiz

Chapter Review: Installing OpenShift on a Virtualized Environment

Choose the correct answers to the following questions:

- ▶ **1. Which two VMware vSphere versions can be used to deploy an OpenShift Container Platform cluster? (Choose two.)**
 - a. 6
 - b. 7
 - c. 8
 - d. 9

- ▶ **2. Which of the following statements concerning the removal of the bootstrap node from Haproxy is true?**
 - a. The bootstrap node is only removed in a 3 node cluster.
 - b. The bootstrap node is removed after the installation completes.
 - c. The bootstrap node is removed during the installation.
 - d. The bootstrap node is removed after the first reboot.

- ▶ **3. Which of the following statements regarding OpenShift full-stack automation installation on hypervisors is true?**
 - a. Cluster administrators install OpenShift with minimal manual intervention in an opinionated manner.
 - b. Cluster administrators install OpenShift manually.
 - c. Cluster administrators are only responsible for load balancer configurations.
 - d. Cluster administrators are responsible for manually creating virtual machine disks.

- ▶ **4. Which two of the following reasons to deploy OpenShift using pre-existing infrastructure on vSphere are true? (Choose two.)**
 - a. Cluster administrators have more flexibility and can more easily respond to resource requirements.
 - b. Cluster administrators managing their infrastructure allows for detailed customizations.
 - c. The OpenShift installer uses the credentials provided to connect to vCenter, upload a template, and then clone that template to be used for the bootstrap and control plane nodes.
 - d. There are only two DNS requirements, Ingress VIP and API VIP.

- ▶ **5. How many data centers can vCenter have when deploying OpenShift using full-stack on vSphere?**
 - a. Only one
 - b. Only two
 - c. Only five
 - d. Unlimited

- ▶ **6. Which of the following files are downloaded from vCenter for authentication?**
 - a. OpenShift Certificates
 - b. OpenStack root CA certificates
 - c. vCenter root CA certificates
 - d. OpenShift cluster SSL certificates

► Solution

Chapter Review: Installing OpenShift on a Virtualized Environment

Choose the correct answers to the following questions:

- ▶ **1. Which two VMware vSphere versions can be used to deploy an OpenShift Container Platform cluster? (Choose two.)**
 - a. 6
 - b. 7
 - c. 8
 - d. 9

- ▶ **2. Which of the following statements concerning the removal of the bootstrap node from Haproxy is true?**
 - a. The bootstrap node is only removed in a 3 node cluster.
 - b. The bootstrap node is removed after the installation completes.
 - c. The bootstrap node is removed during the installation.
 - d. The bootstrap node is removed after the first reboot.

- ▶ **3. Which of the following statements regarding OpenShift full-stack automation installation on hypervisors is true?**
 - a. Cluster administrators install OpenShift with minimal manual intervention in an opinionated manner.
 - b. Cluster administrators install OpenShift manually.
 - c. Cluster administrators are only responsible for load balancer configurations.
 - d. Cluster administrators are responsible for manually creating virtual machine disks.

- ▶ **4. Which two of the following reasons to deploy OpenShift using pre-existing infrastructure on vSphere are true? (Choose two.)**
 - a. Cluster administrators have more flexibility and can more easily respond to resource requirements.
 - b. Cluster administrators managing their infrastructure allows for detailed customizations.
 - c. The OpenShift installer uses the credentials provided to connect to vCenter, upload a template, and then clone that template to be used for the bootstrap and control plane nodes.
 - d. There are only two DNS requirements, Ingress VIP and API VIP.

- ▶ **5. How many data centers can vCenter have when deploying OpenShift using full-stack on vSphere?**
 - a. Only one
 - b. Only two
 - c. Only five
 - d. Unlimited

- ▶ **6. Which of the following files are downloaded from vCenter for authentication?**
 - a. OpenShift Certificates
 - b. OpenStack root CA certificates
 - c. vCenter root CA certificates
 - d. OpenShift cluster SSL certificates

Summary

- The rationale and options for installing OpenShift in a hypervisor.
- The architecture of an OpenShift deployment in vSphere using the full-stack automation installation method.
- The rational, requirements, and high-level steps for deploying OpenShift in vSphere using the full-stack automation installation method.
- The architecture of an OpenShift deployment in vSphere using the pre-existing infrastructure installation method.
- The rational, requirements, and high-level steps for deploying OpenShift in a hypervisor using the pre-existing infrastructure installation method.

Chapter 4

Planning to Install OpenShift Without an Infrastructure Provider

Goal

Configure the prerequisites for provisioning OpenShift clusters without integration with the underlying infrastructure.

Objectives

- Describe the architecture and workflow for installing OpenShift without integration with the underlying infrastructure.
- Provide the prerequisites to install OpenShift without integration with the underlying infrastructure.

Sections

- Introducing OpenShift Installation Without an Infrastructure Provider (and Quiz)
- Configuring Network Services and Hosts for Installing OpenShift Without an Infrastructure Provider (and Guided Exercise) (and Quiz)

Introducing OpenShift Installation Without an Infrastructure Provider

Objectives

- Describe the architecture and workflow for installing OpenShift without integration with the underlying infrastructure.

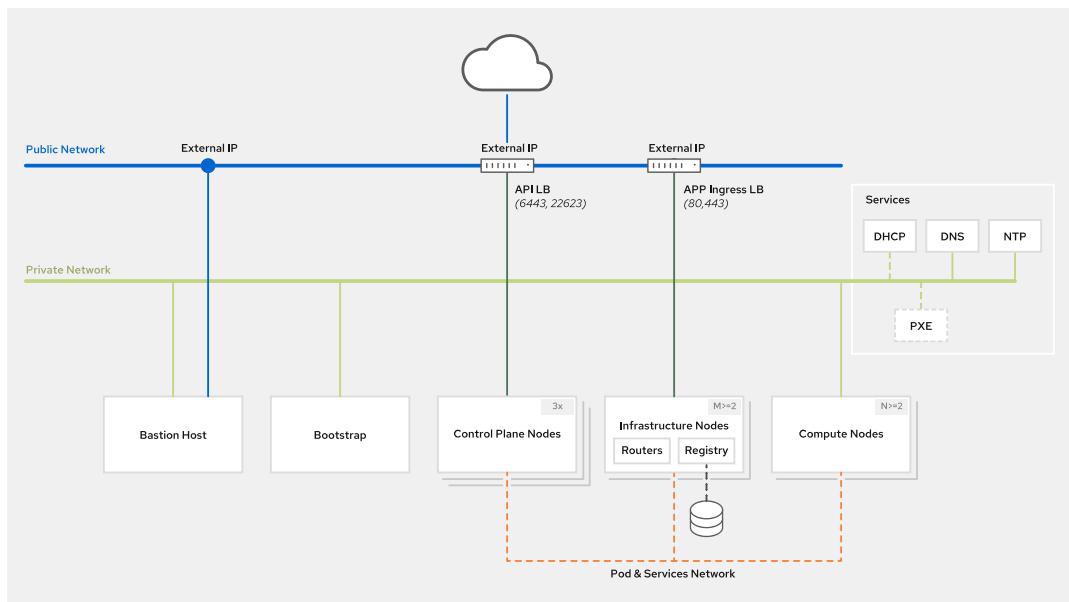
Reasons for Installing OpenShift Without Integration with an Infrastructure Provider

- In some cases, administrators have limited access to the infrastructure services and components required for performing a full-stack installation.
- In other cases, administrators need more flexibility for installing OpenShift in a heterogeneous environment.

For these or other reasons, administrators can choose to install OpenShift without integration with a defined infrastructure provider.

Architecture of an Installation Without Integration with an Infrastructure Provider

The following diagram shows the architecture of an installation without integration with an infrastructure provider:



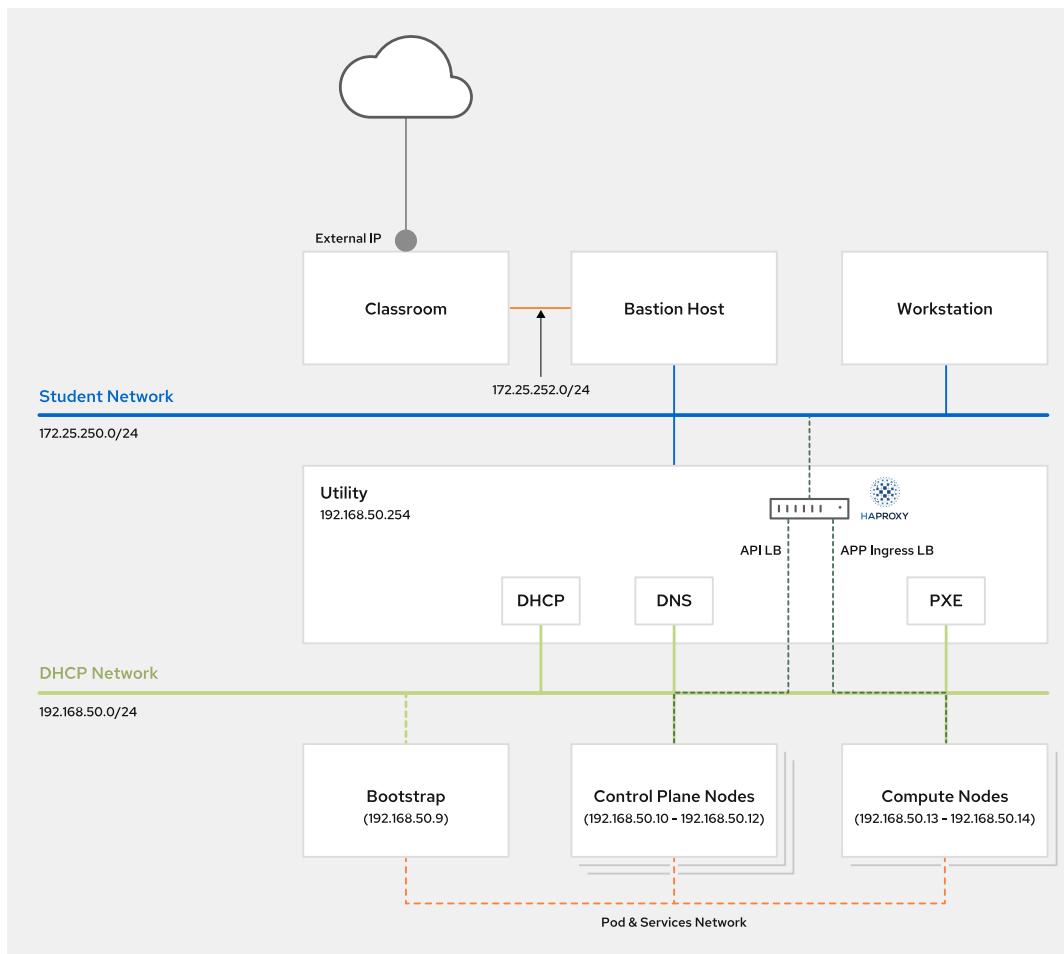
As you can see in the previous diagram, the necessary resources and services to perform a standard installation without integration with an infrastructure provider are:

- NTP** (mandatory): can be a server running inside the cluster or a public NTP server.

- **DHCP** (optional): administrators can use a DHCP server to assign IPs dynamically from a predefined range. Alternatively, administrators can manually assign the IPs to the servers.
- **PXE** (optional): administrators can use a PXE server for network provisioning. Alternatively, you can install the hosts using an ISO image.
- **DNS** (mandatory): all the servers in the cluster must be able to resolve the records from the following: table [https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_bare_metal/index#installation-infrastructure-user-infra_installing-bare-metal].
- **Load balancers** (mandatory): serves the API ports and the application ports, as shown in the following: documentation [https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_bare_metal/index#installation-infrastructure-user-infra_installing-bare-metal].
- **Network connectivity** (mandatory): the machines in the cluster must communicate with each other using the following ports and protocols: table [https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_bare_metal/index#installation-infrastructure-user-infra_installing-bare-metal].

Architecture of the Classroom Environment

The following diagram shows the architecture of the classroom environment for installing OpenShift without integration with an infrastructure provider:



In the classroom environment, you can find the necessary services running on the **utility** server, including:

- **dhcpd**: the DHCP server
- **named**: the DNS server
- **tftp**: the server for PXE booting
- **haproxy**: the load balancer for API and application services
- **httpd**: the HTTP server, used as a file server



References

- For more information, refer to the *Installing on bare metal* chapter of the Red Hat OpenShift Container Platform 4.6 documentation at
https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_bare_metal/index#installing-bare-metal

► Quiz

Introducing OpenShift Installation Without an Infrastructure Provider

Choose the correct answers to the following questions:

- ▶ 1. **Which two of the following are circumstances that could cause an administrator to choose to install OpenShift on pre-existing infrastructure? (Choose two.)**
 - a. The Administrator has full flexibility and permissions on their company infrastructure, including a private cloud environment.
 - b. The Administrator does not have access to a cloud environment and has restricted access to the company's network configuration.
 - c. The Administrator wants fine-grained control over the OpenShift installation process.
 - d. The Administrator wants a fully automated method for installing OpenShift.

- ▶ 2. **Which of the following is the Red Hat recommendation about using NTP servers when installing OpenShift in production environments?**
 - a. An NTP server is not necessary.
 - b. A private NTP server is preferred over a public one.
 - c. A public NTP server is preferred over a private one.
 - d. An NTP server is required regardless of whether it is private or public.

- ▶ 3. **Which of the following services are required for installing OpenShift on pre-existing infrastructure?**
 - a. A PXE server
 - b. A DNS server
 - c. A DHCP server

- ▶ 4. **Which of the following services are created by the OpenShift installer when installing on pre-existing infrastructure?**
 - a. An HAProxy server
 - b. A DHCP server for assigning IPs dynamically to new nodes
 - c. A keepalived server for making the APIserver load balancer highly available
 - d. None of the services mentioned above

► Solution

Introducing OpenShift Installation Without an Infrastructure Provider

Choose the correct answers to the following questions:

- ▶ **1. Which two of the following are circumstances that could cause an administrator to choose to install OpenShift on pre-existing infrastructure? (Choose two.)**
 - a. The Administrator has full flexibility and permissions on their company infrastructure, including a private cloud environment.
 - b. The Administrator does not have access to a cloud environment and has restricted access to the company's network configuration.
 - c. The Administrator wants fine-grained control over the OpenShift installation process.
 - d. The Administrator wants a fully automated method for installing OpenShift.
- ▶ **2. Which of the following is the Red Hat recommendation about using NTP servers when installing OpenShift in production environments?**
 - a. An NTP server is not necessary.
 - b. A private NTP server is preferred over a public one.
 - c. A public NTP server is preferred over a private one.
 - d. An NTP server is required regardless of whether it is private or public.
- ▶ **3. Which of the following services are required for installing OpenShift on pre-existing infrastructure?**
 - a. A PXE server
 - b. A DNS server
 - c. A DHCP server
- ▶ **4. Which of the following services are created by the OpenShift installer when installing on pre-existing infrastructure?**
 - a. An HAProxy server
 - b. A DHCP server for assigning IPs dynamically to new nodes
 - c. A keepalived server for making the APIserver load balancer highly available
 - d. None of the services mentioned above

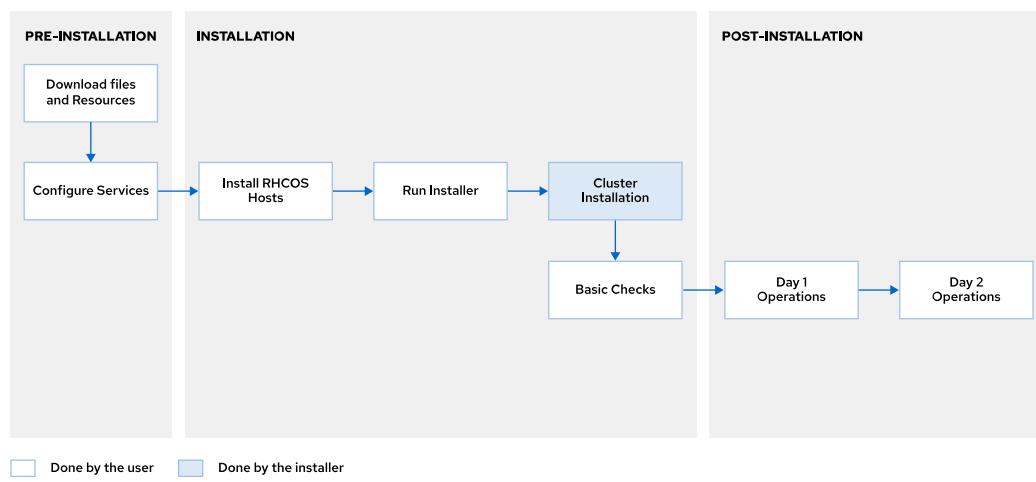
Configuring Network Services and Hosts for Installing OpenShift Without an Infrastructure Provider

Objectives

- Provide the prerequisites to install OpenShift without integration with the underlying infrastructure.

Overview of the Installation Process Using Pre-existing Infrastructure

The following diagram shows the different phases of the installation process using pre-existing infrastructure.



Downloading Resources for the Installation

- First, download both the `openshift-install` and `oc` binaries from the OpenShift clients mirror [<https://mirror.openshift.com/pub/openshift-v4/clients/ocp/>].
- Next, download your pull secret from `cloud.redhat.com` [<https://cloud.redhat.com/openshift/install/metal/installer-provisioned/>]. You must use the pull secret later to complete the `install-config.yaml` file.
- Finally, download the RHCOS images for a PXE-based installation from the RHCOS mirror [<https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/>].

There is a private registry with a mirror of the OpenShift release images and other resources in the classroom environment. The classroom environment has access to the internet.

**Note**

After installing the `oc` and `openshift-install` binaries, it is useful to set up (and source) the bash commands completion:

```
[root@demo ~]# oc completion bash > /etc/bash_completion.d/openshift
[root@demo ~]# openshift-install completion bash \
> /etc/bash_completion.d/openshift-install
[root@demo ~]# source /etc/bash_completion.d/openshift
[root@demo ~]# source /etc/bash_completion.d/openshift-install
```

Analyzing the Necessary Services for the Installation

As discussed elsewhere, the utility server runs the necessary services for the installation in the classroom environment. In the classroom environment, all the services are already up and running but you will need to configure some of them to complete the installation successfully. You can explore the environment and its current configurations using the corresponding `systemd` service units.

- `dhcpd` DHCP server

The following example file shows a possible DHCP configuration.

```
[root@utility ~]# cat /etc/dhcp/dhcpd.conf
ddns-update-style interim;
ignore client-updates;
authoritative;
allow booting;
allow bootp;
allow unknown-clients;
# Set default and max IP lease time to infinite with -1 value
default-lease-time -1;
max-lease-time -1;

subnet 192.168.50.0 netmask 255.255.255.0 {
    option routers 192.168.50.254;
    option domain-name-servers 192.168.50.254;
    option ntp-servers 103.16.182.23,103.16.182.214;
    option domain-search "ocp4.example.com","example.com";
    filename "pxelinux.0";
    next-server 192.168.50.254;
        host master01.ocp4.example.com { hardware ethernet
52:54:00:00:32:0A; fixed-address 192.168.50.10; option host-name "master01"; }
        host master02.ocp4.example.com { hardware ethernet
52:54:00:00:32:0B; fixed-address 192.168.50.11; option host-name "master02"; }
        host master03.ocp4.example.com { hardware ethernet
52:54:00:00:32:0C; fixed-address 192.168.50.12; option host-name "master03"; }
        host bootstrap.ocp4.example.com { hardware ethernet
52:54:00:00:32:09; fixed-address 192.168.50.9; option host-name "bootstrap"; }
```

```

        host worker01.ocp4.example.com { hardware ethernet
52:54:00:00:32:0D; fixed-address 192.168.50.13; option host-name "worker01"; }
        host worker02.ocp4.example.com { hardware ethernet
52:54:00:00:32:0E; fixed-address 192.168.50.14; option host-name "worker02"; }
    }

```



Note

For more information, refer to the chapter *Providing DHCP services* of the *Red Hat Enterprise Linux 8 Configuring and Managing Networking Guide* at https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html-single/configuring_and_managing_networking/index#providing-dhcp-services_configuring-and-managing-networking

- named DNS server

The following example file shows a possible DNS configuration.

```
[root@utility ~]# cat /etc/named.conf

options {
#listen-on port 53 { 127.0.0.1; };
#listen-on-v6 port 53 { ::1; };
directory "/var/named";
dump-file "/var/named/data/cache_dump.db";
statistics-file "/var/named/data/named_stats.txt";
memstatistics-file "/var/named/data/named_mem_stats.txt";
secroots-file "/var/named/data/named.secroots";
recursing-file "/var/named/data/named.recurse";
allow-query { localhost; 192.168.50.0/24; 172.25.250.254; };
recursion yes;
dnssec-enable yes;
dnssec-validation yes;
managed-keys-directory "/var/named/dynamic";
pid-file "/run/named/named.pid";
session-keyfile "/run/named/session.key";

forwarders {
    172.25.250.254;
};

/* https://fedoraproject.org/wiki/Changes/CryptoPolicy */
include "/etc/crypto-policies/back-ends/bind.config";
};

logging {
    channel default_debug {
        file "data/named.run";
        severity dynamic;
    };
};

zone "." IN {
    type hint;
}
```

```
file "named.ca";
};

include "/etc/named.rfc1912.zones";
include "/etc/named.root.key";

zone "example.com" {
    type master;
    file "example.com.db";
    allow-update { none; };
};

zone "50.168.192.in-addr.arpa" IN {
    type master;
    file "example.com.reverse.db";
    allow-update { none; };
};
```

This configuration file makes reference to the files `example.com.db` (forward resolution) and `example.com.reverse.db` (reverse resolution) placed in the directory `/var/named`:

```
[root@utility ~]# cat /var/named/example.com.db
$TTL 1D
@      IN  SOA dns.ocp4.example.com. root.example.com. (
            2019022400 ; serial
            3h        ; refresh
            15        ; retry
            1w        ; expire
            3h        ; minimum
)
IN  NS  dns.ocp4.example.com.
dns.ocp4      IN A 192.168.50.254
api.ocp4      IN A 192.168.50.254
api-int.ocp4  IN A 192.168.50.254
*.apps.ocp4   IN A 192.168.50.254
bootstrap.ocp4 IN A 192.168.50.9
master01.ocp4 IN A 192.168.50.10
etcd-0.ocp4   IN A 192.168.50.10
_etcfd-server-ssl._tcp.ocp4  IN SRV 0 10 2380 etcd-0.ocp4
master02.ocp4 IN A 192.168.50.11
etcd-1.ocp4   IN A 192.168.50.11
_etcfd-server-ssl._tcp.ocp4  IN SRV 0 10 2380 etcd-1.ocp4
master03.ocp4 IN A 192.168.50.12
etcd-2.ocp4   IN A 192.168.50.12
_etcfd-server-ssl._tcp.ocp4  IN SRV 0 10 2380 etcd-2.ocp4
worker01.ocp4 IN A 192.168.50.13
worker02.ocp4 IN A 192.168.50.14
```

```
[root@utility ~]# cat /var/named/example.com.reverse.db
$TTL 1D
@      IN  SOA dns.ocp4.example.com. root.example.com. (
            2019022400 ; serial
            3h        ; refresh
```

```

        15      ; retry
        1w      ; expire
        3h      ; minimum
    )
IN NS dns.ocp4.example.com.
254 IN PTR api.ocp4.example.com.
254 IN PTR api-int.ocp4.example.com.
9 IN PTR bootstrap.ocp4.example.com.
10 IN PTR master01.ocp4.example.com.
11 IN PTR master02.ocp4.example.com.
12 IN PTR master03.ocp4.example.com.
13 IN PTR worker01.ocp4.example.com.
14 IN PTR worker02.ocp4.example.com.

```



Note

For more information, refer to the *BIND* section in the *Red Hat Enterprise Linux 7 Networking Guide* at https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html-single/networking_guide/index#sec-BIND

- tftp server for PXE booting

The PXE boot files must be placed in the directory `/var/lib/tftpboot/pxelinux.cfg/`. To use different PXE boot files for every single MAC address, name the files `01-aa-bb-cc-dd-ee-ff`, where `aa-bb-cc-dd-ee-ff` is the MAC address in lower case, using hyphens instead of colons.

An example of PXE boot file for a the MAC address `52:54:00:00:32:09` is:

```
[root@utility ~]# cat /var/lib/tftpboot/pxelinux.cfg/01-52-54-00-00-32-09
default menu.c32
prompt 0
timeout 0
menu title **** OpenShift 4 BOOTSTRAP PXE Boot Menu ****

label Install RHCOS 4.6.1 Bootstrap Node
kernel http://192.168.50.254:8080/openshift4/images/rhcos-4.6.1-x86_64-live-
kernel-x86_64
append ip=dhcp rd.neednet=1 coreos.inst.install_dev=vda console=tty0
console=ttyS0 coreos.inst=yes coreos.live.rootfs_url=http://192.168.50.254:8080/
openshift4/images/rhcos-4.6.1-x86_64-live-rootfs.x86_64.img
coreos.inst.ignition_url=http://192.168.50.254:8080/openshift4/4.6.4/ignitions/
bootstrap.ign initrd=http://192.168.50.254:8080/openshift4/images/rhcos-4.6.1-
x86_64-live-initramfs.x86_64.img
```



Note

For more information, refer to the chapter *Preparing to install from the network using PXE* of the *Red Hat Enterprise Linux 8 Performing an Advanced RHEL Installation Guide* at https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html-single/installation_guide/index#chap-installation-server-setup

- haproxy load balancer for API and application services

The following example file shows a possible HAProxy configuration:

```
[root@utility ~]# cat /etc/haproxy/haproxy.cfg

#-----
# Global settings
#-----
global
    log          127.0.0.1 local2
    chroot       /var/lib/haproxy
    pidfile      /var/run/haproxy.pid
    maxconn     4000
    user         haproxy
    group        haproxy
    daemon
    # turn on stats unix socket
    stats socket /var/lib/haproxy/stats
    # utilize system-wide crypto-policies
    ssl-default-bind-ciphers PROFILE=SYSTEM
    ssl-default-server-ciphers PROFILE=SYSTEM

#-----
# common defaults that all the 'listen' and 'backend' sections will
# use if not designated in their block
#-----
defaults
    mode          http
    log           global
    option         httplog
    option         dontlognull
    option http-server-close
    option forwardfor   except 127.0.0.0/8
    option         redispatch
    retries        3
    timeout http-request 10s
    timeout queue   1m
    timeout connect 10s
    timeout client   1m
    timeout server   1m
    timeout http-keep-alive 10s
    timeout check    10s
    maxconn       3000

#-----
# round robin balancing for RHOC P Kubernetes API Server
#-----
frontend k8s_api
    bind *:6443
    mode tcp
    default_backend k8s_api_backend
backend k8s_api_backend
    balance roundrobin
    mode tcp
```

```
server bootstrap 192.168.50.9:6443 check
server master01 192.168.50.10:6443 check
server master02 192.168.50.11:6443 check
server master03 192.168.50.12:6443 check

# -----
# round robin balancing for RHOCOP Machine Config Server
# -----
frontend machine_config
  bind *:22623
  mode tcp
  default_backend machine_config_backend
backend machine_config_backend
  balance roundrobin
  mode tcp
  server bootstrap 192.168.50.9:22623 check
  server master01 192.168.50.10:22623 check
  server master02 192.168.50.11:22623 check
  server master03 192.168.50.12:22623 check

# -----
# round robin balancing for RHOCOP Ingress Insecure Port
# -----
frontend ingress_insecure
  bind *:80
  mode tcp
  default_backend ingress_insecure_backend
backend ingress_insecure_backend
  balance roundrobin
  mode tcp
  server worker01 192.168.50.13:80 check
  server worker02 192.168.50.14:80 check

# -----
# round robin balancing for RHOCOP Ingress Secure Port
# -----
frontend ingress_secure
  bind *:443
  mode tcp
  default_backend ingress_secure_backend
backend ingress_secure_backend
  balance roundrobin
  mode tcp
  server worker01 192.168.50.13:443 check
  server worker02 192.168.50.14:443 check

# -----
# Exposing HAProxy Statistic Page
# -----
listen stats
  bind :32700
  stats enable
```

```
stats uri /
stats hide-version
stats auth admin:RedH@t322
```

**Note**

For more information, refer to the *HAProxy 2.4 Configuration Manual* at <https://www.haproxy.org/download/2.4/doc/configuration.txt>

- **httpd** for serving files

By default, the Apache HTTPD server is configured for publishing the `/var/www/html/` folder. You can modify this configuration in the file `/etc/httpd/conf/httpd.conf`.

In the classroom environment, the default HTTP port for the Apache HTTPD server has been set to the port 8080. This is because the HAProxy server is using port 80 for insecure HTTP connections to the ingress router pods.

**Note**

For more information, refer to the chapter *Setting up the Apache HTTP web server* of the *Red Hat Enterprise Linux 8 Deploying Different Types of Servers Guide* at https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html-single/deploying_different_types_of_servers/index#setting-apache-http-server_Deploying-different-types-of-servers

**Note**

The chrony server is enabled by default for any RHCOS installation. This way, the cluster nodes are configured to use the same upstream NTP servers. You can configure the NTP servers using a machine configuration file.

For more information, refer to the *Installing and configuring OpenShift Container Platform clusters* guide of the Red Hat OpenShift Container Platform 4.6 documentation at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing/index#installation-special-config-chrony_installing-customizing

Explaining the Installation Configuration File and Creating the Ignition Files

The `install-config.yaml` is the configuration file for the installation. The `openshift-install` command has an option to explain the fields contained in the `install-config.yaml` file:

```
[user@demo ~]$ openshift-install explain installconfig

KIND:     InstallConfig
VERSION:  v1

RESOURCE: <object>
InstallConfig is the configuration for an OpenShift install.
```

FIELDS:

```
additionalTrustBundle <string>
AdditionalTrustBundle is a PEM-encoded X.509 certificate bundle that is added to the nodes' trusted certificate store.

apiVersion <string>
APIVersion defines the versioned schema of this representation of an object. Servers should convert recognized schemas to the latest internal value, and may reject unrecognized values. More info: https://git.k8s.io/community/contributors/devel/sig-architecture/api-conventions.md#resources

baseDomain <string> -required-
BaseDomain is the base domain to which the cluster should belong.

...output omitted...
```

You can see more details appending the field:

```
[user@demo ~]$ openshift-install explain installconfig.baseDomain
KIND:     InstallConfig
VERSION:  v1

RESOURCE: <string>
BaseDomain is the base domain to which the cluster should belong.
```

You can find an example of the `install-config.yaml` file in the *Installing on bare metal* chapter of the Red Hat OpenShift Container Platform 4.6 documentation [https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_bare_metal/index#installation-initializing-manual_installing-bare-metal].

After creating the `install-config.yaml` file, you must create the Kubernetes manifests, customize them if necessary, and finally create the ignition files.



Warning

The `install-config.yaml` file is removed after processing to generate the Kubernetes manifests. Even if it can be created again, Red Hat recommends making a copy of the `install-config.yaml` file before it is processed by the `openshift-install` command. If the installation fails, the `install-config.yaml` file can be reviewed for debugging possible issues.

Finally, you must make the ignition files available for the nodes during the installation and configure the `tftpboot` files to retrieve them during the RHCOS installation.

Generating an SSH Key for Troubleshooting the Installer

To troubleshoot the installation process in the `bootstrap` server, you can create an SSH key. After generating the SSH key pair, add the public SSH key into the `install-config.yaml`. The installer will include the public key in the `bootstrap` node for connecting via SSH to troubleshoot the first stages of the installation.

This approach is useful for troubleshooting issues while pulling the images, the installation of RHCOS in the `bootstrap` or `master` nodes, or connectivity issues to the `master` nodes or the APIserver.



References

- For more information, refer to the *Installing on bare metal* chapter of the Red Hat OpenShift Container Platform 4.6 documentation at
https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/installing_on_bare_metal/index#installing-bare-metal

► Guided Exercise

Configuring Network Services and Hosts for Installing OpenShift Without an Infrastructure Provider

- Check and configure services and prepare the environment for installing OpenShift on pre-existing infrastructure.

Outcomes

You should be able to:

- Download the RHCOS files.
- Create the ignition files from the `install-config.yaml` file.
- Check and configure the necessary services.

Before You Begin

To perform this exercise, ensure that you have read the *Planning to Install OpenShift Without an Infrastructure Provider* lecture.

Also, ensure you have completed the *Completing the OpenShift Installation Prerequisites* guided exercise.

If you have made any changes to the classroom environment, you must destroy the classroom and recreate to complete this activity successfully.

Instructions

- 1. From the utility server as the root user, check and configure the DNS service.
- 1.1. Check and configure (if necessary) the DNS service provided by the named daemon. The configuration file is `/etc/named.conf`, as shown in the output of the `systemctl status` unit status:

```
[student@workstation ~]$ ssh lab@utility
...output omitted...
[lab@utility ~]$ sudo -i
[root@utility ~]#
```

```
[root@utility ~]# systemctl status named
● named.service - Berkeley Internet Name Domain (DNS)
   Loaded: loaded (/usr/lib/systemd/system/named.service; enabled; vendor preset:
dis>
   Active: active (running) since Thu 2020-11-19 06:13:07 EST; 3h 32min ago
     Main PID: 944 (named)
        Tasks: 5 (limit: 11345)
```

```
Memory: 60.7M
CGroup: /system.slice/named.service
└─944 /usr/sbin/named -u named -c /etc/named.conf
```

12. Open the /etc/named.conf file to see its content. This file does not require any changes.

```
[root@utility ~]# cat /etc/named.conf
...output omitted...
```

13. Inspect the database files for the domain example.com in /var/named:

```
[root@utility ~]# cat /var/named/example.com.db
$TTL 1D
@ IN SOA dns.ocp4.example.com. root.example.com. (
    2019022400 ; serial
    3h         ; refresh
    15         ; retry
    1w         ; expire
    3h         ; minimum
)
IN NS dns.ocp4.example.com.
dns.ocp4    IN A 192.168.50.254
api.ocp4     IN A 192.168.50.254
api-int.ocp4 IN A 192.168.50.254
*.apps.ocp4  IN A 192.168.50.254
bootstrap.ocp4 IN A 192.168.50.9
master01.ocp4 IN A 192.168.50.10
etcd-0.ocp4  IN A 192.168.50.10
_etcd-server-ssl._tcp.ocp4 IN SRV 0 10 2380 etcd-0.ocp4
master02.ocp4 IN A 192.168.50.11
etcd-1.ocp4  IN A 192.168.50.11
_etcd-server-ssl._tcp.ocp4 IN SRV 0 10 2380 etcd-1.ocp4
master03.ocp4 IN A 192.168.50.12
etcd-2.ocp4  IN A 192.168.50.12
_etcd-server-ssl._tcp.ocp4 IN SRV 0 10 2380 etcd-2.ocp4
worker01.ocp4 IN A 192.168.50.13
worker02.ocp4 IN A 192.168.50.14
```

```
[root@utility ~]# cat /var/named/example.com.reverse.db
$TTL 1D
@ IN SOA dns.ocp4.example.com. root.example.com. (
    2019022400 ; serial
    3h         ; refresh
    15         ; retry
    1w         ; expire
    3h         ; minimum
)
IN NS dns.ocp4.example.com.
254 IN PTR api.ocp4.example.com.
254 IN PTR api-int.ocp4.example.com.
9 IN PTR bootstrap.ocp4.example.com.
10 IN PTR master01.ocp4.example.com.
```

```
11 IN PTR master02.ocp4.example.com.  
12 IN PTR master03.ocp4.example.com.  
13 IN PTR worker01.ocp4.example.com.  
14 IN PTR worker02.ocp4.example.com.
```

1.4. Use the dig command to test the DNS resolution:

```
[root@utility ~]# dig api.ocp4.example.com  
...output omitted...  
;api.ocp4.example.com. IN A  
  
;; ANSWER SECTION:  
api.ocp4.example.com. 86400 IN A 192.168.50.254  
  
;; AUTHORITY SECTION:  
example.com. 86400 IN NS dns.ocp4.example.com.  
  
;; ADDITIONAL SECTION:  
dns.ocp4.example.com. 86400 IN A 192.168.50.254  
  
;; Query time: 2 msec  
;; SERVER: 172.25.250.254#53(172.25.250.254)  
;; WHEN: Wed Jan 27 12:07:49 EST 2021  
;; MSG SIZE rcvd: 127  
  
[root@utility ~]# dig test.apps.ocp4.example.com  
...output omitted...  
;test.apps.ocp4.example.com. IN A  
  
;; ANSWER SECTION:  
test.apps.ocp4.example.com. 86400 IN A 192.168.50.254  
  
;; AUTHORITY SECTION:  
example.com. 86400 IN NS dns.ocp4.example.com.  
  
;; ADDITIONAL SECTION:  
dns.ocp4.example.com. 86400 IN A 192.168.50.254  
  
;; Query time: 2 msec  
;; SERVER: 172.25.250.254#53(172.25.250.254)  
;; WHEN: Wed Jan 27 12:08:41 EST 2021  
;; MSG SIZE rcvd: 133  
  
[root@utility ~]# dig master01.ocp4.example.com  
...output omitted...  
;master01.ocp4.example.com. IN A  
  
;; ANSWER SECTION:  
master01.ocp4.example.com. 86400 IN A 192.168.50.10  
  
;; AUTHORITY SECTION:  
example.com. 86400 IN NS dns.ocp4.example.com.  
  
;; ADDITIONAL SECTION:
```

```
dns.ocp4.example.com. 86400 IN A 192.168.50.254

;; Query time: 1 msec
;; SERVER: 172.25.250.254#53(172.25.250.254)
;; WHEN: Wed Jan 27 12:09:19 EST 2021
;; MSG SIZE  rcvd: 132
```

► 2. Check and configure the DHCP service.

- 2.1. Check and configure (if necessary) the DHCP service provided by the `dhcpd` daemon. The configuration file is `/etc/dhcp/dhcpd.conf`, as shown in the output of the `systemctl status` command:

```
[root@utility ~]# systemctl status dhcpcd
● dhcpcd.service - DHCPv4 Server Daemon
    Loaded: loaded (/usr/lib/systemd/system/dhcpcd.service; enabled; vendor preset: disabled)
      Active: active (running) since Thu 2020-11-19 06:13:08 EST; 3h 39min ago
        Docs: man:dhcpcd(8)
               man:dhcpcd.conf(5)
    Main PID: 1261 (dhcpcd)
      Status: "Dispatching packets..."
       Tasks: 1 (limit: 11345)
     Memory: 11.9M
      CGroup: /system.slice/dhcpcd.service
              └─1261 /usr/sbin/dhcpcd -f -cf /etc/dhcp/dhcpcd.conf -user dhcpcd -group
                dhcpcd --no-pid
```

- 2.2. Open the `/etc/dhcp/dhcpd.conf` file to see its content. This file contains the MAC addresses of the servers in the classroom environment. Later, you will use these MAC addresses to create the PXE boot configuration files. Do not make any changes to this file.

```
[root@utility ~]# cat /etc/dhcp/dhcpcd.conf
...output omitted...
host master01.ocp4.example.com { hardware ethernet 52:54:00:00:32:0A; fixed-
address 192.168.50.10; option host-name "master01"; }
host master02.ocp4.example.com { hardware ethernet 52:54:00:00:32:0B; fixed-
address 192.168.50.11; option host-name "master02"; }
host master03.ocp4.example.com { hardware ethernet 52:54:00:00:32:0C; fixed-
address 192.168.50.12; option host-name "master03"; }
host bootstrap.ocp4.example.com { hardware ethernet 52:54:00:00:32:09; fixed-
address 192.168.50.9; option host-name "bootstrap"; }
host worker01.ocp4.example.com { hardware ethernet 52:54:00:00:32:0D; fixed-
address 192.168.50.13; option host-name "worker01"; }
host worker02.ocp4.example.com { hardware ethernet 52:54:00:00:32:0E; fixed-
address 192.168.50.14; option host-name "worker02"; }

...output omitted...
```

► 3. Check and configure the HTTPD service.

- 3.1. Check and configure (if necessary) the HTTP service provided by the `httpd` daemon.

```
[root@utility ~]# systemctl status httpd
● httpd.service - The Apache HTTP Server
  Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
  Active: active (running) since Thu 2020-11-19 06:13:07 EST; 3h 44min ago
    Docs: man:httpd.service(8)
 Main PID: 957 (httpd)
   Status: "Running, listening on: port 8080"
     Tasks: 213 (limit: 11345)
    Memory: 39.4M
      CGroup: /system.slice/httpd.service
              ├─957 /usr/sbin/httpd -DFOREGROUND
              ├─966 /usr/sbin/httpd -DFOREGROUND
              ├─970 /usr/sbin/httpd -DFOREGROUND
              ├─971 /usr/sbin/httpd -DFOREGROUND
              └─974 /usr/sbin/httpd -DFOREGROUND
```

- 3.2. The default configuration file for the `httpd` server is `/etc/httpd/conf/httpd.conf`. Because you will use it as a file server, you must find the directory in which to place the files. To find this directory, run the following command.

```
[root@utility ~]# grep DocumentRoot /etc/httpd/conf/httpd.conf
# DocumentRoot: The directory out of which you will serve your
DocumentRoot "/var/www/html"
...output omitted...
```

- 3.3. Create the `/var/www/html/openshift4/images` folder.

```
[root@utility ~]# mkdir -p /var/www/html/openshift4/images
```

- 3.4. Create the `/var/www/html/openshift4/4.6.4/ignitions` folder.

```
[root@utility ~]# mkdir -p /var/www/html/openshift4/4.6.4/ignitions
```

- 3.5. Restore the SELinux security context of the `/var/www/html/openshift4/images` folder.

```
[root@utility ~]# restorecon -Rv /var/www/html/openshift4
```

- 3.6. Review the structure of the `/var/www/html` folder:

```
[root@utility ~]# tree /var/www/html/
/var/www/html/
└── openshift4
    ├── 4.6.4
    │   └── ignitions
    └── images
```

The httpd server is prepared to place the ignition files in /var/www/html/openshift4/4.6.4/ignitions/, and the RHCOS images in /var/www/html/openshift4/images/.

- 3.7. Download the RHCOS images to the /var/www/html/openshift4/images/ folder:

```
[root@utility ~]# ocp_maj=4.6;rhcov_ver=4.6.1
[root@utility ~]# mirror=https://mirror.openshift.com/pub/openshift-v4
[root@utility ~]# baseurl=${mirror}/dependencies/rhcos/${ocp_maj}/${rhcov_ver}
[root@utility ~]# cd /var/www/html/openshift4/images/
[root@utility images]# wget \
> ${baseurl}/rhcos-${rhcov_ver}-x86_64-live-rootfs.x86_64.img
...output omitted...
```

```
2020-11-23 10:14:02 (57,0 MB/s) - "rhcos-4.6.1-x86_64-live-rootfs.x86_64.img"
 saved [819969024/819969024]
```

```
[root@utility images]# wget \
> ${baseurl}/rhcos-${rhcov_ver}-x86_64-live-kernel-x86_64
...output omitted...
```

```
2020-11-23 10:15:50 (6,26 MB/s) - "rhcos-4.6.1-x86_64-live-kernel-x86_64" saved
[8924528/8924528]
```

```
[root@utility images]# wget \
> ${baseurl}/rhcos-${rhcov_ver}-x86_64-live-initramfs.x86_64.img
...output omitted...
```

```
2020-11-23 10:16:33 (29,2 MB/s) - "rhcos-4.6.1-x86_64-live-initramfs.x86_64.img"
 saved [80239428/80239428]
```

- 3.8. Return to the root home folder:

```
[root@utility images]# cd ~
[root@utility ~]#
```

▶ 4. Check and Configure the Load Balancer.

- 4.1. Check and configure (if necessary) the HAProxy load balancer service provided by the haproxy daemon. The configuration file is /etc/haproxy/haproxy.cfg, as shown in the output of the systemctl unit status:

```
[root@utility ~]$ systemctl status haproxy
● haproxy.service - HAProxy Load Balancer
   Loaded: loaded (/usr/lib/systemd/system/haproxy.service; enabled; vendor
   preset: disabled)
     Active: active (running) since Fri 2020-11-20 10:20:07 EST; 26min ago
       Process: 917 ExecStartPre=/usr/sbin/haproxy -f $CONFIG -c -q (code=exited,
      Status: 0/SUCCESS)
     Main PID: 932 (haproxy)
        Tasks: 2 (limit: 11345)
```

```
Memory: 4.7M
CGroup: /system.slice/haproxy.service
    └─932 /usr/sbin/haproxy -Ws -f /etc/haproxy/haproxy.cfg -p /run/
      haproxy.pid
          └─943 /usr/sbin/haproxy -Ws -f /etc/haproxy/haproxy.cfg -p /run/
            haproxy.pid
```

- 4.2. Edit the /etc/haproxy/haproxy.cfg file to add the back-end servers for the API Server, Machine Config Server, and applications (secure and insecure).

```
[root@utility ~]# vi /etc/haproxy/haproxy.cfg

...output omitted...

# -----
# round robin balancing for RHOC P Kubernetes API Server
# -----
frontend k8s_api
  bind *:6443
  mode tcp
  default_backend k8s_api_backend
backend k8s_api_backend
  balance roundrobin
  mode tcp
  server bootstrap 192.168.50.9:6443 check
  server master01 192.168.50.10:6443 check
  server master02 192.168.50.11:6443 check
  server master03 192.168.50.12:6443 check

# -----
# round robin balancing for RHOC P Machine Config Server
# -----
frontend machine_config
  bind *:22623
  mode tcp
  default_backend machine_config_backend
backend machine_config_backend
  balance roundrobin
  mode tcp
  server bootstrap 192.168.50.9:22623 check
  server master01 192.168.50.10:22623 check
  server master02 192.168.50.11:22623 check
  server master03 192.168.50.12:22623 check

# -----
# round robin balancing for RHOC P Ingress Insecure Port
# -----
frontend ingress_insecure
  bind *:80
  mode tcp
  default_backend ingress_insecure_backend
backend ingress_insecure_backend
  balance roundrobin
  mode tcp
  server worker01 192.168.50.13:80 check
```

```
server worker02 192.168.50.14:80 check

# -----
# round robin balancing for RHOPC Ingress Secure Port
# -----
frontend ingress_secure
  bind *:443
  mode tcp
  default_backend ingress_secure_backend
backend ingress_secure_backend
  balance roundrobin
  mode tcp
  server worker01 192.168.50.13:443 check
  server worker02 192.168.50.14:443 check

# -----
# Exposing HAProxy Statistic Page
# -----
listen stats
  bind :32700
  stats enable
  stats uri /
  stats hide-version
  stats auth admin:RedH@t322
```

Save the changes to the /etc/haproxy/haproxy.cfg file.



Note

You can compare the content of your /etc/haproxy/haproxy.cfg file to the solution file in <http://classroom.example.com/materials/solutions/haproxy/haproxy.cfg>

Use the wget command to download this file into your working directory:

```
[root@utility ~]# wget http://classroom.example.com/materials/solutions/
haproxy/haproxy.cfg
...output omitted...
```

- 4.3. Verify that the haproxy.cfg file does not contain any errors. Reload the HAProxy service to apply the changes. Verify the service status.

```
[root@utility ~]# haproxy -c -f /etc/haproxy/haproxy.cfg
[WARNING] 026/114452 (1646) : parsing [/etc/haproxy/haproxy.cfg:49] : 'option
httplog' not usable with frontend 'k8s_api' (needs 'mode http'). Falling back to
'option tcplog'.
[WARNING] 026/114452 (1646) : config : 'option forwardfor' ignored for frontend
'k8s_api' as it requires HTTP mode.
[WARNING] 026/114452 (1646) : config : 'option forwardfor' ignored for backend
'k8s_api_backend' as it requires HTTP mode.
[WARNING] 026/114452 (1646) : parsing [/etc/haproxy/haproxy.cfg:49] : 'option
httplog' not usable with frontend 'machine_config' (needs 'mode http'). Falling
back to 'option tcplog'.
```

```
[WARNING] 026/114452 (1646) : config : 'option forwardfor' ignored for frontend  
'machine_config' as it requires HTTP mode.  
[WARNING] 026/114452 (1646) : config : 'option forwardfor' ignored for backend  
'machine_config_backend' as it requires HTTP mode.  
[WARNING] 026/114452 (1646) : parsing [/etc/haproxy/haproxy.cfg:49] : 'option  
httplog' not usable with frontend 'ingress_insecure' (needs 'mode http'). Falling  
back to 'option tcplog'.  
[WARNING] 026/114452 (1646) : config : 'option forwardfor' ignored for frontend  
'ingress_insecure' as it requires HTTP mode.  
[WARNING] 026/114452 (1646) : config : 'option forwardfor' ignored for backend  
'ingress_insecure_backend' as it requires HTTP mode.  
[WARNING] 026/114452 (1646) : parsing [/etc/haproxy/haproxy.cfg:49] : 'option  
httplog' not usable with frontend 'ingress_secure' (needs 'mode http'). Falling  
back to 'option tcplog'.  
[WARNING] 026/114452 (1646) : config : 'option forwardfor' ignored for frontend  
'ingress_secure' as it requires HTTP mode.  
[WARNING] 026/114452 (1646) : config : 'option forwardfor' ignored for backend  
'ingress_secure_backend' as it requires HTTP mode.  
Configuration file is valid
```



Note

The command above returns warning messages because some HTTP default options cannot be applied to TCP connections. You can safely ignore those warnings.

```
[root@utility ~]# systemctl reload haproxy
```

```
[root@utility ~]$ systemctl status haproxy  
● haproxy.service - HAProxy Load Balancer  
    Loaded: loaded (/usr/lib/systemd/system/haproxy.service; enabled; vendor  
    preset: disabled)  
      Active: active (running) since Fri 2020-11-20 10:20:07 EST; 26min ago  
        ...output omitted...
```

- 5. Check and configure the TFTP service. After this step, the `/var/lib/tftpboot/pxelinux.cfg/` folder structure should look like this:

```
[root@utility ~]# tree /var/lib/tftpboot/pxelinux.cfg/  
/var/lib/tftpboot/pxelinux.cfg/  
└── 01-52-54-00-00-32-09  
    └── 01-52-54-00-00-32-0a  
        └── 01-52-54-00-00-32-0b
```

```
└── 01-52-54-00-00-32-0c  
└── 01-52-54-00-00-32-0d  
└── 01-52-54-00-00-32-0e  
└── 01-example
```

**Note**

You can use `wget` to download the pxeboot files from the classroom server:

```
[root@utility ~]# sol_url=http://classroom.example.com/materials/solutions  
[root@utility ~]# wget $sol_url/pxelinux.cfg/01-52-54-00-00-32-09 \  
> -P /var/lib/tftpboot/pxelinux.cfg/  
...output omitted...
```

- 5.1. Check and configure (if necessary) the TFTP service provided by the `tftp` daemon.

```
[root@utility ~]# systemctl status tftp  
● tftp.service - Tftp Server  
  Loaded: loaded (/usr/lib/systemd/system/tftp.service; indirect; vendor preset:  
disabled)  
  Active: active (running) since Mon 2020-11-23 09:37:46 EST; 21min ago  
    Docs: man:in.tftpd(8)  
 Main PID: 1331 (in.tftpd)  
   Tasks: 1 (limit: 11345)  
  Memory: 1.2M  
 CGroup: /system.slice/tftp.service  
         └─1331 /usr/sbin/in.tftpd -s /var/lib/tftpboot
```

- 5.2. Using the MAC addresses from the `/etc/dhcp/dhcpd.conf` file referenced in a previous step, create six files in `/var/lib/tftpboot/pxelinux.cfg` as follows.

Prepend `01-` to the MAC address in lowercase, separating each pair of hexadecimal digits with a hyphen.

- `master01 - /var/lib/tftpboot/pxelinux.cfg/01-52-54-00-00-32-0a`
- `master02 - /var/lib/tftpboot/pxelinux.cfg/01-52-54-00-00-32-0b`
- `master03 - /var/lib/tftpboot/pxelinux.cfg/01-52-54-00-00-32-0c`

Content of the three files:

```

default menu.c32
prompt 0
timeout 0
menu title **** OpenShift 4 MASTER PXE Boot Menu ****

label Install RHCOS 4.6.1 Master Node
kernel http://192.168.50.254:8080/openshift4/images/rhcos-4.6.1-x86_64-live-
kernel-x86_64
append ip=dhcp rd.neednet=1 coreos.inst.install_dev=vda console=tty0
console=ttyS0 coreos.inst=yes coreos.live.rootfs_url=http://192.168.50.254:8080/
openshift4/images/rhcos-4.6.1-x86_64-live-rootfs.x86_64.img
coreos.inst.ignition_url=http://192.168.50.254:8080/openshift4/4.6.4/
ignitions/master.ign initrd=http://192.168.50.254:8080/openshift4/images/
rhcos-4.6.1-x86_64-live-initramfs.x86_64.img

```



Note

Ensure that after `label Install RHCOS 4.6.1 Master Node` there are only two more lines in the file. The penultimate line starts with `kernel` and the last line of the file starts with `append`. There should only be one line break between `http://192.168.50.254:8080/openshift4/images/rhcos-4.6.1-x86_64-live-kernel-x86_64` and `append`.

- `bootstrap - /var/lib/tftpboot/pxelinux.cfg/01-52-54-00-00-32-09`

Content of the file:

```

default menu.c32
prompt 0
timeout 0
menu title **** OpenShift 4 BOOTSTRAP PXE Boot Menu ****

label Install RHCOS 4.6.1 Bootstrap Node
kernel http://192.168.50.254:8080/openshift4/images/rhcos-4.6.1-x86_64-live-
kernel-x86_64
append ip=dhcp rd.neednet=1 coreos.inst.install_dev=vda console=tty0
console=ttyS0 coreos.inst=yes coreos.live.rootfs_url=http://192.168.50.254:8080/
openshift4/images/rhcos-4.6.1-x86_64-live-rootfs.x86_64.img
coreos.inst.ignition_url=http://192.168.50.254:8080/openshift4/4.6.4/
ignitions/bootstrap.ign initrd=http://192.168.50.254:8080/openshift4/images/
rhcos-4.6.1-x86_64-live-initramfs.x86_64.img

```



Note

Ensure that after `label Install RHCOS 4.6.1 Bootstrap Node` there are only two more lines in the file. The penultimate line starts with `kernel` and the last line of the file starts with `append`. There should only be one line break between `http://192.168.50.254:8080/openshift4/images/rhcos-4.6.1-x86_64-live-kernel-x86_64` and `append`.

- worker01 - /var/lib/tftpboot/pixelinux.cfg/01-52-54-00-00-32-0d
- worker02 - /var/lib/tftpboot/pixelinux.cfg/01-52-54-00-00-32-0e

Content of the two files:

```
default menu.c32
prompt 0
timeout 0
menu title **** OpenShift 4 WORKER PXE Boot Menu ****

label Install RHCOS 4.6.1 Worker Node
kernel http://192.168.50.254:8080/openshift4/images/rhcos-4.6.1-x86_64-live-
kernel-x86_64
append ip=dhcp rd.neednet=1 coreos.inst.install_dev=vda console=tty0
console=ttyS0 coreos.inst=yes coreos.live.rootfs_url=http://192.168.50.254:8080/
openshift4/images/rhcos-4.6.1-x86_64-live-rootfs.x86_64.img
coreos.inst.ignition_url=http://192.168.50.254:8080/openshift4/4.6.4/
ignitions/worker.ign initrd=http://192.168.50.254:8080/openshift4/images/
rhcos-4.6.1-x86_64-live-initramfs.x86_64.img
```

**Note**

Ensure that after label `Install RHCOS 4.6.1 Worker Node` there are only two more lines in the file. The penultimate line starts with `kernel` and the last line of the file starts with `append`. There should only be one line break between `http://192.168.50.254:8080/openshift4/images/rhcos-4.6.1-x86_64-live-kernel-x86_64` and `append`.

5.3. Check the `/var/lib/tftpboot/pixelinux.cfg/` folder structure:

Now, the tftp server is ready for serving the PXE boot files per MAC address. The servers retrieve those files upon restart.

**Note**

You can compare the content of your files in `/var/lib/tftpboot/pixelinux.cfg/` with the solution files in <http://classroom.example.com/materials/solutions/pixelinux.cfg/>

**Warning**

Do not attempt to install RHCOS in the servers yet. You must create the ignition files first, or the installation will fail.

► 6. As the `lab` user, create the ignition files and place them on the file server.

**Note**

Use the `lab` user to complete the remaining steps in this guided exercise.

- 6.1. Create a folder called `ocp4upi` and change to that folder:

```
[root@utility ~]# exit  
logout  
[lab@utility ~]$ mkdir ocp4upi && cd ocp4upi  
[lab@utility ocp4upi]$
```

- 6.2. Create the `install-config.yaml` file.

To speed up the installation, use the local registry of the region where your cluster is deployed, containing the OpenShift release images mirrored from `quay.io`.



Note

By default, the lectures and guided exercises in this course use the local registry of the `northamerica` region `nexus-registry-int.apps.tools-na150.prod.ole.redhat.com`.

Ensure that you use the local registry FQDN of the region of your classroom environment in all the guided exercises. Using a local registry from other regions will cause the failure of the OpenShift installation.

You can find out the region where your cluster is deployed using Firefox to navigate to the **Lab Environment** tab on the Red Hat OpenShift Installation Lab course web page from the **workstation** machine.

Once there, click the **information** button to display the information of the classroom environment. The classroom environment region is shown in the **Published region** field.

The screenshot shows the 'Lab Environment' tab of the Red Hat OpenShift Installation Lab course. At the top, there are navigation links: 'Table of Contents', 'Course', 'Lab Environment', a star icon, and a question mark icon. Below the navigation, there's a section titled '▶ Lab Controls'. It contains a message about creating the lab environment and a warning about deleting it. A tooltip is overlaid on the 'Published region' field, which is highlighted with a red box. The tooltip displays the following information:

Project id:	2ebdcefb42a144ada7e8clc3fceb8b56
Project name:	ole-3030f6f7-9253-47ed-9ea1-73c0aa7e62c9
Project state:	stopped
Lab definition id:	do322ea-4.6
Published region:	northamerica
openstack region:	us-east-1

Below the tooltip, there are three buttons: 'DELETE' (red), 'START' (green), and an info icon. A table lists three virtual machines: 'bastion' (stopped), 'bootstrap' (stopped), and 'classroom' (stopped). Each row has an 'ACTION' dropdown and an 'OPEN CONSOLE' button. The 'classroom' row is highlighted with a green bar.

Previously in this course, you generated the `pull-secret-oneline.json` file using the local registry in your region and copied it to the **utility** server.

```
[lab@utility ocp4upi]$ cat ~/pull-secret-oneline.json
>{"auths":{"cloud.openshift.com":
>{"auth":"UxUUj...UUjYw==","email":"student@redhat.com"}, "nexus-
registry-int.apps.tools-na150.prod.ole.redhat.com":
>{"auth":"cmVnd...STTM=","email":"nobody@example.com"}, "registry.connect.redhat.com":
>{"auth":"CvmsW...qKLJN=","email":"student@redhat.com"}, "registry.redhat.io":
>{"auth":"CvmsW...qKLJN=","email":"student@redhat.com"}}}
```

You must include the pull secret in the `pullSecret` section of the `install-config.yaml` file.

Also, include the public SSH key located in `/home/lab/.ssh/ocp4upi.pub`

Finally, include the `imageContentSources` configuration. This configuration enables the use of the local mirror registry during the installation, and it depends on the region where your cluster is deployed.

The following table shows the FQDN of the local registry for each region.

Local Registry FQDN per Classroom Environment Region

Region	Local registry FQDN
northamerica	nexus-registry-int.apps.tools-na150.prod.ole.redhat.com
emea	nexus-registry-int.apps.tools-emea150.prod.ole.redhat.com
apac	nexus-registry-int.apps.tools-apac150.prod.ole.redhat.com

```
[lab@utility ocp4upi]$ vi install-config.yaml
```

```
apiVersion: v1
baseDomain: example.com
compute:
- hyperthreading: Enabled
  name: worker
  replicas: 2
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
metadata:
  name: ocp4
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  none: {}
```

```
fips: false
pullSecret: |
<CHANGE_ME_KEEPING_THE_INDENTATION_LEVEL>
sshKey: |
<CHANGE_ME_KEEPING_THE_INDENTATION_LEVEL>
imageContentSources:
- mirrors:
  - <LOCAL_REGISTRY_FQDN>/openshift/ocp4
    source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <LOCAL_REGISTRY_FQDN>/openshift/ocp4
    source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

Save the file and close the editor.



Note

You can download the base `install-config.yaml` file from <http://classroom.example.com/materials/solutions/ocp4upi/install-config.yaml>

You still must complete that file with the proper `pullSecret`, `sshKey`, and `imageContentSources` information.

You can use `wget` to download this file to your working directory:

```
[root@utility ~]# wget \
> http://classroom.example.com/materials/solutions/ocp4upi/install-config.yaml
...output omitted...
```

- 6.3. Override the OpenShift install release image to use the image stored in the local registry. You must use the local registry FQDN of the region where your cluster is deployed. For instance, for a cluster deployed in the `northamerica` region, you would use the FQDN `nexus-registry-int.apps.tools-na150.prod.ole.redhat.com`.

```
[lab@utility
ocp4upi]$ reg="nexus-registry-int.apps.tools-na150.prod.ole.redhat.com"
[lab@utility ocp4upi]$ releaseimg="/openshift/ocp4:4.6.4-x86_64"
[lab@utility ocp4upi]$ export \
> OPENSHIFT_INSTALL_RELEASE_IMAGE_OVERRIDE=$reg$releaseimg
```

Verify the complete URL path to the release image as follows:

```
[lab@utility ~]$ echo $OPENSHIFT_INSTALL_RELEASE_IMAGE_OVERRIDE
nexus-registry-int.apps.tools-na150.prod.ole.redhat.com/openshift/ocp4:4.6.4-
x86_64
```

- 6.4. Save a copy of the `install-config.yaml` file in the `home` folder of the `lab` user.

```
[lab@utility ocp4upi]$ cp install-config.yaml /home/lab/
```

- 6.5. Generate the installation manifests using the `openshift-install` tool. The `install-config.yaml` is processed into the manifest files.

```
[lab@utility ocp4upi]$ openshift-install create manifests --dir=.
INFO Consuming Install Config from target directory
INFO Manifests created in: manifests and openshift
```

- 6.6. Inspect the `manifests` and `openshift` folders structure. These folders contain `yaml` configuration files that you can use to create the RHCOS ignition files.

```
[lab@utility ocp4upi]$ tree .
.
└── manifests
    ├── 04-openshift-machine-config-operator.yaml
    ├── cluster-config.yaml
    ├── cluster-dns-02-config.yml
    ├── cluster-infrastructure-02-config.yml
    ├── cluster-ingress-02-config.yml
    ├── cluster-network-01-crd.yaml
    ├── cluster-network-02-config.yml
    ├── cluster-proxy-01-config.yaml
    ├── cluster-scheduler-02-config.yml
    ├── cvo-overrides.yaml
    ├── etcd-ca-bundle-configmap.yaml
    ├── etcd-client-secret.yaml
    ├── etcd-metric-client-secret.yaml
    ├── etcd-metric-serving-ca-configmap.yaml
    ├── etcd-metric-signer-secret.yaml
    ├── etcd-namespace.yaml
    ├── etcd-service.yaml
    ├── etcd-serving-ca-configmap.yaml
    ├── etcd-signer-secret.yaml
    ├── kube-cloud-config.yaml
    ├── kube-system-configmap-root-ca.yaml
    ├── machine-config-server-tls-secret.yaml
    └── openshift-config-secret-pull-secret.yaml
.
└── openshift
    ├── 99_kubeadmin-password-secret.yaml
    ├── 99_openshift-cluster-api_master-user-data-secret.yaml
    ├── 99_openshift-cluster-api_worker-user-data-secret.yaml
    ├── 99_openshift-machineconfig_99-master-ssh.yaml
    ├── 99_openshift-machineconfig_99-worker-ssh.yaml
    └── openshift-install-manifests.yaml
```

- 6.7. Create the ignition files.

```
[lab@utility ocp4upi]$ openshift-install create ignition-configs --dir=.
WARNING Found override for release image. Please be warned, this is not advised
INFO Consuming Worker Machines from target directory
INFO Consuming Openshift Manifests from target directory
INFO Consuming Master Machines from target directory
INFO Consuming OpenShift Install (Manifests) from target directory
INFO Consuming Common Manifests from target directory
INFO Ignition-Configs created in: . and auth
```

6.8. Inspect the new folder structure.

```
[lab@utility ocp4upi]$ tree .
.
└── auth
    ├── kubeadmin-password
    └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

6.9. Copy the ignition files to the `httpd` server DocumentRoot and make them readable by any user.

```
[lab@utility ocp4upi]$ sudo cp *.ign /var/www/html/openshift4/4.6.4/ignitions/
[lab@utility ocp4upi]$ sudo chmod +r \
> /var/www/html/openshift4/4.6.4/ignitions/*.ign
```

► 7. Return to the `/home/lab/` folder.

```
[lab@utility ocp4upi]$ cd ~
[lab@utility ~]$
```

Finish

Do not make any other changes to the lab environment until the next guided exercise. You will continue using it in later guided exercises.

This concludes the guided exercise.

► Quiz

Chapter Review: Planning to Install OpenShift Without an Infrastructure Provider

Choose the correct answers to the following questions:

- ▶ 1. Which two of the following services are NOT required for installing OpenShift on pre-existing infrastructure? (Choose two.)
 - a. A PXE server
 - b. A DNS server
 - c. A DHCP server
 - d. A Load Balancer for the APIserver

- ▶ 2. Which of the following sentences about connecting to the bootstrap node is correct?
 - a. The Administrator can connect to the bootstrap node via SSH after the installation is completed using the `core` user.
 - b. The Administrator can add a public SSH key to the `install-config.yaml` file for connecting to the bootstrap node and for debugging the installation process.
 - c. The Administrator can create users in the bootstrap node using the `install-config.yaml` file.
 - d. The Administrator must add the SSH public key to the `.ssh/authorized_keys` file on the bootstrap server after the installation has begun.

- ▶ 3. Which of the following sentences about the ignition files is true?
 - a. The ignition files are injected into the different hosts using the `install-config.yaml` file.
 - b. Red Hat recommends creating the ignition files manually from the examples found in the official documentation to avoid mistakes.
 - c. The ignition files are created from the `install-config.yaml` file using the `openshift-install` command.
 - d. The ignition files are created from the `install-config.yaml` file using the `oc` command.

- **4. Which two of the following resources are necessary to complete the installation of RHCOS? (Choose two.)**
- a. The ignition files
 - b. The kickstart files
 - c. The oc CLI tool
 - d. The RHCOS images

► Solution

Chapter Review: Planning to Install OpenShift Without an Infrastructure Provider

Choose the correct answers to the following questions:

- ▶ 1. Which two of the following services are NOT required for installing OpenShift on pre-existing infrastructure? (Choose two.)
 - a. A PXE server
 - b. A DNS server
 - c. A DHCP server
 - d. A Load Balancer for the APIserver

- ▶ 2. Which of the following sentences about connecting to the bootstrap node is correct?
 - a. The Administrator can connect to the bootstrap node via SSH after the installation is completed using the `core` user.
 - b. The Administrator can add a public SSH key to the `install-config.yaml` file for connecting to the bootstrap node and for debugging the installation process.
 - c. The Administrator can create users in the bootstrap node using the `install-config.yaml` file.
 - d. The Administrator must add the SSH public key to the `.ssh/authorized_keys` file on the bootstrap server after the installation has begun.

- ▶ 3. Which of the following sentences about the ignition files is true?
 - a. The ignition files are injected into the different hosts using the `install-config.yaml` file.
 - b. Red Hat recommends creating the ignition files manually from the examples found in the official documentation to avoid mistakes.
 - c. The ignition files are created from the `install-config.yaml` file using the `openshift-install` command.
 - d. The ignition files are created from the `install-config.yaml` file using the `oc` command.

- **4. Which two of the following resources are necessary to complete the installation of RHCOS? (Choose two.)**
- a. The ignition files
 - b. The kickstart files
 - c. The oc CLI tool
 - d. The RHCOS images

Summary

- The network services that are required to install OpenShift without integration with a specific provider.
- The generic architecture that is necessary to install OpenShift without a specific provider.
- How to check and configure the network services for the installation and how to generate the SSH key for troubleshooting.

Chapter 5

Installing OpenShift Without an Infrastructure Provider

Goal

Provision OpenShift clusters without integration with the underlying infrastructure.

Objectives

- Install OpenShift by starting the bootstrap machine and cluster nodes, and monitoring the progress of the installation process.
- Performing the Installation of OpenShift Without an Infrastructure Provider (and Guided Exercise) (and Quiz)

Sections



Performing the Installation of OpenShift Without an Infrastructure Provider

Objectives

- Install OpenShift by starting the bootstrap machine and cluster nodes, and monitoring the progress of the installation process.

Preparing the Cluster Machines for Installation

Each infrastructure node, including the bootstrap, control plane, and compute machines, requires the installation of an operating system before the OpenShift cluster deployment begins. Without an infrastructure provider, the nodes may require manual installation, depending on the particular environment and approach to this task. Preferably, configure a PXE server to perform the installation for each infrastructure node. This installation approach requires using images made available through a file server and accessible by the infrastructure nodes over the network. Generate and supply a common SSH key during these installations for remote access to these infrastructure nodes.

To initiate the PXE installations, network boot each infrastructure node and follow the prompts from the PXE menu. In most scenarios, this installation does not require intervention during the process and occurs rapidly, with a limiting factor on network latency to and from the file server and PXE systems.

When the installation completes, verify a proper installation by testing remote access using the SSH key to ensure each node is properly prepared.

```
[user@demo ~]$ ssh -i ~/.ssh/sample_key.pub core@<HOSTNAME>
```

After the operating system installation process and validation is complete, the machines are ready for an OpenShift cluster deployment.

Initiating the Installation and Monitoring the Progress

When all services and nodes are prepared, you are ready to deploy the OpenShift cluster. After starting the OpenShift installation, use the `openshift-install` command to provide further insights into the process:

```
[user@demo ~]$ openshift-install wait-for bootstrap-complete
```

This command delivers output from the installation process, concluding after the bootstrap phase concludes and the transition to the cluster control plane nodes is complete.

Monitoring the deployment is informative, helping you to identify issues as they occur, or view successful transitions through the various phases of the process. Connecting to the bootstrap node to monitor the bootstrap service process also aids in noticing any issues that arose during the installation. Inspect the services using an SSH connection and the following command:

```
sh-4.2# journalctl -b -f -u release-image.service -u bootkube.service
```

Additionally, if installation failures occur, then gathering and reviewing the logs provides information that assists in remediation. After taking corrective actions to address the underlying cause of the failure, you can try the installation again.

Authenticating with the New Cluster

When the installation completes, the final output messages include information detailing the credentials file path and web console link for accessing the cluster, as shown below:

```
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
  KUBECONFIG=/home/ocp_install_directory/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
  console.apps.ocp4.example.com
INFO Login to the console with user: "kubeadmin", and password: "XXXXXX-XXXXX-
  XXXX-XXXX"
DEBUG Time elapsed per stage:
DEBUG Cluster Operators: 27m11s
INFO Time elapsed: 27m11s
```

During this phase, while the infrastructure control plane nodes are deploying the remainder of the cluster components, access is available in several ways:

- The `kubeadm-password` file provides credentials for the `kubeadmin` user.
- The `kubeconfig` file provides authentication information that can be referenced for CLI interactions using the `oc` command.

Testing this interaction allows for verification of the credentials and configuration, enabling the monitoring and inspection of the remainder of the installation process.

Use any of the following methods to authenticate to the cluster:

- Supply the `kubeadmin` credentials on the web console login page.
- Supply a path to an authentication file using the `oc login` command option `--kubeconfig=/path/to/kubconfig`.
- Use the `oc` command and supply the username, password and server arguments through interactive prompts.
- Configure a `KUBECONFIG` environmental variable that resolves to the path for the `kubeconfig` file and allows for `oc` interactions, directly.

```
[user@demo ~]$ export KUBECONFIG=~/ocp_install_dir/auth/kubeconfig
```

Accessing the cluster using any of these approaches allows for inspection, as well as cluster administration, during the remainder of the installation.

Interacting with the Cluster During Installation

During this portion of the installation, connecting to the control plane nodes in the cluster using `oc debug` allows for monitoring the creation of the containers for the various cluster services. Watching the cluster version, operators, pods, and nodes statuses through simple `oc get` interactions provides insights into the installation progress. The following `watch` command

Chapter 5 | Installing OpenShift Without an Infrastructure Provider

can gather all of this information and allow monitoring of all of these components during the installation:

```
[user@demo ~]$ watch 'oc get clusterversion; oc get clusteroperators; oc get \
> pods --all-namespaces | grep -v -E "Running|Completed"; \
> oc get csr; oc get nodes'
```

Additionally, you can use the following command to monitor cluster events for further details about installation progress:

```
[user@demo ~]$ oc get events -A -w
```

Finalizing the OpenShift Installation

While the installation performs the remainder of the tasks required to complete the deployment, monitoring the cluster installation continues to be helpful for ensuring success.

It is necessary to approve the CSRs for each node in the cluster before the installation can complete. View any pending requests using the following command:

```
[user@demo ~]$ oc get csr
```

Approve the outstanding requests and allow the installation to continue using the following command:

```
[user@demo ~]$ oc adm certificate approve <CSR>
```

After the initial approval of the pending CSRs, several other CSRs are generated and require additional approvals before proper cluster installation can complete. Check for CSRs after several minutes and approve any outstanding requests. These requests continue to queue during this phase of the installation, which cannot complete without these approvals. Outstanding CSR requests waiting for approval can inhibit cluster performance and functionality, so it is imperative to continue to poll and approve any requests that the installation generates.

As the installation proceeds toward completion, utilize the following command to get verbose output of the statuses:

```
[user@demo ~]$ openshift-install --dir=ocp_install_directory wait-for
install-complete --log-level=debug
```

After the installation concludes, credentials and a web console link display and the OpenShift Container Platform is available. At this stage, perform validation checks and test workload deployments to ensure a properly functioning cluster.



References

- For more information on reviewing logs during an installation failure, refer to the *Troubleshooting installation issues* chapter of the Red Hat OpenShift Container Platform 4.6 documentation at <https://docs.openshift.com/container-platform/4.6/installing/installing-troubleshooting.html>

► Guided Exercise

Performing the Installation of OpenShift Without an Infrastructure Provider

- Perform an installation of OpenShift without integration with an infrastructure provider.

Outcomes

You should be able to:

- Run the `openshift-install` tool to initiate a manual installation of OpenShift.
- Monitor various aspects of the deployment during an OpenShift installation without an infrastructure provider.

Before You Begin

To perform this exercise, ensure that you have completed all prior exercises in this course.

The classroom environment is unchanged, apart from the course exercises.

If you have made any additional changes to the environment, recreate a new classroom environment before you begin. You must repeat the steps from the guided exercises in the previous chapters to prepare for this exercise.

Instructions

This guided exercise relies upon the configured environment from the previous chapter. The images are configured for installation of the infrastructure machines via PXE boot. The installation of the operating system on each node is the initial step before proceeding to deploy your cluster.

- ▶ 1. Install the Red Hat Enterprise Linux CoreOS (RHCOS) bootstrap machine using the configured PXE boot environment.
 - 1.1. Reboot the `bootstrap` machine from within the Red Hat Learning portal console by clicking `Ctrl Alt Del` in the top right corner.
 - 1.2. Wait until the reboot is completed and the machine boots into the PXE menu.
 - 1.3. From the PXE menu, press `Enter` to proceed with the installation of Red Hat Enterprise Linux CoreOS on the machine.
- ▶ 2. Perform a PXE-based installation process for each of the `master01`, `master02`, `master03`, `worker01`, and `worker02` nodes by following the restart process described in the previous step.
 - 2.1. Reboot each of the listed machines from within the Red Hat Learning portal console by clicking `Ctrl Alt Del` in the top right corner.
 - 2.2. Wait until the reboot is completed and the machines boot into the PXE menu.
 - 2.3. From the PXE menu, press `Enter` to proceed with the installation of Red Hat Enterprise Linux Core OS on the machine.

**Note**

The **master** nodes can encounter errors during this process as they attempt to retrieve their ignition files through the Kubernetes API until the **bootstrap** node completes installation and begins serving these files for the other nodes.

```
[ 61.852424] ignition[726]: GET error: Get "https://api-int.ocp4.example.com:2623/config/master": EOF"
```

Additionally, the **worker** nodes can encounter similar errors during this process as they attempt to retrieve their ignition files from the production control plane.

```
[ 61.852424] ignition[726]: GET error: Get "https://api-int.ocp4.example.com:2623/config/master": EOF"
```

- 2.4. Press **Enter** to proceed with the installation of Red Hat Enterprise Linux CoreOS on the machine.

**Note**

If the installation process fails to start from the PXE menu, and does not provide any error message, check the **httpd** logs in `/var/log/httpd/access_log`. Typically, you can find HTTP 403 (forbidden) errors if the file permissions are not correct, and HTTP 404 (not found) errors if the RHCOS file names in the pxeboot files are not correct.

**Note**

If the RHCOS installation fails for any reason after starting, click **Reset** on the **ACTION** menu from the **Lab Environment** tab in the Red Hat Learning Portal to perform a full reset of the desired virtual machine.

When these installations conclude, the nodes are properly prepared and ready for the OpenShift cluster deployment.

- 3. Create the OpenShift cluster.

- 3.1. From the **workstation** machine, log in to the **utility** machine as the **lab** user.

```
[student@workstation ~]$ ssh lab@utility  
...output omitted...
```

- 3.2. Initiate the installation and poll the bootstrap machine for status messages regarding the progress of the deployment using the following command:

```
[lab@utility ~]$ openshift-install --dir=./ocp4upi wait-for bootstrap-complete  
INFO Waiting up to 20m0s for the Kubernetes API at https://  
api.ocp4.example.com:6443...
```

```
INFO API v1.19.0+9f84db3 up
INFO Waiting up to 30m0s for bootstrapping to complete...
E1203 02:31:28.643262    2817 reflector.go:307] k8s.io/client-go/tools/watch/informerwatcher.go:146: Failed to watch *v1.ConfigMap: Get "https://api.ocp4.example.com:6443/api/v1/namespaces/kube-system/configmaps?allowWatchBookmarks=true&fieldSelector=metadata.name%3Dbootstrap&resourceVersion=1&timeoutSeconds=401&watch=true": net/http: TLS handshake timeout
E1203 02:38:49.170288    2817 reflector.go:307] k8s.io/client-go/tools/watch/informerwatcher.go:146: Failed to watch *v1.ConfigMap: Get "https://api.ocp4.example.com:6443/api/v1/namespaces/kube-system/configmaps?allowWatchBookmarks=true&fieldSelector=metadata.name%3Dbootstrap&resourceVersion=3331&timeoutSeconds=307&watch=true": net/http: TLS handshake timeout
I1203 02:39:00.172528    2817 trace.go:116] Trace[2001942447]: "Reflector ListAndWatch" name:k8s.io/client-go/tools/watch/informerwatcher.go:146 (started: 2020-12-03 02:38:50.170439408 -0500 EST m+=705.503447896) (total time: 10.002043765s):
Trace[2001942447]: [10.002043765s] [10.002043765s] END
E1203 02:39:00.172549    2817 reflector.go:153] k8s.io/client-go/tools/watch/informerwatcher.go:146: Failed to list *v1.ConfigMap: Get "https://api.ocp4.example.com:6443/api/v1/namespaces/kube-system/configmaps?fieldSelector=metadata.name%3Dbootstrap&limit=500&resourceVersion=0": net/http: TLS handshake timeout
I1203 02:39:11.175147    2817 trace.go:116] Trace[1730834876]: "Reflector ListAndWatch" name:k8s.io/client-go/tools/watch/informerwatcher.go:146 (started: 2020-12-03 02:39:01.172662955 -0500 EST m+=716.505671444) (total time: 10.002460755s):
Trace[1730834876]: [10.002460755s] [10.002460755s] END
E1203 02:39:11.175176    2817 reflector.go:153] k8s.io/client-go/tools/watch/informerwatcher.go:146: Failed to list *v1.ConfigMap: Get "https://api.ocp4.example.com:6443/api/v1/namespaces/kube-system/configmaps?fieldSelector=metadata.name%3Dbootstrap&limit=500&resourceVersion=0": net/http: TLS handshake timeout
E1203 02:39:16.233355    2817 reflector.go:153] k8s.io/client-go/tools/watch/informerwatcher.go:146: Failed to list *v1.ConfigMap: Get "https://api.ocp4.example.com:6443/api/v1/namespaces/kube-system/configmaps?fieldSelector=metadata.name%3Dbootstrap&limit=500&resourceVersion=0": EOF
INFO It is now safe to remove the bootstrap resources
INFO Time elapsed: 18m6s
```

**Note**

The errors shown in the preceding output result from unavailability of the Kubernetes API in the **bootstrap** machine. Once the Kubernetes API is running, these errors will disappear.

After the Kubernetes API becomes available, a message displays that it is now safe to remove the bootstrap resources.

► 4. Remove the **bootstrap machine from the load balancer.**

- 4.1. Edit the `/etc/haproxy/haproxy.cfg` file and remove the **bootstrap** machine entries. Find and remove the following entries:

```
#-----  
# round robin balancing for RHOC P Kubernetes API Server  
#-----  
  
server bootstrap 192.168.50.9:6443 check  
  
# -----  
# round robin balancing for RHOC P Machine Config Server  
# -----  
  
server bootstrap 192.168.50.9:22623 check
```

**Note**

The haproxy.cfg file syntax can be validated with the following command:

```
haproxy -c -f /etc/haproxy/haproxy.cfg
```

- 4.2. Reload the HAProxy configuration.

```
[lab@utility ~]$ sudo systemctl reload haproxy
```

- 5. Inspect the logs on the bootstrap machine by connecting via SSH to view the progress of the installation. Wait for the bootkube service to report a status of complete.

```
[lab@utility ~]$ ssh -i .ssh/ocp4upi core@bootstrap.ocp4.example.com  
  
Red Hat Enterprise Linux CoreOS 46.82.202010091720-0  
Part of OpenShift 4.6, RHCOS is a Kubernetes native operating system  
managed by the Machine Config Operator (`clusteroperator/machine-config`).  
  
WARNING: Direct SSH access to machines is not recommended; instead,  
make configuration changes via `machineconfig` objects:  
https://docs.openshift.com/container-platform/4.6/architecture/architecture-  
rhcos.html  
  
---  
This is the bootstrap node; it will be destroyed when the master is fully up.  
  
The primary services are release-image.service followed by bootkube.service. To  
watch their status, run e.g.  
  
journalctl -b -f -u release-image.service -u bootkube.service  
Last login: Thu Dec  3 08:10:24 2020 from 192.168.50.254  
[core@bootstrap ~]$ journalctl -b -f -u release-image.service -u bootkube.service  
-- Logs begin at Thu 2020-12-03 07:27:53 UTC. --  
Dec 03 07:45:01 bootstrap bootkube.sh[2314]: Skipped "secret-initial-kube-  
controller-manager-service-account-private-key.yaml" secrets.v1./initial-service-  
account-private-key -n openshift-config as it already exists
```

Chapter 5 | Installing OpenShift Without an Infrastructure Provider

```
Dec 03 07:45:03 bootstrap bootkube.sh[2314]: Sending bootstrap-finished
event.Tearing down temporary bootstrap control plane...
Dec 03 07:45:03 bootstrap bootkube.sh[2314]: Waiting for CEO to finish...
Dec 03 07:45:03 bootstrap bootkube.sh[2314]: I1203 07:45:03.848000      1
  waitforceo.go:64] Cluster etcd operator bootstrapped successfully
Dec 03 07:45:03 bootstrap bootkube.sh[2314]: I1203 07:45:03.849938      1
  waitforceo.go:58] cluster-etcd-operator bootstrap etcd
Dec 03 07:45:03 bootstrap bootkube.sh[2314]: bootkube.service complete
```

Close the connection to the bootstrap machine.

```
[core@bootstrap ~]$ exit
[lab@utility ~]$
```

▶ **6.** Configure access to the cluster using the `kubeconfig` file generated during installation.

- 6.1. Configure a `KUBECONFIG` environmental variable using the `kubeconfig` authentication file generated during installation.

```
[lab@utility ~]$ export KUBECONFIG=~/ocp4upi/auth/kubeconfig
```

- 6.2. Test the configured credentials and inspect the current cluster status.

- Credentials

```
[lab@utility ~]$ oc whoami
system:admin
```

- Nodes

```
[lab@utility ~]$ oc get nodes
master01  Ready    master    76m    v1.19.0+9f84db3
master02  Ready    master    70m    v1.19.0+9f84db3
master03  Ready    master    69m    v1.19.0+9f84db3
```

- Cluster Version. Because the cluster installation is in progress, you can ignore the status message.

```
[lab@utility ~]$ oc get clusterversion
NAME      VERSION      AVAILABLE      PROGRESSING      SINCE      STATUS
version          False        True           84m      Unable to apply 4.6.4: some
cluster operators have not yet rolled out
```

- Cluster Operators. Because the cluster installation is in progress, you can ignore the available and degraded status.

```
[lab@utility ~]$ oc get clusteroperator
NAME                  VERSION      AVAILABLE      PROGRESSING
DEGRADED      SINCE
authentication                      False        False
True          78m
```

cloud-credential	4.6.4	True	False
False 84m			
config-operator	4.6.4	True	False
False 79m			
console	4.6.4	Unknown	True
False 68m			
csi-snapshot-controller	4.6.4	True	False
False 78m			
<i>...output omitted...</i>			

► 7. Complete the installation.

- 7.1. Using a separate terminal window, log in to the utility server as the lab user and launch the `openshift-install` command and monitor the status output for messages regarding the installation completion.

```
[lab@utility ~]$ export KUBECONFIG=/ocp4upi/auth/kubeconfig
[lab@utility ~]$ openshift-install --dir=ocp4upi wait-for install-complete \
> --log-level=debug
DEBUG OpenShift Installer 4.6.4
DEBUG Built from commit 6e02d049701437fa81521fe981405745a62c86c5
DEBUG Loading Install Config...
DEBUG Loading SSH Key...
DEBUG Loading Base Domain...
DEBUG Loading Platform...
DEBUG Loading Cluster Name...
DEBUG Loading Base Domain...
DEBUG Loading Platform...
DEBUG Loading Pull Secret...
DEBUG Loading Platform...
DEBUG Using Install Config loaded from state file
INFO Waiting up to 40m0s for the cluster at https://api.ocp4.example.com:6443 to
initialize...
```



Note

The current installation is pending progress because it is necessary to approve pending CSRs before the procedure continues. Proceed to the next step to address this aspect of the installation.

► 8. Approve any pending CSRs.

- 8.1. Return to the original terminal window and view the current status of the cluster CSRs.

```
[lab@utility ~]$ oc get csr
NAME          AGE     SIGNERNAME             REQUESTOR
              CONDITION
csr-2hm74    39m     kubernetes.io/kube-apiserver-client-kubelet
              system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
              Pending
csr-58zsz    101m    kubernetes.io/kube-apiserver-client-kubelet
              system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
              Pending
csr-75c4j    55m     kubernetes.io/kube-apiserver-client-kubelet
              system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
              Pending
csr-7zx55    8m59s   kubernetes.io/kube-apiserver-client-kubelet
              system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
              Pending
...output omitted...
```

8.2. Approve any pending CSRs using the following command.

```
[lab@utility ~]$ oc get csr -o go-template='{{range .items}}{{if not .status}}
{{.metadata.name}}\n{{end}}{{end}}' | xargs oc adm certificate approve
certificatesigningrequest.certificates.k8s.io/csr-2hm74 approved
certificatesigningrequest.certificates.k8s.io/csr-58zsz approved
certificatesigningrequest.certificates.k8s.io/csr-75c4j approved
certificatesigningrequest.certificates.k8s.io/csr-7mrvd approved
certificatesigningrequest.certificates.k8s.io/csr-7zx55 approved
...output omitted...
```

**Note**

The previous command does not contain any line breaks.

It is often necessary to run the previous approval command several times to ensure all generated CSRs are approved. The additional CSRs can take several minutes to submit their requests. Check the status using the `oc get csr` command until no further CSRs appear. This command will return an error when no pending CSRs exist and you are ready to proceed:

**Note**

The initial execution of the CSR approval command in the preceding step is required to approve the kubelet service for the nodes. A subsequent execution of this command is required to approve the node machine objects.

- ▶ 9. Return to the secondary terminal and monitor the installation output until it completes. The installation will take around 20 minutes to finish.

```
[lab@utility ~]$ openshift-install --dir=ocp4upi wait-for install-complete  
--log-level=debug  
  
...output omitted...  
DEBUG Still waiting for the cluster to initialize: Some cluster operators are  
still updating: authentication, console, ingress, kube-storage-version-migrator,  
monitoring  
DEBUG Still waiting for the cluster to initialize: Working towards 4.6.4: 99%  
complete  
DEBUG Still waiting for the cluster to initialize: Multiple errors are preventing  
progress:  
* Cluster operator authentication is reporting a failure:  
WellKnownReadyControllerDegraded: kube-apiserver oauth endpoint  
https://192.168.50.12:6443/.well-known/oauth-authorization-server is not yet  
served and authentication operator keeps waiting (check kube-apiserver operator,  
and check that instances roll out successfully, which can take several minutes  
per instance)  
* Cluster operator console is reporting a failure: RouteHealthDegraded: route not  
yet available, https://console-openshift-console.apps.ocp4.example.com/health  
returns '503 Service Unavailable'  
DEBUG Still waiting for the cluster to initialize: Working towards 4.6.4: 99%  
complete  
DEBUG Still waiting for the cluster to initialize: Working towards 4.6.4: 100%  
complete  
DEBUG Cluster is initialized  
INFO Waiting up to 10m0s for the openshift-console route to be created...  
DEBUG Route found in openshift-console namespace: console  
DEBUG Route found in openshift-console namespace: downloads  
DEBUG OpenShift console route is created  
INFO Install complete!  
INFO To access the cluster as the system:admin user when using 'oc', run 'export  
KUBECONFIG=/home/lab/ocp4upi/auth/kubeconfig'  
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.ocp4.example.com  
INFO Login to the console with user: "kubeadmin", and password: "GQabC-Sva8j-nF7r7-M4Ui6"  
DEBUG Time elapsed per stage:  
DEBUG Cluster Operators: 27m11s  
INFO Time elapsed: 27m11s
```



Note

The failure messages in the output from the previous step result from the kube-apiserver pods not being available yet. After the kube-apiserver pods are deployed on the control plane nodes, the cluster installation is finished.

After the install script completes, you are provided with authentication information and a link to the OpenShift web console.

- ▶ 10. Test access to the OpenShift web console.

- 10.1. Right-click the console link from the installation output or visit `https://console-openshift-console.apps.ocp4.example.com` to test access.
 - 10.2. Use the password provided from the installation output or contained in the file `/home/lab/ocp4upi/auth/kubeadmin-password` on the utility server to log in as the `kubeadmin` user.
- 11. Perform a comparison of the cluster status output with that gathered during the installation process.
- Nodes

```
[lab@utility ~]$ oc get nodes
NAME     STATUS   ROLES    AGE     VERSION
master01  Ready    master   173m    v1.19.0+9f84db3
master02  Ready    master   167m    v1.19.0+9f84db3
master03  Ready    master   166m    v1.19.0+9f84db3
worker01  Ready    worker   14m     v1.19.0+9f84db3
worker02  Ready    worker   14m     v1.19.0+9f84db3
```

- Cluster Version

```
[lab@utility ~]$ oc get clusterversion
NAME      VERSION  AVAILABLE  PROGRESSING  SINCE    STATUS
version   4.6.4    True       False        3m14s   Cluster version is 4.6.4
```

- Cluster Operators

```
[lab@utility ~]$ oc get clusteroperator
NAME          VERSION  AVAILABLE  PROGRESSING
DEGRADED      SINCE
authentication 4.6.4    True       False
cloud-credential 4.6.4    True       False
cluster-autoscaler 4.6.4    True       False
config-operator 4.6.4    True       False
console        4.6.4    True       False
csi-snapshot-controller 4.6.4    True       False
...output omitted...
```



Note

After the installation completes, all cluster operators show the status `AVAILABLE=true`, `PROGRESSING=False`, and `DEGRADED=False`. Proceed with the cluster verification steps to ensure a proper installation. For more information on verifying the new cluster, please review the content in the *Verifying the Installation of OpenShift on AWS* section of the *Installing OpenShift on a Cloud Provider* chapter.

Finish

Do not make any other changes to the lab environment until the next guided exercise. You will continue using it in later guided exercises.

This concludes the guided exercise.

► Quiz

Chapter Review: Installing OpenShift Without an Infrastructure Provider

Choose the correct answers to the following questions:

- ▶ 1. Which two methods of cluster authentication are available after the OpenShift installation is finished? (Choose two).
 - a. htpasswd authentication with the `admin` user.
 - b. Export the `KUBECONFIG` environment variable using the `/auth/kubeconfig` file in the OpenShift installation folder.
 - c. As the `admin` user with the credentials stored in the `/auth/kubeadmin` file in the OpenShift installation folder.
 - d. As the `kubeadmin` user with the password stored in the `/auth/kubeadmin-password` file in the OpenShift installation folder.

- ▶ 2. Which command can be used for authenticating into OpenShift from the host where the installation was performed after the installation is finished?
 - a. `oc login -u system -p admin`
 - b. `oc login -u kubeadmin -p kubeadmin`
 - c. `oc export KUBECONFIG && oc login`
 - d. `export KUBECONFIG=~/ocp4_installation_folder/auth/kubeconfig`

- ▶ 3. Which two of the following sentences about the `openshift-install` command are correct? (Choose two).
 - a. The `openshift-install` generates the Kubernetes manifests.
 - b. The `openshift-install` generates the RHCOS kickstart files.
 - c. The `openshift-install` command generates the RHCOS ignition files.
 - d. The `openshift-install` command must be run as `root`.
 - e. The `openshift-install` command automatically authenticates the `kubeadmin` user to OpenShift after the installation is complete.

► **4. Which three of the following statements are true about the Certificate Signing Requests (CSRs) during the OpenShift installation without an Infrastructure Provider (Choose three).**

- a. All the CSRs for the control plane nodes are automatically issued and accepted during the installation.
- b. All the CSRs for any nodes are automatically issued and accepted during the installation.
- c. The CSRs of the compute nodes must be approved manually.
- d. Administrators need to approve two different CSRs for each compute node to join the cluster properly.
- e. Administrators need to approve only one CSR for each compute node to join the cluster properly.

► Solution

Chapter Review: Installing OpenShift Without an Infrastructure Provider

Choose the correct answers to the following questions:

- ▶ 1. Which two methods of cluster authentication are available after the OpenShift installation is finished? (Choose two).
 - a. htpasswd authentication with the admin user.
 - b. Export the KUBECONFIG environment variable using the /auth/kubeconfig file in the OpenShift installation folder.
 - c. As the admin user with the credentials stored in the /auth/kubeadmin file in the OpenShift installation folder.
 - d. As the kubeadmin user with the password stored in the /auth/kubeadmin-password file in the OpenShift installation folder.

- ▶ 2. Which command can be used for authenticating into OpenShift from the host where the installation was performed after the installation is finished?
 - a. oc login -u system -p admin
 - b. oc login -u kubeadmin -p kubeadmin
 - c. oc export KUBECONFIG && oc login
 - d. export KUBECONFIG=~/ocp4_installation_folder/auth/kubeconfig

- ▶ 3. Which two of the following sentences about the openshift-install command are correct? (Choose two).
 - a. The openshift-install generates the Kubernetes manifests.
 - b. The openshift-install generates the RHCOS kickstart files.
 - c. The openshift-install command generates the RHCOS ignition files.
 - d. The openshift-install command must be run as root.
 - e. The openshift-install command automatically authenticates the kubeadmin user to OpenShift after the installation is complete.

► **4. Which three of the following statements are true about the Certificate Signing Requests (CSRs) during the OpenShift installation without an Infrastructure Provider (Choose three).**

- a. All the CSRs for the control plane nodes are automatically issued and accepted during the installation.
- b. All the CSRs for any nodes are automatically issued and accepted during the installation.
- c. The CSRs of the compute nodes must be approved manually.
- d. Administrators need to approve two different CSRs for each compute node to join the cluster properly.
- e. Administrators need to approve only one CSR for each compute node to join the cluster properly.

Summary

- The installation process for deploying OpenShift without any specific infrastructure provider.
- The command line interactions using the 'openshift-install' binary that perform the installation.
- The methods of monitoring and inspecting the OpenShift installation during the procedure.
- The available command line interactions that provide process validation during the installation.
- The techniques that provide the most verbose output during the installation of OpenShift.
- The precise interactions, output, and information that occurs during a successful OpenShift installations.

Chapter 6

Completing the Installation of OpenShift Without an Infrastructure Provider

Goal

Perform essential tasks that are required before onboarding users and applications on a newly provisioned OpenShift cluster.

Objectives

- Perform required customizations before onboarding users and applications into a newly installed cluster.
- Backup and restore a control plane node.

Sections

- Performing Day 1 and Day 2 Operations (and Guided Exercise)
- Replacing a Control Plane Node (and Guided Exercise) (and Quiz)

Performing Day 1 and Day 2 Operations

Objectives

- Perform required customizations before onboarding users and applications into a newly installed cluster.

Explaining Day 1 and Day 2 Operations

In production environments, administrators must make OpenShift clusters ready for the end users. The set of tasks typically performed right after completing the installation are commonly called Day 1 and Day 2 operations.

Day 1 operations are the customizations performed during the installation, whereas Day 2 operations are customizations done after the installation is complete and the cluster is made available to the end users.

The number of Day 1 and Day 2 customizations is fluid, depending on the needs of the company, the cluster administrators, and the end users.

Describing Usual Day 1 and Day 2 Operations for OpenShift Clusters

Before releasing an OpenShift cluster to production, administrators must perform one or more of the following tasks:

- Add persistent storage for the OpenShift image registry or configure the corporate registry.
- Add persistent storage for the monitoring stack to preserve metrics for future analysis.
- Configure an authentication and authorization provider.
- Configure a dynamic storage provider for the persistent storage of the containerized applications.
- Run functional tests after completing the desired configurations.

Furthermore, the cluster might require customizations, as explained in Red Hat DO280: *Red Hat OpenShift Administration II: Operating a Production Kubernetes Cluster* and DO380: *Red Hat OpenShift Administration III: Scaling Kubernetes Deployments in the Enterprise*. Customizations that you can apply to the cluster include:

- Setting up network policies (DO280).
- Controlling the pods scheduling (DO280).
- Adding resource limits (DO280).
- Installing and configuring the cluster logging (DO380).
- Configuring OpenShift alerts (DO380).
- Adding a custom CA (DO380).

Chapter 6 | Completing the Installation of OpenShift Without an Infrastructure Provider

- Making different operators available for the end users (DO380).
- Creating custom Machine Config Pools (DO380).
- Recovering failed worker nodes (DO380).

You can find more details on various Day 1 and Day 2 customizations in the *Post-installation configuration Guide* at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/post-installation_configuration/index

You learn how to backup and restore a failed Control Plane node elsewhere in this course.

Completing the Installation

After the cluster installation is completed, you must still configure the OpenShift cluster in the classroom environment before considering it fully functional. This configuration steps are required for every OpenShift installation on pre-existing infrastructure. After the `openshift-install` command finishes, administrators might need to complete one or more of the following tasks:

- Make a copy of the `kubeconfig` file and save it.
- Configure the image registry operator, adding ephemeral storage or persistent storage.
- Add a dynamic storage provider.
- Configure an authentication and authorization provider.
- Perform functional testing of the cluster.



Warning

Save the `kubeconfig` file in a secure location. This file grants `cluster-admin` privileges in the cluster. Handle it accordingly.



References

For more information, refer to the *Post-installation configuration Guide* at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/post-installation_configuration/index

► Guided Exercise

Performing Day 1 and Day 2 Operations

- Perform a basic configuration before releasing an OpenShift cluster to production.

Outcomes

You should be able to:

- Save a copy of the kubeconfig file.
- Configure a dynamic storage provider.
- Configure the registry operator, adding persistent storage.
- Deploy an application to perform a functional test of the cluster.

Before You Begin

To perform this exercise, ensure you have completed the `install-perform-practice` guided exercise.

Instructions

► 1. Save the kubeconfig file.

- 1.1. As the student user, copy the kubeconfig file from `/home/lab/ocp4upi/auth/kubeconfig` to the workstation machine.

```
[student@workstation ~]$ ssh lab@utility

[lab@utility ~]$ scp /home/lab/ocp4upi/auth/kubeconfig student@workstation:
The authenticity of host 'workstation (172.25.250.9)' can't be established.
ECDSA key fingerprint is SHA256:DDmfpJODDjGdZHRAhDuN6XyPrL43F4IXMYh5orlnQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'workstation,172.25.250.9' (ECDSA) to the list of known
hosts.
student@workstation's password: student
kubeconfig                                         100%  8942      5.2MB/s   00:00
```

► 2. Configure the image registry with persistent storage.

- 2.1. Use the kubeconfig file to authenticate to OpenShift.

```
[lab@utility ~]$ export KUBECONFIG=~/ocp4upi/auth/kubeconfig
```

- 2.2. Investigate the NFS exports on the utility server.

```
[lab@utility ~]$ cat /etc/exports
(exports *(rw,sync,no_wdelay,no_root_squash,insecure,fsid=0)
```

```
[lab@utility ~]$ lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda    252:0    0   10G  0 disk
└─vda1 252:1    0    1M  0 part
└─vda2 252:2    0  100M  0 part /boot/efi
└─vda3 252:3    0  9,9G  0 part /
vdb    252:16   0   40G  0 disk
└─vdb1 252:17   0   40G  0 part /exports

[lab@utility ~]$ df -h
Filesystem      Size  Used Avail Use% Mounted on
...output omitted...
/dev/vdb1        40G  318M   40G   1% /exports
/dev/vda2     100M   6,8M   94M   7% /boot/efi
tmpfs          183M     0  183M   0% /run/user/1000
```

The /exports NFS export has approximately 40 GB of disk space available.

2.3. Create the persistent volume (PV) file named `pv.yaml`.

```
[lab@utility ~]$ vi pv.yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: registry-pv
spec:
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteMany
  nfs:
    path: /exports/registry
    server: 192.168.50.254
  persistentVolumeReclaimPolicy: Recycle
```

Save the file and then close the editor. You can retrieve that file by running the `wget http://classroom.example.com/materials/solutions/etherpad/pv.yaml` command.

2.4. Create the /exports/registry folder with the appropriate permissions and the persistent volume (PV) from the `pv.yaml` file:

```
[lab@utility ~]$ mkdir /exports/registry
```

```
[lab@utility ~]$ sudo chmod 777 /exports/registry/
```

```
[lab@utility ~]$ oc create -f pv.yaml
persistentvolume/registry-pv created
```

2.5. Create a file named `pvc.yaml` with the following content:

```
[lab@utility ~]$ vi pvc.yaml
kind: PersistentVolumeClaim
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: registry-claim
  namespace: openshift-image-registry
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 5Gi
```

Save the file and then close the editor. You can retrieve that file by running the wget <http://classroom.example.com/materials/solutions/etherpad/pvc.yaml> command.

- 2.6. Create the persistent volume claim (PVC) from the pvc.yaml file:

```
[lab@utility ~]$ oc create -f pvc.yaml
persistentvolumeclaim/registry-claim created
```

- 2.7. In the spec section, set the image registry operator in a "Managed" state. Edit the configuration of the cluster image registry to add the PVC. Also, configure the image registry to have two pod replicas.

```
[lab@utility ~]$ oc edit configs.imageregistry/cluster
...output omitted...
spec:
...output omitted...
  managementState: Managed
...output omitted...
  proxy: {}
  replicas: 2
  requests:
...output omitted...
  rolloutStrategy: RollingUpdate
  storage:
    pvc:
      claim: registry-claim
...output omitted...
```

**Note**

Ensure you are editing the **spec** section, not the **status** section. Editing the **status** section does not have any effect in the configuration.

- 2.8. Check the PV, PVC, and image registry operator status.

```
[lab@utility ~]$ oc get pv -A
NAME          CAPACITY   ACCESS MODES  RECLAIM POLICY  STATUS    CLAIM
              STORAGECLASS      AGE
registry-pv   5Gi        RWX           Recycle       Bound     openshift-image-
registry/registry-claim                         111m

[lab@utility ~]$ oc get pvc -A
NAMESPACE          NAME      STATUS   VOLUME   CAPACITY
ACCESS MODES     STORAGECLASS   AGE
openshift-image-registry   registry-claim   Bound   registry-pv   5Gi     RWX
                                         105m

[lab@utility ~]$ oc get clusteroperator
NAME                           VERSION  AVAILABLE  PROGRESSING
DEGRADED   SINCE
authentication                   4.6.4    True       False
  False    174m
cloud-credential                 4.6.4    True       False
  False    3d
cluster-autoscaler                4.6.4    True       False
  False    3d
config-operator                  4.6.4    True       False
  False    3d
console                        4.6.4    True       False
  False    175m
csi-snapshot-controller            4.6.4    True       False
  False    3d
dns                            4.6.4    True       False
  False    3d
etcd                           4.6.4    True       False
  False    3d
image-registry                  4.6.4    True       False
  False    99m
...output omitted...
```

- 2.9. Check the number of registry pods in the `openshift-image-registry` namespace.

```
[lab@utility ~]$ oc get pods -n openshift-image-registry -o wide
NAME                               READY   STATUS    RESTARTS
AGE      IP           NODE
cluster-image-registry-operator-8455b85cc6-z7wcv   1/1    Running   1
  78m    10.130.0.18   master03 ...
image-registry-76467959f4-kmdf8   1/1    Running   0
  8m38s   10.131.0.21   worker01 ...
image-registry-76467959f4-tcczp   1/1    Running   0
  9m45s   10.128.2.39   worker02 ...
node-ca-dczt7                     1/1    Running   0
  66m    192.168.50.14  worker02 ...
node-ca-gsbtk                    1/1    Running   0
  67m    192.168.50.13  worker01 ...
```

node-ca-mpbkm	192.168.50.12	master03	...	1/1	Running	0
node-ca-xjbp8	192.168.50.10	master01	...	1/1	Running	0
node-ca-xwfsf	192.168.50.11	master02	...	1/1	Running	0

► 3. Configure a dynamic storage provisioner.

- 3.1. Clone the `kubernetes-sigs/nfs-subdir-external-provisioner` GitHub repository:

```
[lab@utility ~]$ git clone \
> https://github.com/kubernetes-sigs/nfs-subdir-external-provisioner/
...output omitted...
```



Warning

The `nfs-subdir-external-provisioner` is not supported by Red Hat nor recommended for production environments.

- 3.2. Edit the files in the `deploy` folder using information for the NFS server in `utility`. The output should be as follows:

```
[lab@utility ~]$ vi nfs-subdir-external-provisioner/deploy/class.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: nfs-client
  annotations:
    storageclass.kubernetes.io/is-default-class: "true"
provisioner: nfs-dynamic-provisioner
reclaimPolicy: Retain
...output omitted...
```

```
[lab@utility ~]$ vi nfs-subdir-external-provisioner/deploy/rbac.yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: nfs-client-provisioner
  namespace: nfs-dynamic-namespace
---
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: nfs-client-provisioner-runner
rules:
  - apiGroups: []
    resources: ["persistentvolumes"]
    verbs: ["get", "list", "watch", "create", "delete"]
  - apiGroups: []
    resources: ["persistentvolumeclaims"]
```

Chapter 6 | Completing the Installation of OpenShift Without an Infrastructure Provider

```
verbs: ["get", "list", "watch", "update"]
- apiGroups: ["storage.k8s.io"]
  resources: ["storageclasses"]
  verbs: ["get", "list", "watch"]
- apiGroups: []
  resources: ["events"]
  verbs: ["create", "update", "patch"]
---

kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: run-nfs-client-provisioner
subjects:
- kind: ServiceAccount
  name: nfs-client-provisioner
  namespace: nfs-dynamic-namespace
roleRef:
  kind: ClusterRole
  name: nfs-client-provisioner-runner
  apiGroup: rbac.authorization.k8s.io
---

kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: leader-locking-nfs-client-provisioner
  namespace: nfs-dynamic-namespace
rules:
- apiGroups: []
  resources: ["endpoints"]
  verbs: ["get", "list", "watch", "create", "update", "patch"]
---

kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: leader-locking-nfs-client-provisioner
  namespace: nfs-dynamic-namespace
subjects:
- kind: ServiceAccount
  name: nfs-client-provisioner
  namespace: nfs-dynamic-namespace
roleRef:
  kind: Role
  name: leader-locking-nfs-client-provisioner
  apiGroup: rbac.authorization.k8s.io
```

```
[lab@utility ~]$ vi nfs-subdir-external-provisioner/deploy/deployment.yaml
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nfs-client-provisioner
  labels:
    app: nfs-client-provisioner
    namespace: nfs-dynamic-namespace
```

```
spec:
  replicas: 1
  strategy:
    type: Recreate
  selector:
    matchLabels:
      app: nfs-client-provisioner
  template:
    metadata:
      labels:
        app: nfs-client-provisioner
    spec:
      serviceAccountName: nfs-client-provisioner
      containers:
        - name: nfs-client-provisioner
          image: registry.k8s.io/sig-storage/nfs-subdir-external-
provisioner:v4.0.2
          volumeMounts:
            - name: nfs-client-root
              mountPath: /persistentvolumes
      env:
        - name: PROVISIONER_NAME
          value: nfs-dynamic-provisioner
        - name: NFS_SERVER
          value: 192.168.50.254
        - name: NFS_PATH
          value: /exports
      volumes:
        - name: nfs-client-root
          nfs:
            server: 192.168.50.254
            path: /exports
```

3.3. Create the objects in the nfs-dynamic-namespace namespace.

```
[lab@utility ~]$ oc create namespace nfs-dynamic-namespace
namespace/nfs-dynamic-namespace created
```

```
[lab@utility ~]$ oc create -f nfs-subdir-external-provisioner/deploy/rbac.yaml
serviceaccount/nfs-client-provisioner created
clusterrole.rbac.authorization.k8s.io/nfs-client-provisioner-runner created
clusterrolebinding.rbac.authorization.k8s.io/run-nfs-client-provisioner created
role.rbac.authorization.k8s.io/leader-locking-nfs-client-provisioner created
rolebinding.rbac.authorization.k8s.io/leader-locking-nfs-client-provisioner
created
```

3.4. Create the use-scc-hostmount-anyuid role in the nfs-dynamic-namespace namespace. Add the use-scc-hostmount-anyuid role to the nfs-client-provisioner service account.

```
[lab@utility ~]$ oc create role use-scc-hostmount-anyuid --verb=use \
> --resource=scc --resource-name=hostmount-anyuid -n nfs-dynamic-namespace
role.rbac.authorization.k8s.io/use-scc-hostmount-anyuid created
```

```
[lab@utility ~]$ oc project nfs-dynamic-namespace
Now using project "nfs-dynamic-namespace" on server "https://
api.ocp4.example.com:6443".
```

```
[lab@utility ~]$ oc adm policy add-role-to-user use-scc-hostmount-anyuid \
> -z nfs-client-provisioner --role=namespace='nfs-dynamic-namespace'
role.rbac.authorization.k8s.io/use-scc-hostmount-anyuid added: "nfs-client-
provisioner"
```

3.5. Create the deployment.

```
[lab@utility ~]$ oc create -f \
> nfs-subdir-external-provisioner/deploy/deployment.yaml
deployment.apps/nfs-client-provisioner created
```

3.6. Verify all the resources created in the nfs-dynamic-namespace namespace.

```
[lab@utility ~]$ oc get all
NAME                                     READY   STATUS    RESTARTS   AGE
pod/nfs-client-provisioner-69785747-h66hh   1/1     Running   0          25s

NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nfs-client-provisioner   1/1     1           1           25s

NAME                           DESIRED   CURRENT   READY   AGE
replicaset.apps/nfs-client-provisioner-69785747   1         1         1         25s
```

3.7. Create the NFS StorageClass.

```
[lab@utility ~]$ oc create -f \
> nfs-subdir-external-provisioner/deploy/class.yaml
storageclass.storage.k8s.io/nfs-client created
```

```
[lab@utility ~]$ oc get storageclass
NAME          PROVISIONER          RECLAIMPOLICY   VOLUMEBINDINGMODE
ALLOWVOLUMEEXPANSION   AGE
nfs-client (default)   nfs-dynamic-provisioner   Retain        Immediate
false           46s
```

3.8. Test the NFS dynamic volume provisioning.

```
[lab@utility ~]$ oc create -f \
> nfs-subdir-external-provisioner/deploy/test-claim.yaml
persistentvolumeclaim/test-claim created
```

```
[lab@utility ~]$ oc get pvc
NAME           STATUS  VOLUME
MODES   STORAGECLASS AGE
test-claim     Bound    pvc-7c628580-6de8-416c-8750-9e3d315e2e69  1Mi        RWX
nfs-client     17s
```

```
[lab@utility ~]$ oc create -f \
> nfs-subdir-external-provisioner/deploy/test-pod.yaml
pod/test-pod created
```

```
[lab@utility ~]$ oc get all
NAME                           READY  STATUS   RESTARTS  AGE
pod/nfs-client-provisioner-69785747-h66hh  1/1    Running  0          13m
pod/test-pod                  0/1    Completed 0          9s

NAME                           READY  UP-TO-DATE  AVAILABLE  AGE
deployment.apps/nfs-client-provisioner  1/1    1           1          13m

NAME                           DESIRED  CURRENT  READY  AGE
replicaset.apps/nfs-client-provisioner-69785747  1        1        1       13m
```

- 3.9. If the NFS dynamic provisioner is working properly, there will be a SUCCESS file in the NFS export folder.

```
[lab@utility ~]$ tree /exports/
(exports/
└── nfs-dynamic-namespace-test-claim-pvc-7c628580-6de8-416c-8750-9e3d315e2e69
    └── SUCCESS
└── registry

2 directories, 1 file
```



Note

If the creation of the test PVC or test pod fails, then verify the changes made to the YAML files in the previous steps. If you forgot to change a parameter, then delete the resources using the `oc delete -f` command and recreate them using the correct YAML file.

- 4. Perform a functional test of the cluster.

- 4.1. Create a new project named `etherpad` and set it as the working project.

```
[lab@utility ~]$ oc new-project etherpad
...output omitted...
```

- 4.2. Create a new folder named `etherpad` and change into it.

```
[lab@utility ~]$ mkdir etherpad  
[lab@utility ~]$ cd etherpad  
[lab@utility etherpad]$
```

4.3. Download the following files and save the content into the etherpad folder.

```
[lab@utility etherpad]$ SOL_URL=http://classroom.example.com/materials/solutions  
[lab@utility etherpad]$ wget ${SOL_URL}/etherpad/etherpad-svc.yaml  
...output omitted...  
[lab@utility etherpad]$ cat etherpad-svc.yaml  
apiVersion: v1  
kind: Service  
metadata:  
  name: etherpad  
  labels:  
    app.kubernetes.io/name: etherpad  
    app.kubernetes.io/version: "latest"  
spec:  
  type: ClusterIP  
  ports:  
    - port: 9001  
      targetPort: http  
      protocol: TCP  
      name: http  
  selector:  
    app.kubernetes.io/name: etherpad
```

```
[lab@utility etherpad]$ wget ${SOL_URL}/etherpad/etherpad-route.yaml  
...output omitted...  
[lab@utility etherpad]$ cat etherpad-route.yaml  
apiVersion: route.openshift.io/v1  
kind: Route  
metadata:  
  annotations:  
    openshift.io/host.generated: "true"  
  name: etherpad  
  labels:  
    app.kubernetes.io/name: etherpad  
    app.kubernetes.io/version: "latest"  
spec:  
  host:  
  port:  
    targetPort: http  
  to:  
    kind: Service  
    name: etherpad  
    weight: 100  
  tls:  
    insecureEdgeTerminationPolicy: Redirect  
    termination: edge
```

```
[lab@utility etherpad]$ wget ${SOL_URL}/etherpad/etherpad-pvc.yaml
...output omitted...
[lab@utility etherpad]$ cat etherpad-pvc.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: etherpad
  labels:
    app.kubernetes.io/name: etherpad
    app.kubernetes.io/version: "latest"
spec:
  accessModes:
    - "ReadWriteOnce"
  resources:
    requests:
      storage: "1Gi"
```

```
[lab@utility etherpad]$ wget ${SOL_URL}/etherpad/etherpad-deployment.yaml
...output omitted...
[lab@utility etherpad]$ cat etherpad-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: etherpad
  labels:
    app.kubernetes.io/name: etherpad
    app.kubernetes.io/version: "latest"
spec:
  replicas: 1
  selector:
    matchLabels:
      app.kubernetes.io/name: etherpad
  template:
    metadata:
      labels:
        app.kubernetes.io/name: etherpad
    spec:
      securityContext:
        {}
      containers:
        - env:
            - name: TITLE
              value: DO322 Etherpad
            - name: DEFAULT_PAD_TEXT
              value: Etherpad for sharing ideas between the students.
          name: etherpad
          securityContext:
            {}
          image: "quay.io/redhattraining/etherpad:latest"
          imagePullPolicy: IfNotPresent
          ports:
            - name: http
              containerPort: 9001
```

```

        protocol: TCP
livenessProbe:
  httpGet:
    path: /
    port: http
readinessProbe:
  httpGet:
    path: /
    port: http
resources:
  {}
volumeMounts:
  - name: etherpad-data
    mountPath: /opt/etherpad-lite/var
volumes:
  - name: etherpad-data
    persistentVolumeClaim:
      claimName: etherpad

```

- 4.4. Create the OpenShift resources using the files that you created in the `etherpad` folder.

```
[lab@utility etherpad]$ oc create -f etherpad-pvc.yaml
persistentvolumeclaim/etherpad created
```

```
[lab@utility etherpad]$ oc create -f etherpad-svc.yaml
service/etherpad created
```

```
[lab@utility etherpad]$ oc create -f etherpad-route.yaml
route.route.openshift.io/etherpad created
```

```
[lab@utility etherpad]$ oc create -f etherpad-deployment.yaml
deployment.apps/etherpad created
```

- 4.5. Verify the creation of the resources:

```
[lab@utility etherpad]$ oc get all
NAME                                     READY   STATUS    RESTARTS   AGE
pod/etherpad-c7476d8d8-8b8j5      1/1     Running   0          25s

NAME           TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/etherpad   ClusterIP   172.30.15.113 <none>       9001/TCP   31s

NAME           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/etherpad   1/1     1           0          25s

NAME           DESIRED  CURRENT   READY   AGE
replicaset.apps/etherpad-c7476d8d8  1        1         1      25s

NAME          HOST/PORT   PATH
SERVICES     PORT   TERMINATION   WILDCARD
```

```
route.route.openshift.io/etherpad etherpad-etherpad.apps.ocp4.example.com
  etherpad http edge/Redirect None

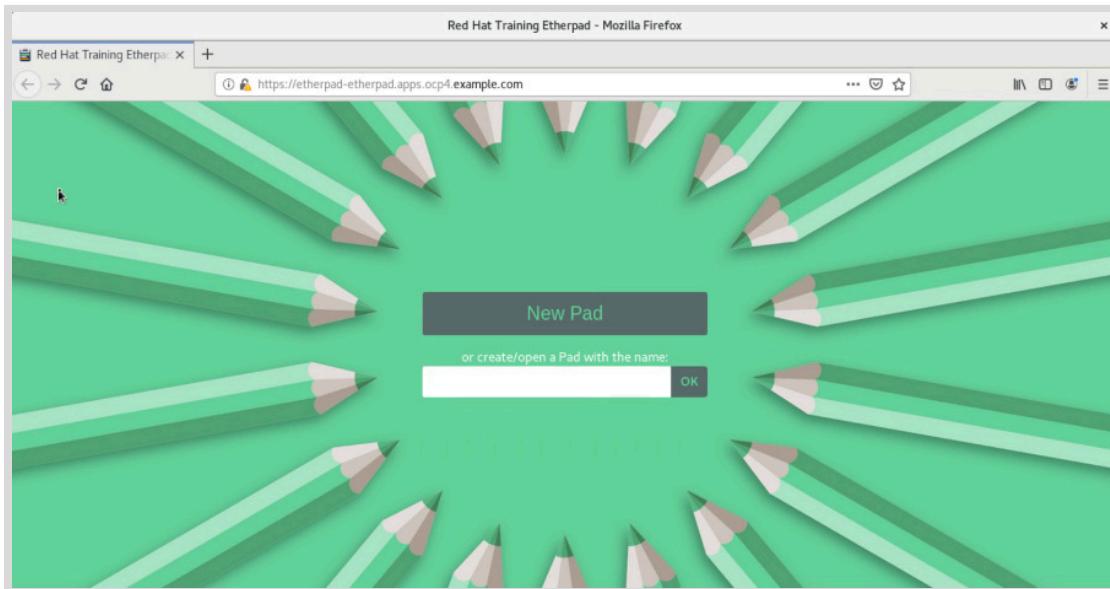
[lab@utility etherpad]$ oc get events -o template --template \
> '{{range .items}}{{.message}}\n{{end}}'
...output omitted...
Created pod: etherpad-c7476d8d8-pnv4r
waiting for a volume to be created, either by external provisioner "nfs-dynamic-provisioner" or manually created by system administrator
External provisioner is provisioning volume for claim "etherpad/etherpad"
Successfully provisioned volume pvc-17d90bb5-ae24-46e5-a264-4e057383cdd4
...output omitted...
```

The PV was automatically created by the nfs-dynamic-provisioner.

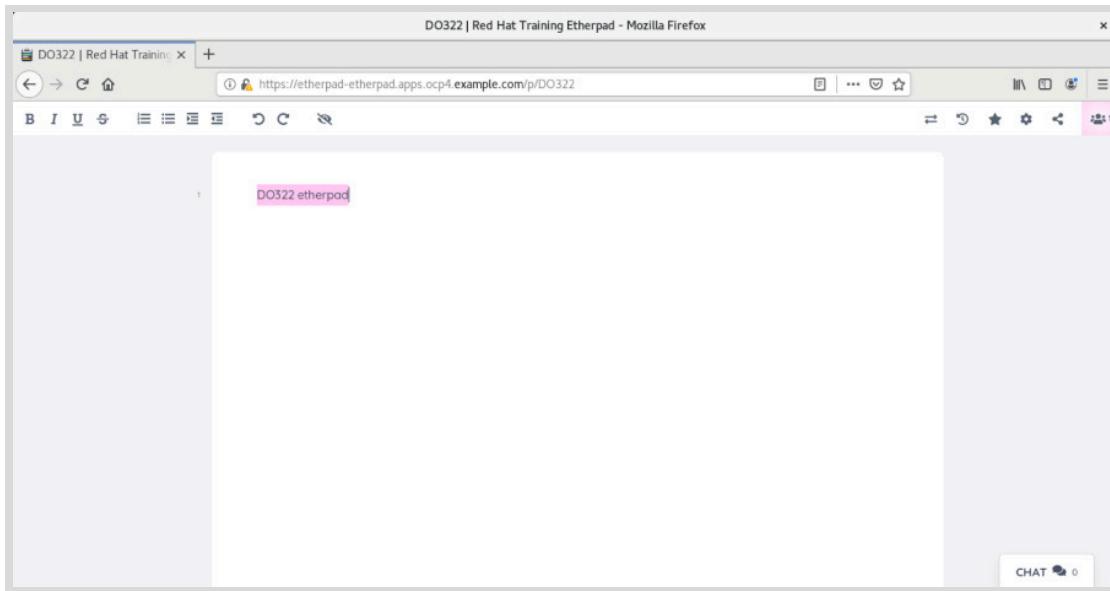
- 4.6. Open the Firefox web browser on the workstation machine and navigate to the route `etherpad-etherpad.apps.ocp4.example.com`.

**Note**

When prompted by Firefox about a potential security risk, click **Advanced** and then click **Accept the Risk and Continue**.



- 4.7. Create a new etherpad named "DO322". Remove the text in the etherpad and type: "DO322 etherpad".



4.8. In the terminal, delete the etherpad pod and verify that the database is preserved.

```
[lab@utility etherpad]$ oc get pods
NAME           READY   STATUS    RESTARTS   AGE
etherpad-c7476d8d8-pnv4r   1/1     Running   0          23m
```

```
[lab@utility etherpad]$ oc delete pod etherpad-c7476d8d8-pnv4r
pod "etherpad-c7476d8d8-pnv4r" deleted
```

Verify the creation of the new pod:

```
[lab@utility etherpad]$ oc get pods
NAME           READY   STATUS    RESTARTS   AGE
etherpad-c7476d8d8-sfvsj   1/1     Running   0          26s
```

4.9. Return to the Firefox web browser. Reload the web page and verify that the text "DO322 etherpad" is still present.

► 5. Return to the lab home folder and clean up the environment:

```
[lab@utility etherpad]$ cd ~
[lab@utility ~]$ oc delete project etherpad
project.project.openshift.io "etherpad" deleted
```

Finish

Do not make any other changes to the lab environment until the next guided exercise. You will continue using it in later guided exercises.

This concludes the guided exercise.

Replacing a Control Plane Node

Objectives

- Backup and restore a control plane node.

Explaining Etcd Backups

The OpenShift etcd database is a key-value store containing the cluster status data. To restore an OpenShift cluster to a previous state, administrators first must perform an etcd backup.

An etcd backup contains the status of the cluster and the configuration of the static pods running on a control plane node. It is not necessary to save a backup from each of the three control plane nodes. The backup of one of the control plane nodes is enough.

Performing regular etcd backups is necessary to ensure that a cluster can be recovered if all control plane nodes are lost, or the cluster becomes unstable because of human error or a hardware failure.



Note

The default etcd space quota size is approximately 7.5 GB, and is defined by the variable `etcd_server_quota_backend_bytes`. If an etcd database exceeds this quota, you must defragment and compact it.

For more information, refer to the *Recommended etcd practices* section of the *Scalability and Performance Guide* at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/scalability_and_performance/index#recommended-etcd-practices_

Restoring the Cluster to a Previous State

OpenShift provides automatic healing capabilities because of the nature of Kubernetes operators. Sometimes, such as when a cluster is misconfigured or data corrupted, an OpenShift cluster might not be able to heal itself. If the cluster cannot self-heal, OpenShift enables restoring the cluster to a previous state using an etcd backup.

The restore procedure of an OpenShift cluster requires:

- SSH access to the control plane nodes
- A user with `cluster-admin` permissions
- An etcd backup

You can find more information about disaster recovery in the *Disaster recovery* chapter of the *Red Hat OpenShift Container Platform Backup and Restore Guide* at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/backup_and_restore/index#disaster-recovery

Restoring a Lost Control Plane Node

In production environments, it is possible to lose one or more control plane nodes due to human error or a hardware failure.

To replace an unhealthy control plane node or etcd member, the cluster administrators must use the same machine configuration to create the new node as was used to create the unhealthy node being replaced.

The machine configuration is stored in different formats depending on the original installation method. For Full-stack Automation OpenShift installations, retrieve the machine configuration from the OpenShift machine API. For installations on Pre-existing Infrastructure, the ignition files are required.

Elsewhere in this chapter, you will learn how to redeploy a lost control plane node using the classroom environment.



Note

In the future, Red Hat plans to include automatic handling and replacement of lost control plane members by the etcd operator.



References

For more information, refer to the *Post-Installation Cluster Tasks* chapter in the *Red Hat OpenShift Container Platform Post-Installation Configuration Guide* at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/post-installation_configuration/

For more information, you can also refer to the *Red Hat OpenShift Container Platform Backup and Restore Guide* at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/backup_and_restore/

► Guided Exercise

Replacing a Control Plane Node

- Reinstall a failed control plane node.

Outcomes

You should be able to:

- Generate an etcd backup.
- Recover a failed control plane node.

Before You Begin

To perform this exercise, ensure that you have completed all prior exercises in this course.

The classroom environment is unchanged, apart from the course exercises.

If you have made any additional changes to the environment, recreate a new classroom environment before you begin. You must repeat the steps from the guided exercises in the previous chapters to prepare for this exercise.

Instructions

- 1. Use the kubeconfig file to authenticate into OpenShift from the utility server.

- 1.1. From the workstation machine, log into the utility server as the lab user:

```
[student@workstation ~]$ ssh lab@utility  
...output omitted...
```

- 1.2. Export the KUBECONFIG variable to authenticate into OpenShift:

```
[lab@utility ~]$ export KUBECONFIG=~/ocp4upi/auth/kubeconfig
```

- 2. Create the etcd backup.

- 2.1. Start a debug session in the master01 control plane node. Change the root directory to /host.

```
[lab@utility ~]$ oc get nodes  
NAME      STATUS    ROLES     AGE      VERSION  
master01   Ready     master    2d18h   v1.19.0+9f84db3  
master02   Ready     master    2d18h   v1.19.0+9f84db3  
master03   Ready     master    2d18h   v1.19.0+9f84db3  
worker01   Ready     worker   2d17h   v1.19.0+9f84db3  
worker02   Ready     worker   2d17h   v1.19.0+9f84db3
```

```
[lab@utility ~]$ oc debug node/master01
```

Chapter 6 | Completing the Installation of OpenShift Without an Infrastructure Provider

```
Creating debug namespace/openshift-debug-node-sws5d ...
Starting pod/master01-debug ...
To use host binaries, run `chroot /host`
Pod IP: 192.168.50.10
If you don't see a command prompt, try pressing enter.
sh-4.4# chroot /host
sh-4.4#
```

- 2.2. Run the etcd backup script in `/usr/local/bin/cluster-backup.sh`. Save the output in `/home/core/backup`.

```
sh-4.4# /usr/local/bin/cluster-backup.sh /home/core/backup
...output omitted...
snapshot db and kube resources are successfully saved to /home/core/backup
```

- 2.3. List the resources created by the backup script.

```
sh-4.4# ls /home/core/backup/
snapshot_2020-12-03_110352.db  static_kuberresources_2020-12-03_110352.tar.gz
```

- The `snapshot_2020-12-03_110352.db` file is the etcd snapshot.
- The `static_kuberresources_2020-12-03_110352.tar.gz` contains the static pods resources: `kube-apiserver`, `kube-controller-manager`, `kube-scheduler` and `etcd`.

- 2.4. Change the backup folder permissions so that the new owner is the `core` user and the `core` group.

```
sh-4.4# chown -R core:core /home/core/backup
```

- 2.5. Close the debug session.

```
sh-4.4# exit
exit
sh-4.4# exit
exit

Removing debug pod ...
Removing debug namespace/openshift-debug-node-85dg7 ...
```

- 2.6. Create a new folder, `/home/lab/etcd_backup`, and save the backup files to the utility server. Use the ssh key `ocp4upi` that you generated as a prerequisite. Type `yes` to establish the connection to the `master01` server.

```
[lab@utility ~]$ mkdir /home/lab/etcd_backup
[lab@utility ~]$ scp -i .ssh/ocp4upi \
> core@192.168.50.10:/home/core/backup/* /home/lab/etcd_backup/
The authenticity of host '192.168.50.10 (192.168.50.10)' can't be established.
ECDSA key fingerprint is SHA256:lk205E7PCvFYMACnHH26Sy1wHowbuEN1bACvyu+WjCU.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

```
Warning: Permanently added '192.168.50.10' (ECDSA) to the list of known hosts.
snapshot_2020-12-16_105145.db          100%   73MB 181.3MB/s  00:00
static_kubereresources_2020-12-16_105145.tar.gz 100%   65KB 52.1MB/s  00:00
```

- 3. Remove a control plane node.



Note

This procedure for recovering a failed control plane member is valid when the failed machine is not running or in a `NotReady` status.

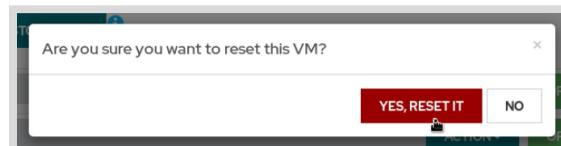
If the machine is running and the control plane member is listed as `Ready`, but the `etcd` pod is in an `Error` status, you must follow the procedure described in the *Replacing an unhealthy etcd member whose etcd pod is crashlooping* section in the *Red Hat OpenShift Container Platform Backup and Restore Guide* at https://access.redhat.com/documentation/en-us/openshift_container_platform/4.6/html-single/backup_and_restore/

- 3.1. Reset the `master01` machine in the lab environment.

From the `workstation` machine, use Firefox to navigate to the **Lab Environment** tab on the Red Hat OpenShift Installation Lab web page. Click **ACTION** for the `master01` server, and then select **Reset**, as shown in the following image:

	DELETE	STOP		
bastion	active		ACTION ▾	OPEN CONSOLE
bootstrap	active		ACTION ▾	OPEN CONSOLE
classroom	active		ACTION ▾	OPEN CONSOLE
master01	active		ACTION ▾	OPEN CONSOLE
master02	active		Start	OPEN CONSOLE
master03	active		Shutdown Power Off	OPEN CONSOLE
utility	active		Reset	OPEN CONSOLE
worker01	active		ACTION ▾	OPEN CONSOLE

When prompted to confirm the reset, click **YES, RESET IT**.



This restores the `master01` server to the initial status when the lab environment was created (a blank disk, ready for PXE boot).

- 3.2. From the `utility` server, verify the status of `master01` server and the `etcd` cluster. Wait until the `master01` server displays `NotReady`. Press **Ctrl+C** to exit.

```
[lab@utility ~]$ oc get nodes -w
NAME      STATUS    ROLES     AGE      VERSION
master01  NotReady master    73m     v1.19.0+9f84db3
master02  Ready     master    73m     v1.19.0+9f84db3
master03  Ready     master    73m     v1.19.0+9f84db3
worker01  Ready     worker   47m     v1.19.0+9f84db3
worker02  Ready     worker   47m     v1.19.0+9f84db3
```

```
[lab@utility ~]$ oc get etcd -o=jsonpath='{range .items[0].status.conditions[?(@.type=="EtcdMembersAvailable")]}{.message}{"\n"}'
2 of 3 members are available, master01 is unhealthy
```

```
[lab@utility ~]$ oc get nodes -o jsonpath='{range .items[*]}{"\n"}{.metadata.name}
{"\t"}{range .spec.taints[*]}{.key}" " "
master01 node-role.kubernetes.io/master node.kubernetes.io/unreachable
node.kubernetes.io/unreachable
```

The node **master01** is **NotReady** and has the taint **node.kubernetes.io/unreachable**.



Note

The pods running on the **master01** server might show the status **Running** for some time after the server connection is lost. This is because the kubelet running on **master01** was unable to report the **NotReady** status.



Note

Because you deleted a member of the control plane, you might encounter temporary errors in pods or services of the cluster. If so, ignore the error messages and retry your commands to complete the exercise.

► 4. Remove the unhealthy etcd member.

4.1. From a running etcd pod, remove the missing etcd member.

```
[lab@utility ~]$ oc get pods -n openshift-etcd | grep etcd
etcd-master01                      3/3     Running   0          89m
etcd-master02                      3/3     Running   0          88m
etcd-master03                      3/3     Running   0          90m
etcd-quorum-guard-644f5747b8-bb8vw  1/1     Running   0          93m
etcd-quorum-guard-644f5747b8-c9b5k  1/1     Running   0          93m
etcd-quorum-guard-644f5747b8-fbc5f  1/1     Running   0          93m

[lab@utility ~]$ oc rsh -n openshift-etcd etcd-master02
Defaulting container name to etcdctl.
Use 'oc describe pod/etcd-master02 -n openshift-etcd' to see all of the containers
in this pod.
sh-4.4# etcdctl member list -w table
```

```
+-----+-----+-----+
+-----+-----+
|      ID      | STATUS | NAME   |          PEER ADDRS      |
| CLIENT ADDRS | IS LEARNER |
+-----+-----+-----+
+-----+-----+
| 6c05b85443152afa | started | master01 | https://192.168.50.10:2380 |
| https://192.168.50.10:2379 |     false |
| a8fc53b7e8e11c19 | started | master02 | https://192.168.50.11:2380 |
| https://192.168.50.11:2379 |     false |
| f955e62306188c83 | started | master03 | https://192.168.50.12:2380 |
| https://192.168.50.12:2379 |     false |
+-----+-----+-----+
+-----+-----+
```

**Warning**

The etcd members are sorted by the **ID** field, not the **NAME**. Verify that you are using the **master01** member ID, which might not be first on the list.

Next, remove the unhealthy etcd member.

```
sh-4.4# etcdctl member remove 6c05b85443152afa
Member 6c05b85443152afa removed from cluster c73435d0ce2db908

sh-4.4# etcdctl member list -w table
+-----+-----+-----+
+-----+-----+
|      ID      | STATUS | NAME   |          PEER ADDRS      |
| CLIENT ADDRS | IS LEARNER |
+-----+-----+-----+
+-----+-----+
| a8fc53b7e8e11c19 | started | master02 | https://192.168.50.11:2380 |
| https://192.168.50.11:2379 |     false |
| f955e62306188c83 | started | master03 | https://192.168.50.12:2380 |
| https://192.168.50.12:2379 |     false |
+-----+-----+-----+
+-----+-----+
sh-4.4# exit
exit
```

The etcd member **master01** is now removed.

4.2. Remove the secrets used by the removed etcd member.

```
[lab@utility ~]$ oc get secrets -n openshift-etcd | grep master01
etcd-peer-master01           kubernetes.io/tls            2      102m
etcd-serving-master01        kubernetes.io/tls            2      102m
etcd-serving-metrics-master01 kubernetes.io/tls            2      102m
```

```
[lab@utility ~]$ oc delete secret -n openshift-etcd etcd-peer-master01
secret "etcd-peer-master01" deleted
```

```
[lab@utility ~]$ oc delete secret -n openshift-etcd etcd-serving-master01
secret "etcd-serving-master01" deleted
```

```
[lab@utility ~]$ oc delete secret -n openshift-etcd etcd-serving-metrics-master01
secret "etcd-serving-metrics-master01" deleted
```

- 4.3. Delete the master01 node object.

```
[lab@utility ~]$ oc delete node master01
node "master01" deleted
```

```
[lab@utility ~]$ oc get nodes
NAME      STATUS    ROLES      AGE      VERSION
master02   Ready     master     105m    v1.19.0+9f84db3
master03   Ready     master     106m    v1.19.0+9f84db3
worker01   Ready     worker     79m     v1.19.0+9f84db3
worker02   Ready     worker     79m     v1.19.0+9f84db3
```

- 4.4. Remove the secrets used by the removed etcd member.

```
[lab@utility ~]$ oc get secrets -n openshift-etcd | grep master01
etcd-peer-master01          kubernetes.io/tls            2      102m
etcd-serving-master01        kubernetes.io/tls            2      102m
etcd-serving-metrics-master01 kubernetes.io/tls            2      102m
[lab@utility ~]$ oc delete secret -n openshift-etcd etcd-peer-master01
secret "etcd-peer-master01" deleted
[lab@utility ~]$ oc delete secret -n openshift-etcd etcd-serving-master01
secret "etcd-serving-master01" deleted
[lab@utility ~]$ oc delete secret -n openshift-etcd etcd-serving-metrics-master01
secret "etcd-serving-metrics-master01" deleted
```

- 5. From the workstation machine, use Firefox to provision the new master01 node from the **Lab Environment** tab within the Red Hat Learning portal.

- 5.1. Use the **Console** button to open the web console and install the master01 node using the PXE Boot Menu.

Table of Contents Course Lab Environment ⚡ ⓘ

▶ Lab Controls

Click **CREATE** to build all of the virtual machines needed for the classroom lab environment. This may take several minutes to complete. Once created the environment can then be stopped and restarted to pause your experience.

If you **DELETE** your lab, you will remove all of the virtual machines in your classroom and lose all of your progress.

[WATCH TUTORIAL](#)

DELETE	STOP ⓘ	ACTION ⓘ	OPEN CONSOLE ⓘ
bastion	active	ACTION ⓘ	OPEN CONSOLE ⓘ
bootstrap	active	ACTION ⓘ	OPEN CONSOLE ⓘ
classroom	active	ACTION ⓘ	OPEN CONSOLE ⓘ
master01	active	ACTION ⓘ	OPEN CONSOLE ⓘ
master02	active	ACTION ⓘ	OPEN CONSOLE ⓘ

**Note**

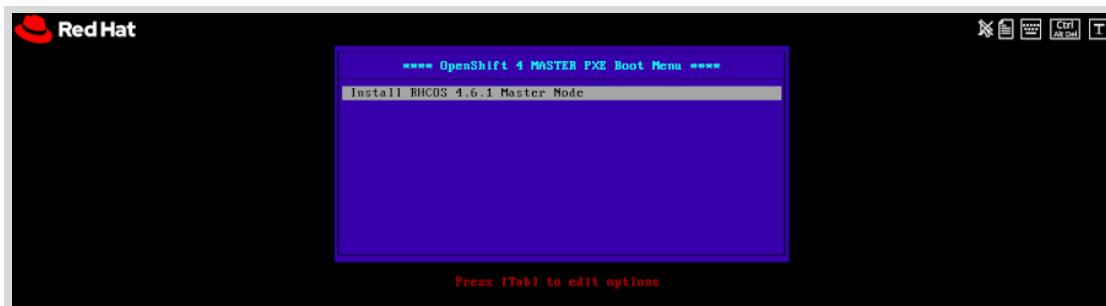
If the RHCOS installation does not start, use the **Ctrl+Alt+Del** button on the server web console to trigger a restart.

**Note**

A warning message displays in the Firefox browser. Click **Preferences** and allow Firefox to show the pop-up windows for rol.redhat.com.

Reboot the **master01** virtual machine from within the Red Hat Learning portal console by clicking **Ctrl Alt Del** in the top right corner.

After the reboot is completed, press **Enter** to start the RHCOS installation in the PXE Boot menu.



After two reboots, the new **master01** control plane node is installed.

```
Red Hat Enterprise Linux CoreOS 46.82.202010091720-0 (0otp) 4.6
SSH host key: SHA256:iv129cJ1I8uuua3rrqgCEq1M0Sj4bU0na_jtrIFY+EEs4 (ECDSA)
SSH host key: SHA256:7X53xu79MWhHqHC2H7Eu0EPkqAhb/J0+1gC+Bfyzg (ED25519)
SSH host key: SHA256:642ig0QbYoGMpkrWFZ57GeboIq5SqCPHXBJuYW9naQk (RSA)
ens3: 192.168.50.10
master01 login:
```

- 6. Use the terminal on the utility server to add the new master01 node to the cluster.

- 6.1. Approve the kubelet and the node certificates.

```
[lab@utility ~]$ oc get csr | grep Pending
csr-7qqzn  2m1s    kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-operator:node-bootstrapper
Pending
```

```
[lab@utility ~]$ oc get csr -o go-template='{{range .items}}{{if not .status}}
{{.metadata.name}}\n{{end}}{{end}}' | xargs oc adm certificate approve
certificatesigningrequest.certificates.k8s.io/csr-7qqzn approved
```

```
[lab@utility ~]$ oc get csr | grep Pending
csr-48rhz  4s      kubernetes.io/kubelet-serving
system:node:master01
Pending
```

```
[lab@utility ~]$ oc get csr -o go-template='{{range .items}}{{if not .status}}
{{.metadata.name}}\n{{end}}{{end}}' | xargs oc adm certificate approve
certificatesigningrequest.certificates.k8s.io/csr-48rhz approved
```

**Note**

There might be more than one pending CSR in the cluster. Do not worry if the output of `oc get csr | grep Pending` shows more than one entry.

- 6.2. Verify the status of the nodes in the cluster.

```
[lab@utility ~]$ oc get nodes
NAME     STATUS   ROLES     AGE      VERSION
master01  Ready    master    6m2s    v1.19.0+9f84db3
master02  Ready    master    120m    v1.19.0+9f84db3
master03  Ready    master    120m    v1.19.0+9f84db3
worker01  Ready    worker    93m     v1.19.0+9f84db3
worker02  Ready    worker    93m     v1.19.0+9f84db3
```

- 6.3. Check the status of the etcd pods in the cluster.

```
[lab@utility ~]$ oc get pods -n openshift-etcd | grep etcd
etcd-master01                      3/3     Running   0          9m7s
etcd-master02                      3/3     Running   0          7m35s
etcd-master03                      3/3     Running   0          8m37s
etcd-quorum-guard-644f5747b8-bb8vw  1/1     Running   0          121m
etcd-quorum-guard-644f5747b8-c9b5k  1/1     Running   0          121m
etcd-quorum-guard-644f5747b8-dfrcj  1/1     Running   0          13m
```

- 6.4. Verify the members of the etcd cluster from within the etcd pod in master01.

```
[lab@utility ~]$ oc rsh -n openshift-etcd etcd-master01
Defaulting container name to etcdctl.
Use 'oc describe pod/etcd-master01 -n openshift-etcd' to see all of the containers
in this pod.
sh-4.4# etcdctl member list -w table
+-----+-----+-----+
+-----+-----+
|      ID      | STATUS | NAME   |          PEER ADDRS           |
| CLIENT ADDRS | IS LEARNER |          |
+-----+-----+-----+
+-----+-----+
| a8fc53b7e8e11c19 | started | master02 | https://192.168.50.11:2380 |
| https://192.168.50.11:2379 |     false |
| f9423fa20fccb5de | started | master01 | https://192.168.50.10:2380 |
| https://192.168.50.10:2379 |     false |
| f955e62306188c83 | started | master03 | https://192.168.50.12:2380 |
| https://192.168.50.12:2379 |     false |
+-----+-----+-----+
+-----+-----+
sh-4.4# etcdctl endpoint health --cluster
https://192.168.50.11:2379 is healthy: successfully committed proposal: took =
8.418937ms
https://192.168.50.10:2379 is healthy: successfully committed proposal: took =
8.842156ms
https://192.168.50.12:2379 is healthy: successfully committed proposal: took =
8.2554ms
sh-4.4# exit
exit
```

6.5. Verify that the three etcd members are available to the OpenShift cluster.

```
[lab@utility ~]$ oc get etcd -o=jsonpath='{range .items[0].status.conditions[?(@.type=="EtcdMembersAvailable")]}{.message}{"\n"}'
3 members are available
```



Note

The etcd operator automatically detects a new etcd member and joins it into the cluster. In this scenario, you do not need to use the etcd backup.

Finish

After completing this guided exercise, delete the complete lab environment using the **DELETE** button from within the Red Hat Learning portal.

This concludes the guided exercise.

► Quiz

Chapter Review: Completing the Installation of OpenShift Without an Infrastructure Provider

Choose the correct answers to the following questions:

- ▶ 1. Which two of the following elements are necessary to replace a failed OpenShift control plane node if the etcd quorum is not lost? (Choose two).
 - a. The ignition configuration file
 - b. An etcd backup
 - c. SSH access to all the control planes in the cluster
 - d. A user with cluster-admin permissions
 - e. The developer user
- ▶ 2. Under which two of the following circumstances is it beneficial to perform etcd backups? (Choose two).
 - a. Every six months.
 - b. After the cluster is installed and released to end users.
 - c. Before changing a cluster configuration affecting the parameters of the kubelet.
 - d. Periodically, and before performing any action that may compromise OpenShift cluster stability.
 - e. After adding a new user.
- ▶ 3. For disaster recovery purposes, what is the most effective backup method to save the OpenShift cluster status and configuration?
 - a. A backup of all the disks of the control plane nodes
 - b. A backup of the etcd database
 - c. A backup of the partition hosting the static pods of the control plane nodes
 - d. A backup of the kubeconfig file
- ▶ 4. Which action does Red Hat recommend that you perform after every OpenShift installation?
 - a. Set up the cluster network policies
 - b. Add a custom certificate authority (CA)
 - c. Install the MySQL operator
 - d. Save a copy of the kubeconfig file generated by the installer

► Solution

Chapter Review: Completing the Installation of OpenShift Without an Infrastructure Provider

Choose the correct answers to the following questions:

- ▶ 1. Which two of the following elements are necessary to replace a failed OpenShift control plane node if the etcd quorum is not lost? (Choose two).
 - a. The ignition configuration file
 - b. An etcd backup
 - c. SSH access to all the control planes in the cluster
 - d. A user with `cluster-admin` permissions
 - e. The developer user

- ▶ 2. Under which two of the following circumstances is it beneficial to perform etcd backups? (Choose two).
 - a. Every six months.
 - b. After the cluster is installed and released to end users.
 - c. Before changing a cluster configuration affecting the parameters of the kubelet.
 - d. Periodically, and before performing any action that may compromise OpenShift cluster stability.
 - e. After adding a new user.

- ▶ 3. For disaster recovery purposes, what is the most effective backup method to save the OpenShift cluster status and configuration?
 - a. A backup of all the disks of the control plane nodes
 - b. A backup of the etcd database
 - c. A backup of the partition hosting the static pods of the control plane nodes
 - d. A backup of the kubeconfig file

- ▶ 4. Which action does Red Hat recommend that you perform after every OpenShift installation?
 - a. Set up the cluster network policies
 - b. Add a custom certificate authority (CA)
 - c. Install the MySQL operator
 - d. Save a copy of the kubeconfig file generated by the installer

Summary

- Typical Day 1 and Day 2 operations to perform in an OpenShift cluster.
- Day 1 operations to complete before considering a cluster ready for production.
- How to configure a dynamic storage provider.
- How to perform a functional test of the cluster.
- The importance of making regular etcd backups.
- How to generate an etcd backup.
- How to replace a failed control plane node.

Chapter 7

Comprehensive Review

Goal

Review tasks from *Red Hat OpenShift Installation Lab*

Objectives

- Review tasks from *Red Hat OpenShift Installation Lab*

Sections

- Comprehensive Review

Lab

- Installing a Compact OpenShift Cluster

Comprehensive Review

Objectives

After completing this section, you should have reviewed and refreshed the knowledge and skills that you learned in Red Hat OpenShift Installation Lab.

Reviewing Red Hat OpenShift Installation Lab

Before beginning the comprehensive review for this course, you should be comfortable with the topics covered in each chapter. Do not hesitate to ask the instructor for extra guidance or clarification on these topics.

Describing the OpenShift Installation Process

Describe and compare the full-stack automation and pre-existing infrastructure installation methods.

- Describe and compare the full-stack automation and pre-existing infrastructure installation methods.
- Describe the OpenShift installer and its configuration files.
- Describe the differences between a self-managed OpenShift cluster and hosted OpenShift offerings.

Installing OpenShift on a Cloud Provider

Provision OpenShift clusters on IaaS cloud providers, with common customizations, using the full-stack automation installation method.

- Describe the architecture and workflow for installing OpenShift on an IaaS Cloud Provider using the full-stack automation method.
- Provide the prerequisites to install OpenShift on Amazon Web Services (AWS) using full-stack automation.
- Install OpenShift on Amazon Web Services (AWS) using full-stack automation with common customizations.
- Assess the success of an installation of OpenShift on AWS.

Installing OpenShift on a Virtualized Environment

Provision OpenShift clusters on hypervisors, with common customizations, using the full-stack automation and the pre-existing infrastructure installation methods.

- Describe the architecture and workflow for installing OpenShift on hypervisors using the full-stack automation and pre-existing infrastructure methods.
- Describe the installation of OpenShift on vSphere using full-stack automation with common customizations.

- Describe the installation of OpenShift on vSphere using pre-existing infrastructure with common customizations.

Planning to Install OpenShift Without an Infrastructure Provider

Configure the prerequisites for provisioning OpenShift clusters without integration with the underlying infrastructure.

- Describe the architecture and workflow for installing OpenShift without integration with the underlying infrastructure.
- Provide the prerequisites to install OpenShift without integration with the underlying infrastructure.

Installing OpenShift Without an Infrastructure Provider

Provision OpenShift clusters without integration with the underlying infrastructure.

- Install OpenShift by starting the bootstrap machine and cluster nodes, and monitoring the progress of the installation process.

Completing the Installation of OpenShift Without an Infrastructure Provider

Perform essential tasks that are required before onboarding users and applications on a newly provisioned OpenShift cluster.

- Perform required customizations before onboarding users and applications into a newly installed cluster.
- Backup and restore a control plane node.

► Lab

Installing a Compact OpenShift Cluster

- Install a compact cluster (three control plane nodes), and verify the installation.

Outcomes

You should be able to complete the installation of a compact OpenShift cluster and verify the installation.

Before You Begin

From within the Red Hat Learning portal, click **DELETE** to delete the complete lab environment.

After the environment is deleted, click **CREATE**, and then wait until all the servers are **active**.

After all the servers become **active**, click **+** to extend the auto-stop timer and increase the interval to at least two hours. This will help ensure that your environment is not stopped in the middle of the exercise. As the **student** user on the **workstation** machine, use the **lab** command to prepare your system for this exercise.

This command ensures that the necessary files to complete the comprehensive review are available on the workstation machine.

```
[student@workstation ~]$ lab comprehensive-review start
```



Note

The Firefox version in the workstation machine does not render properly the <https://cloud.redhat.com/openshift/> site. To complete this guided exercise, install and use the Chromium web browser in the workstation:

```
[student@workstation ~]$ sudo yum install chromium
```

Instructions

- Find the local registry FQDN for the region of your classroom environment. You can locate your lab environment region on the **Lab Environment** tab on the Red Hat OpenShift Installation Lab course web page.

The following table shows the FQDN of the local registry for each region.

Local Registry FQDN per Classroom Environment Region

Region	Local registry FQDN
northamerica	nexus-registry-int.apps.tools-na150.prod.ole.redhat.com
emea	nexus-registry-int.apps.tools-emea150.prod.ole.redhat.com
apac	nexus-registry-int.apps.tools-apac150.prod.ole.redhat.com

During this exercise, you will use a local registry containing the OpenShift release images mirrored from `quay.io`. The local registry runs in the same region as the classroom environment, and its fully qualified domain name (FQDN) depends on that region.

2. Create a valid pull secret for installing OpenShift in the classroom environment using a pull secret from `cloud.redhat.com`. The credentials for the `quay.io` registry must be replaced with the credentials for the local registry.

Assuming that your classroom environment region is `northamerica`, the content of the `pull-secret` for the local registry is:

```
"nexus-registry-int.apps.tools-na150.prod.ole.redhat.com":  
{"auth":"cmVndXNlcjpJbnN0YWxsTTM=", "email":"nobody@example.com"}
```

Finally, copy the pull secret file to the **utility** server as the **lab** user.

3. Run the Ansible Playbook in the folder `/home/student/D0322/labs/comprehensive-review/ansible/`, using the inventory file included in this folder.
4. Create a SSH key on the **utility** server. Save it to the `/home/lab/.ssh/ocp4upi` file.
5. Complete the `install-config.yaml` file in the `/home/lab/ocp4upi/` folder. Use the pull secret generated elsewhere in this exercise. Use the SSH key generated elsewhere in this exercise. Include the `imageContentSources` information to use the local registry to retrieve the OpenShift release images. There must be only three control plane nodes without any compute nodes.
6. Create the Kubernetes manifests and ignition configuration files using the `openshift-install` CLI tool. You must configure the installer to use the classroom local registry to retrieve the OpenShift release version. To do so, you must export the following variable:

```
OPENSHIFT_INSTALL_RELEASE_IMAGE_OVERRIDE=<LOCAL_REGISTRY_FQDN>/openshift/  
ocp4:4.6.4-x86_64"
```

Place the ignition files in the `/var/www/html/openshift4/4.6.4/ignitions/` with world-readable permissions.

7. From the PXE menu, install RHCOS in the `bootstrap`, `master01`, `master02`, and `master03` machines to start the OpenShift cluster deployment.
8. Configure the persistent storage for the local registry using the NFS server running on the **utility** server.

Use a new folder named `registry` within the NFS `/exports` folder on the **utility** server. The folder permissions must be `777`.

9. Verify the cluster health.

Chapter 7 | Comprehensive Review

10. Deploy a `rails-postgresql-example` application in the `test` project to perform a functional test of the cluster.

Evaluation

As the student user on the `workstation` machine, use the `lab` command to grade your work. Correct any reported failures and rerun the command until successful.

```
[lab@utility ~]$ exit  
...output omitted...  
[student@workstation ~]$ lab comprehensive-review grade
```

Finish

As the student user on the `workstation` machine, use the `lab` command to complete this exercise.

```
[student@workstation ~]$ lab comprehensive-review finish
```

**Note**

The `lab comprehensive-review finish` command does not destroy the cluster. Running the `lab comprehensive-review grade` script after the `lab comprehensive-review finish` command shows that the three-node cluster exists.

If you wish to start over, reset the `bootstrap`, `master01`, `master02`, and `master03` machines, and then start again at the step that instructs you to install RHCOS in the `bootstrap`, `master01`, `master02`, and `master03` machines from the PXE menu.

This concludes the lab.

► Solution

Installing a Compact OpenShift Cluster

- Install a compact cluster (three control plane nodes), and verify the installation.

Outcomes

You should be able to complete the installation of a compact OpenShift cluster and verify the installation.

Before You Begin

From within the Red Hat Learning portal, click **DELETE** to delete the complete lab environment.

After the environment is deleted, click **CREATE**, and then wait until all the servers are **active**.

After all the servers become **active**, click **+** to extend the auto-stop timer and increase the interval to at least two hours. This will help ensure that your environment is not stopped in the middle of the exercise. As the **student** user on the **workstation** machine, use the **lab** command to prepare your system for this exercise.

This command ensures that the necessary files to complete the comprehensive review are available on the workstation machine.

```
[student@workstation ~]$ lab comprehensive-review start
```



Note

The Firefox version in the workstation machine does not render properly the <https://cloud.redhat.com/openshift/> site. To complete this guided exercise, install and use the Chromium web browser in the workstation:

```
[student@workstation ~]$ sudo yum install chromium
```

Instructions

1. Find the local registry FQDN for the region of your classroom environment. You can locate your lab environment region on the **Lab Environment** tab on the Red Hat OpenShift Installation Lab course web page.

The following table shows the FQDN of the local registry for each region.

Local Registry FQDN per Classroom Environment Region

Region	Local registry FQDN
northamerica	nexus-registry-int.apps.tools-na150.prod.ole.redhat.com
emea	nexus-registry-int.apps.tools-emea150.prod.ole.redhat.com
apac	nexus-registry-int.apps.tools-apac150.prod.ole.redhat.com

During this exercise, you will use a local registry containing the OpenShift release images mirrored from `quay.io`. The local registry runs in the same region as the classroom environment, and its fully qualified domain name (FQDN) depends on that region.

- 1.1. Install the Chromium web browser from a terminal on the **workstation** machine.

```
[student@workstation ~]$ sudo yum install chromium
...output omitted...
```

When prompted, type yes and press **Enter** to accept the installation.

- 1.2. Find the region of your classroom environment.

From the **workstation** machine, use the Chromium web browser to navigate to the **Lab Environment** tab on the Red Hat OpenShift Installation Lab course web page.

Click **information** to display information about the classroom environment. You can find the classroom environment region in the **Published region** field.

The screenshot shows the 'Lab Environment' tab of the Red Hat OpenShift Installation Lab course. It lists three labs: 'bastion' (stopped), 'bootstrap' (stopped), and 'classroom' (stopped). A tooltip is shown over the 'information' icon for the 'classroom' lab, displaying the following details:

- Project id: 2ebdcfcb42a144ada7e8cfc3fcfb8b56
- Project name: ole-3030f6f7-9253-47ed-9ea1-73c0aa7e62c9
- Project state: stopped
- Lab definition id: do322ea-46
- Published region: northamerica (highlighted)
- openstack region: us-east-1

- 1.3. Locate the FQDN of the local registry for your region using the preceding table.

- 1.4. As the **lab** user on the **utility** server, use the `curl` command to verify that you can communicate with the FQDN of the local registry for your classroom environment.

The following command assumes that the classroom environment region is `northamerica`, with the local registry FQDN `nexus-registry-int.apps.tools-na150.prod.ole.redhat.com`.

```
[student@workstation ~]$ ssh lab@utility
```

```
[lab@utility ~]$ curl -s \  
> https://nexus-registry-int.apps.tools-na150.prod.ole.redhat.com -o /dev/null;  
echo $?  
0
```

If you have used the wrong local registry FQDN, the output of the `curl` command will not be 0.



Note

The local registry of the `northamerica` region `nexus-registry-int.apps.tools-na150.prod.ole.redhat.com` is used in the following steps.

Ensure that you use the local registry FQDN of the region of your classroom environment. Using a local registry from other regions will cause the failure of the guided exercises.

2. Create a valid pull secret for installing OpenShift in the classroom environment using a pull secret from `cloud.redhat.com`. The credentials for the `quay.io` registry must be replaced with the credentials for the local registry.

Assuming that your classroom environment region is `northamerica`, the content of the `pull-secret` for the local registry is:

```
"nexus-registry-int.apps.tools-na150.prod.ole.redhat.com":  
{ "auth": "cmVndXNlcjpJbnN0YWxsTTM=", "email": "nobody@example.com"}
```

Finally, copy the pull secret file to the `utility` server as the `lab` user.

- 2.1. Obtain a pull secret from `cloud.redhat.com`.

On the `workstation` machine, use the Chromium web browser to navigate to `https://cloud.redhat.com/openshift/install/metal/user-provisioned`. Log in using your Red Hat account credentials.

Click **Download pull secret**. Then select **Save File** and click **OK**.

The file is saved as `/home/student/Downloads/pull-secret`.

- 2.2. Format the pull secret as a JSON file.

Open a terminal on the `workstation` machine and format the `pull-secret` file in the `/home/student/Downloads` folder as a JSON.

```
[lab@utility ~]$ exit  
...output omitted...  
[student@workstation ~]$
```

```
[student@workstation ~]$ python3 -m json.tool \  
> Downloads/pull-secret > pull-secret.json
```

```
[student@workstation ~]$ cat pull-secret.json  
{  
    "auths": {  
        "cloud.openshift.com": {
```

Chapter 7 | Comprehensive Review

```

    "auth":  

    "UxUUjYwSTMyb3BlUxUUjYwSTMybnNUxUUjYwSTMyoawZUxUU...YMy3NfNGUXUUjYw==",  

        "email": "student@redhat.com"  

    },  

    "quay.io": {  

    "auth": "UxUUjYwSTMyb3BlUxUUjYwSTMybnNUxUUjYwSTMyoawZUxUU...YMy3NfNGUXUUjYw==",  

        "email": "student@redhat.com"  

    },  

    "registry.connect.redhat.com": {  

        "auth":  

    "CvmsWaJUROSkhCkEQ71NiM1BsracE9ZOVmBMIDrs3R20K0H8Eq...Tx09phtozeXpqKLJN=",  

        "email": "student@redhat.com"  

    },  

    "registry.redhat.io": {  

        "auth":  

    "CvmsWaJUROSkhCkEQ71NiM1BsracE9ZOVmBMIDrs3R20K0H8Eq...Tx09phtozeXpqKLJN=",  

        "email": "student@redhat.com"  

    }  

}
}

```

- 2.3. Edit the `pull-secret.json` file to replace the credentials for `quay.io` with the credentials for your local registry.

```

[student@workstation ~]$ vi pull-secret.json
{
    "auths": {
        "cloud.openshift.com": {
            "auth":  

    "UxUUjYwSTMyb3BlUxUUjYwSTMybnNUxUUjYwSTMyoawZUxUU...YMy3NfNGUXUUjYw==",  

            "email": "student@redhat.com"  

        },  

        "nexus-registry-int.apps.tools-na150.prod.ole.redhat.com": {
            "auth": "cmVndXNlcjpJbnN0YWxsTTM=",  

            "email": "nobody@example.com"  

        },  

        "registry.connect.redhat.com": {
            "auth":  

    "CvmsWaJUROSkhCkEQ71NiM1BsracE9ZOVmBMIDrs3R20K0H8Eq...Tx09phtozeXpqKLJN=",  

            "email": "student@redhat.com"  

        },  

        "registry.redhat.io": {
            "auth":  

    "CvmsWaJUROSkhCkEQ71NiM1BsracE9ZOVmBMIDrs3R20K0H8Eq...Tx09phtozeXpqKLJN=",  

            "email": "student@redhat.com"  

        }
    }
}

```

Save the file and close the editor.

- 2.4. Use the `jq` tool to generate the compact version of the `pull-secret.json` file in the `pull-secret-one-line.json` file.

```
[student@workstation ~]$ cat pull-secret.json | jq . -c > pull-secret-oneline.json
```

**Note**

The jq parsing used in the preceding command also checks that the `pull-secret.json` file is valid. If you do not get any errors when running this command, it means that the `pull-secret.json` and `pull-secret-oneline.json` files are valid.

```
[student@workstation ~]$ cat pull-secret-oneline.json
{
  "auths": {
    "cloud.openshift.com": {
      "auth": "UxUUj...UUjYw==",
      "email": "student@redhat.com",
      "nexus-registry-int.apps.tools-na150.prod.ole.redhat.com": {
        "auth": "cmVnd...sTTM=",
        "email": "nobody@example.com",
        "registry.connect.redhat.com": {
          "auth": "CvmsW...qKLJN=",
          "email": "student@redhat.com",
          "registry.redhat.io": {
            "auth": "CvmsW...qKLJN=",
            "email": "student@redhat.com"
          }
        }
      }
    }
  }
}
```

2.5. Copy the `pull-secret-oneline.json` file to the utility server.

```
[student@workstation ~]$ scp pull-secret-oneline.json lab@utility:
...output omitted...
pull-secret-oneline.json      100% 2624     4.6MB/s   00:00
```

You will use the `pull-secret-oneline.json` file later on this lab to complete the `install-config.yaml` file.

3. Run the Ansible Playbook in the folder `/home/student/D0322/labs/comprehensive-review/ansible/`, using the inventory file included in this folder.

3.1. Inspect the content of the folder `/home/student/D0322/`.

```
[student@workstation ~]$ tree D0322/
D0322/
└── labs
    └── comprehensive-review
        └── ansible
            ├── inventory
            ├── prereq.yaml
            └── roles
                └── ocp_install_prereq
                    ├── files
                    │   ├── haproxy.cfg
                    │   └── install-config.yaml
                    ├── tasks
                    │   └── main.yaml
                    └── vars
                        └── main.yml
```

3.2. Change to the `/home/student/D0322/labs/comprehensive-review/ansible/` folder.

```
[student@workstation ~]$ cd DO322/labs/comprehensive-review/ansible/  
[student@workstation ansible]$
```

- 3.3. Run the `prereq.yaml` Ansible Playbook. Use the inventory file included in the same folder.

```
[student@workstation ansible]$ ansible-playbook -i inventory prereq.yaml  
  
PLAY [Configure Prerequisites] ****  
  
TASK [ocp_install_prereq : Download "oc"] ****  
changed: [utility.lab.example.com]  
  
TASK [ocp_install_prereq : Download "openshift-install"] ****  
changed: [utility.lab.example.com]  
  
TASK [ocp_install_prereq : Extract "oc" to /usr/bin/] ****  
changed: [utility.lab.example.com]  
  
TASK [ocp_install_prereq : Extract "openshift-install" to /usr/bin/] ****  
changed: [utility.lab.example.com]  
  
TASK [ocp_install_prereq : Download RHCOS rootfs] ****  
changed: [utility.lab.example.com]  
  
TASK [ocp_install_prereq : Download RHCOS kernel] ****  
changed: [utility.lab.example.com]  
  
TASK [ocp_install_prereq : Download RHCOS initramfs] ****  
changed: [utility.lab.example.com]  
  
TASK [ocp_install_prereq : Download bootstrap PXE Boot file] ****  
changed: [utility.lab.example.com]  
  
TASK [ocp_install_prereq : Download master01 PXE boot file] ****  
changed: [utility.lab.example.com]  
  
TASK [ocp_install_prereq : Download master02 PXE Boot file] ****  
changed: [utility.lab.example.com]  
  
TASK [ocp_install_prereq : Download master03 PXE Boot file] ****  
changed: [utility.lab.example.com]  
  
TASK [ocp_install_prereq : Configure haproxy.cfg file] ****  
changed: [utility.lab.example.com]  
  
TASK [ocp_install_prereq : Reload HAProxy service] ****  
changed: [utility.lab.example.com]  
  
TASK [ocp_install_prereq : Create the OpenShift installation directory if it does  
not exist] *****  
changed: [utility.lab.example.com]
```

```
TASK [ocp_install_prereq : Create custom install-config.yaml file] ****
changed: [utility.lab.example.com]

PLAY RECAP ****
utility.lab.example.com    : ok=15    changed=15    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

As you can see, this playbook performs several prerequisite tasks in `utility.lab.example.com`.

- Downloads and installs the `oc` and `openshift-install` CLI tools
- Downloads the RHCOS files
- Downloads the PXE Boot files for the bootstrap node and the control plane nodes
- Configures the HAProxy Load Balancer for the API and Ingress traffic
- Downloads a base `install-config.yaml` file (that you will complete later)

3.4. Return to the home folder belonging to the `student` user.

```
[student@workstation ansible]$ cd ~
[student@workstation ~]$
```

4. Create a SSH key on the `utility` server. Save it to the `/home/lab/.ssh/ocp4upi` file.

4.1. Connect to the `utility` server via SSH as the `lab` user.

```
[student@workstation ~]$ ssh lab@utility
...output omitted...
[lab@utility ~]$
```

4.2. Generate the SSH key, and then save it in the `/home/lab/.ssh/ocp4upi` file.

```
[lab@utility ~]$ ssh-keygen -t rsa -b 4096 -N '' -f .ssh/ocp4upi
Generating public/private rsa key pair.
Your identification has been saved in .ssh/ocp4upi.
Your public key has been saved in .ssh/ocp4upi.pub.
...output omitted...
[lab@utility ~]$
```

The public key, which you will use later, is in the file `/home/lab/.ssh/ocp4upi.pub`.

5. Complete the `install-config.yaml` file in the `/home/lab/ocp4upi/` folder. Use the pull secret generated elsewhere in this exercise. Use the SSH key generated elsewhere in this exercise. Include the `imageContentSources` information to use the local registry to retrieve the OpenShift release images. There must be only three control plane nodes without any compute nodes.

5.1. Edit the `install-config.yaml` file in the `/home/lab/ocp4upi/` folder. Use the compact pull secret in the `pull-secret-oneline.json` file.

```
[lab@utility ~]$ vi ocp4upi/install-config.yaml
apiVersion: v1
baseDomain: example.com
```

```

compute:
- hyperthreading: Enabled
  name: worker
  replicas: 0
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
metadata:
  name: ocp4
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  networkType: OpenShiftSDN
  serviceNetwork:
    - 172.30.0.0/16
platform:
  none: {}
fips: false
pullSecret: |
  {"auths":{"cloud.openshift.com": {
    "auth":"UxUUj...UUjYw==",
    "email":"student@redhat.com"}, "nexus-registry-int.apps.tools-na150.prod.ole.redhat.com": {
    "auth":"cmVnd...STTM=",
    "email":"nobody@example.com"}, "registry.connect.redhat.com": {
    "auth":"CvmsW...qKLJN=",
    "email":"student@redhat.com"}, "registry.redhat.io": {
    "auth":"CvmsW...qKLJN=",
    "email":"student@redhat.com"}}}
  sshKey: |
    <CHANGE_ME_KEEPING_THE_INDENTATION_LEVEL>
imageContentSources:
- mirrors:
  - nexus-registry-int.apps.tools-na150.prod.ole.redhat.com/openshift/ocp4
    source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - nexus-registry-int.apps.tools-na150.prod.ole.redhat.com/openshift/ocp4
    source: quay.io/openshift-release-dev/ocp-v4.0-art-dev

```

Save the file and close the editor.



Note

The number of replicas of the compute machines is 0. This makes the control plane nodes schedulable automatically.

- 5.2. Copy the public SSH key from the /home/lab/.ssh/ocp4upi.pub file. Edit the install-config.yaml file again to include the key in the sshKey section.

```
[lab@utility ~]$ cat .ssh/ocp4upi.pub
ssh-rsa AAAA...UgUsz2w== lab@utility.lab.example.com
```

```
[lab@utility ~]$ vi ocp4upi/install-config.yaml
apiVersion: v1
baseDomain: example.com
```

```
compute:
  - hyperthreading: Enabled
    name: worker
    replicas: 0
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
metadata:
  name: ocp4
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  networkType: OpenShiftSDN
  serviceNetwork:
    - 172.30.0.0/16
platform:
  none: {}
fips: false
pullSecret: |
  {"auths":{"cloud.openshift.com":
  {"auth":"UxUUj...UjYw==","email":"student@redhat.com"},"nexus-
  registry-int.apps.tools-na150.prod.ole.redhat.com":
  {"auth":"cmVnd...STTM=","email":"nobody@example.com"},"registry.connect.redhat.com":
  {"auth":"CvmsW...qKLJN=","email":"student@redhat.com"},"registry.redhat.io":
  {"auth":"CvmsW...qKLJN=","email":"student@redhat.com"}}}
sshKey: |
  ssh-rsa AAAA...UgUsz2w== lab@utility.lab.example.com
imageContentSources:
  - mirrors:
    - nexus-registry-int.apps.tools-na150.prod.ole.redhat.com/openshift/ocp4
      source: quay.io/openshift-release-dev/ocp-release
  - mirrors:
    - nexus-registry-int.apps.tools-na150.prod.ole.redhat.com/openshift/ocp4
      source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

Save the file and close the editor.

- 5.3. Make a copy of the install-config.yaml file in the /home/lab/ folder for troubleshooting purposes.

```
[lab@utility ~]$ cp ocp4upi/install-config.yaml .
```

6. Create the Kubernetes manifests and ignition configuration files using the openshift-install CLI tool. You must configure the installer to use the classroom local registry to retrieve the OpenShift release version. To do so, you must export the following variable:

```
OPENSHIFT_INSTALL_RELEASE_IMAGE_OVERRIDE=<LOCAL_REGISTRY_FQDN>/openshift/
ocp4:4.6.4-x86_64"
```

Place the ignition files in the /var/www/html/openshift4/4.6.4/ignitions/ with world-readable permissions.

Chapter 7 | Comprehensive Review

- 6.1. Set the OPENSHIFT_INSTALL_RELEASE_IMAGE_OVERRIDE environment variable to use the local registry for your cluster region. The release image is in the path: openshift/ocp4:4.6.4-x86_64.

```
[lab@utility ~]$ reg="nexus-registry-int.apps.tools-na150.prod.ole.redhat.com"
[lab@utility ~]$ releaseimg="/openshift/ocp4:4.6.4-x86_64"
[lab@utility ~]$ export OPENSHIFT_INSTALL_RELEASE_IMAGE_OVERRIDE=$reg$releaseimg
```

Verify the complete URL path to the release image as follows:

```
[lab@utility ~]$ echo $OPENSHIFT_INSTALL_RELEASE_IMAGE_OVERRIDE
nexus-registry-int.apps.tools-na150.prod.ole.redhat.com/openshift/ocp4:4.6.4-
x86_64
```

**Note**

The commands above assume that your cluster is in the northamerica region. Use the FQDN of the local registry for your region.

- 6.2. Generate the Kubernetes manifests.

```
[lab@utility ~]$ openshift-install create manifests --dir=./ocp4upi
INFO Consuming Install Config from target directory
WARNING Making control-plane schedulable by setting MastersSchedulable to true for
Scheduler cluster settings
INFO Manifests created in: ocp4upi/manifests and ocp4upi/openshift
```

- 6.3. Generate the ignition configuration files.

```
[lab@utility ~]$ openshift-install create ignition-configs --dir=./ocp4upi
WARNING Found override for release image. Please be warned, this is not advised
INFO Consuming Master Machines from target directory
INFO Consuming Common Manifests from target directory
INFO Consuming OpenShift Install (Manifests) from target directory
INFO Consuming Worker Machines from target directory
INFO Consuming Openshift Manifests from target directory
INFO Ignition-Configs created in: ocp4upi and ocp4upi/auth
```

- 6.4. Copy the ignition files to the /var/www/html/openshift4/4.6.4/ignitions/ folder. Set the files permissions readable by anyone.

```
[lab@utility ~]$ sudo cp ./ocp4upi/*.ign /var/www/html/openshift4/4.6.4/ignitions/
[lab@utility ~]$ sudo chmod +r /var/www/html/openshift4/4.6.4/ignitions/*.ign
```

Verify the read permissions of the ignition config files.

```
[lab@utility ~]$ ls -l /var/www/html/openshift4/4.6.4/ignitions/
total 296
-rw-r--r--. 1 root root 293214 Feb 12 05:37 bootstrap.ign
-rw-r--r--. 1 root root 1718 Feb 12 05:37 master.ign
-rw-r--r--. 1 root root 1718 Feb 12 05:37 worker.ign
```

7. From the PXE menu, install RHCOS in the `bootstrap`, `master01`, `master02`, and `master03` machines to start the OpenShift cluster deployment.
 - 7.1. Reboot the `bootstrap` machine from within the Red Hat Learning portal console by clicking **Ctrl Alt Del** in the top right corner.
 - 7.2. Wait until the reboot is completed and the machine boots into the PXE menu.
 - 7.3. From the PXE menu, press **Enter** to proceed with the installation of Red Hat Enterprise Linux CoreOS on the machine.
 - 7.4. Repeat these steps to install `master01`, `master02`, and `master03`.
 - 7.5. For information purposes, run the `openshift-install` command to wait for the `bootstrap` installation.

```
[lab@utility ~]$ openshift-install --dir=../ocp4upi wait-for bootstrap-complete
INFO Waiting up to 20m0s for the Kubernetes API at https://
api.ocp4.example.com:6443...
INFO API v1.19.0+9f84db3 up
```



Note

As soon as you run the previous step, continue with the next steps in the lab. The previous command will take approximately 10 minutes to finish.

- 7.6. Open a new terminal on the `workstation` machine and connect via SSH to the `utility` server as the `lab` user.

```
[student@workstation ~]$ ssh lab@utility
...output omitted...
[lab@utility ~]$
```

- 7.7. Look at the output of the `openshift-install` command, and wait until the API is running.

Then, configure the `KUBECONFIG` environment variable to interact with the cluster.

```
[lab@utility ~]$ export KUBECONFIG=~/ocp4upi/auth/kubeconfig
```

- 7.8. Watch the creation and configuration of the control plane nodes using the `watch oc get nodes` command.



Note

It can take a few minutes to create the control plane nodes.

```
[lab@utility ~]$ watch oc get nodes
```

The status in the output is updated every two seconds by default.

When you first run the `watch` command, it will not show any nodes.

After a few minutes, the output will display the following:

```
Every 2.0s: oc get nodes
utility.lab.example.com: Wed Feb  3 08:10:51 2021

NAME      STATUS    ROLES      AGE     VERSION
master01  NotReady master      5s      v1.19.0+9f84db3
master02  NotReady master      12s     v1.19.0+9f84db3
master03  NotReady master      13s     v1.19.0+9f84db3
```

The control plane nodes have been created but are not ready yet.

After a few more minutes, the output will display the updated status.

```
Every 2.0s: oc get nodes
utility.lab.example.com: Wed Feb  3 08:11:33 2021

NAME      STATUS    ROLES      AGE     VERSION
master01  Ready     master     47s     v1.19.0+9f84db3
master02  Ready     master     54s     v1.19.0+9f84db3
master03  Ready     master     55s     v1.19.0+9f84db3
```

The control plane nodes are not yet configured as "schedulable". Eventually, the output will show:

```
Every 2.0s: oc get nodes
utility.lab.example.com: Wed Feb  3 08:11:59 2021

NAME      STATUS    ROLES          AGE     VERSION
master01  Ready     master,worker 73s     v1.19.0+9f84db3
master02  Ready     master,worker 80s     v1.19.0+9f84db3
master03  Ready     master,worker 81s     v1.19.0+9f84db3
```

At this point, the control plane nodes are properly configured as "schedulable"



Note

The control plane nodes CSRs are approved automatically during the installation.

- 7.9. After the control plane nodes are Ready and schedulable, the `openshift-install` command displays the following output, and then exits.

```
INFO Waiting up to 20m0s for the Kubernetes API at https://
api.ocp4.example.com:6443...
INFO API v1.19.0+9f84db3 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
INFO Time elapsed: 8m1s
```

Use the `openshift-install` command to monitor the remainder of the installation.

```
[lab@utility ~]$ openshift-install --dir=./ocp4upi \
> wait-for install-complete --log-level=debug
DEBUG OpenShift Installer 4.6.4
DEBUG Built from commit 6e02d049701437fa81521fe981405745a62c86c5
```

```
DEBUG Loading Install Config...
DEBUG Loading SSH Key...
DEBUG Loading Base Domain...
DEBUG Loading Platform...
DEBUG Loading Cluster Name...
DEBUG Loading Base Domain...
DEBUG Loading Platform...
DEBUG Loading Pull Secret...
DEBUG Loading Platform...
DEBUG Using Install Config loaded from state file
INFO Waiting up to 40m0s for the cluster at https://api.ocp4.example.com:6443 to
initialize...
DEBUG Still waiting for the cluster to initialize: Working towards 4.6.4: 99%
complete
DEBUG Still waiting for the cluster to initialize: Multiple errors are preventing
progress:
* Cluster operator authentication is reporting a failure:
WellKnownReadyControllerDegraded: kube-apiserver oauth endpoint
https://192.168.50.12:6443/.well-known/oauth-authorization-server is not yet
served and authentication operator keeps waiting (check kube-apiserver operator,
and check that instances roll out successfully, which can take several minutes
per instance)
* Cluster operator monitoring is reporting a failure: Failed to rollout the stack.
Error: running task Updating Prometheus-k8s failed: reconciling Prometheus rules
PrometheusRule failed: updating PrometheusRule object failed: Internal error
occurred: failed calling webhook "prometheusrules.openshift.io": Post "https://
prometheus-operator.openshift-monitoring.svc:8080/admission-prometheusrules/
validate?timeout=5s": x509: certificate signed by unknown authority
DEBUG Still waiting for the cluster to initialize: Cluster operator authentication
is reporting a failure: WellKnownReadyControllerDegraded: need at least 3 kube-
apiservers, got 2
DEBUG Still waiting for the cluster to initialize: Cluster operator authentication
is reporting a failure: WellKnownReadyControllerDegraded: need at least 3 kube-
apiservers, got 2
DEBUG Still waiting for the cluster to initialize: Cluster operator authentication
is reporting a failure: WellKnownReadyControllerDegraded: need at least 3 kube-
apiservers, got 2
DEBUG Cluster is initialized
INFO Waiting up to 10m0s for the openshift-console route to be created...
DEBUG Route found in openshift-console namespace: console
DEBUG Route found in openshift-console namespace: downloads
DEBUG OpenShift console route is created
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/lab/ocp4upi/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.ocp4.example.com
```

```
INFO Login to the console with user: "kubeadm", and password: "rTbDK-j5rqe-JQKVZ-ftEGH"  
DEBUG Time elapsed per stage:  
DEBUG Cluster Operators: 12m44s  
INFO Time elapsed: 12m44s
```

**Note**

It takes around 20 minutes to complete the installation.

At this point, you can monitor the status of the installation from another terminal with commands such as:

```
[lab@utility ~]$ oc get clusteroperator
```

```
[lab@utility ~]$ oc get clusterversion
```

```
[lab@utility ~]$ watch "oc get pods \  
> --all-namespaces | grep -v -E 'Running|Completed'"
```

**Note**

Transient error messages while the installation progresses are normal and can be ignored safely.

The installation takes around 25 minutes in total to complete.

8. Configure the persistent storage for the local registry using the NFS server running on the **utility** server.

Use a new folder named **registry** within the NFS /exports folder on the **utility** server. The folder permissions must be 777.

- 8.1. Create the /exports/registry folder with 777 permissions.

```
[lab@utility ~]$ mkdir /exports/registry  
[lab@utility ~]$ sudo chmod 777 /exports/registry/
```

- 8.2. Create the persistent volume file (PV) **pv.yaml**.

```
[lab@utility ~]$ vi pv.yaml  
apiVersion: v1  
kind: PersistentVolume  
metadata:  
  name: registry-pv  
spec:  
  capacity:  
    storage: 5Gi  
  accessModes:  
    - ReadWriteMany  
  nfs:
```

```
path: /exports/registry
server: 192.168.50.254
persistentVolumeReclaimPolicy: Recycle
```

**Note**

If not done previously, you must configure the **KUBECONFIG** environment variable to interact with the cluster.

```
[lab@utility ~]$ export KUBECONFIG=~/ocp4upi/auth/kubeconfig
```

```
[lab@utility ~]$ oc create -f pv.yaml
persistentvolume/registry-pv created
```

8.3. Create the persistent volume claim (PVC) file **pvc.yaml**.

```
[lab@utility ~]$ vi pvc.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: registry-claim
  namespace: openshift-image-registry
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 5Gi
```

```
[lab@utility ~]$ oc create -f pvc.yaml
persistentvolumeclaim/registry-claim created
```

8.4. Set the image registry operator in a "Managed" state. Edit the configuration of the cluster image registry to add the PVC. Also, configure the image registry to have two pod replicas.

```
[lab@utility ~]$ oc edit configs.imageregistry/cluster
...output omitted...
spec:
...output omitted...
managementState: Managed
...output omitted...
proxy: {}
replicas: 2
requests:
...output omitted...
rolloutStrategy: RollingUpdate
storage:
  pvc:
```

Chapter 7 | Comprehensive Review

```
claim: registry-claim
...output omitted...
config.imageregistry.operator.openshift.io/cluster edited
```

8.5. Verify the configuration of the registry.

NAME	CAPACITY	ACCESS MODES	RECLAIM POLICY	STATUS	CLAIM
		STORAGECLASS	REASON	AGE	
registry-pv	5Gi	RWX	Recycle	106s	Bound openshift-image-registry/registry-claim

NAMESPACE	NAME	STATUS	VOLUME	CAPACITY
ACCESS MODES	STORAGECLASS	AGE		
openshift-image-registry	registry-claim	Bound	registry-pv	5Gi
		107s		RWX

[lab@utility ~]\$ oc get pods -n openshift-image-registry \> -o wide grep ^image-registry					
image-registry-84dbc74f75-bd9zk	32s	10.128.0.34	master03	1/1	Running 0
image-registry-84dbc74f75-bxdgm	25s	10.129.0.34	master02	1/1	Running 0

**Note**

You might see other pods in the Terminating status while the image-registry pods are configured.

9. Verify the cluster health.

9.1. Verify the NTP configuration.

```
[lab@utility ~]$ oc debug node/master01
Creating debug namespace/openshift-debug-node-pnwv9 ...
Starting pod/master01-debug ...
To use host binaries, run chroot /host
Pod IP: 192.168.50.10
If you don't see a command prompt, try pressing enter.
sh-4.4# chroot /host
sh-4.4# cat /etc/chrony.conf
# Use public servers from the pool.ntp.org project.
# Please consider joining the pool (http://www.pool.ntp.org/join.html).
pool 2.rhel.pool.ntp.org iburst
...output omitted...

sh-4.4# sudo chronyc tracking
Reference ID      : 2D4F6F72 (chl.la)
Stratum          : 3
Ref time (UTC)   : Thu Feb 04 12:52:11 2021
System time      : 0.000170327 seconds slow of NTP time
```

```
Last offset      : -0.000190245 seconds
RMS offset      : 0.000336431 seconds
Frequency       : 16.969 ppm slow
Residual freq   : -0.005 ppm
Skew            : 0.184 ppm
Root delay      : 0.041434124 seconds
Root dispersion : 0.002719518 seconds
Update interval : 1031.9 seconds
Leap status     : Normal
sh-4.4# exit
exit
sh-4.4# exit
exit
```

The system is using the NTP pool 2.rhel.pool.ntp.org. It is in sync with the NTP server chl.la of the stratum 3, and the status is Normal.

Repeat the commands above for the other control plane nodes to check the NTP status.

9.2. Verify the cluster resources usage metrics.

```
[lab@utility ~]$ oc adm top node
NAME      CPU(cores)    CPU%    MEMORY(bytes)   MEMORY%
master01  841m         24%    5803Mi          38%
master02  744m         21%    6282Mi          42%
master03  622m         17%    4431Mi          29%
```

9.3. Verify that there are not pods in Failed status.

```
[lab@utility ~]$ oc get pods --all-namespaces | grep -v -E 'Running|Completed'
NAMESPACE     NAME        READY   STATUS    RESTARTS   AGE
```



Note

It is normal to see pods from the openshift-apiserver namespace for a short time after the installation completes.

After a few minutes, the preceding command will not show any output.

9.4. Check the health of the etcd cluster.

```
[lab@utility ~]$ oc rsh -n openshift-etcd etcd-master01
Defaulting container name to etcdctl.
Use 'oc describe pod/etcd-master01 -n openshift-etcd' to see all of the containers
in this pod.
sh-4.4# etcdctl endpoint health --cluster
https://192.168.50.10:2379 is healthy: successfully committed proposal: took =
8.470285ms
https://192.168.50.12:2379 is healthy: successfully committed proposal: took =
9.379322ms
```

Chapter 7 | Comprehensive Review

```
https://192.168.50.11:2379 is healthy: successfully committed proposal: took =
9.99598ms
sh-4.4# exit
exit
```

9.5. Retrieve the API version to check the status of the cluster API.

```
[lab@utility ~]$ curl -k https://api.ocp4.example.com:6443/version
{
  "major": "1",
  "minor": "19",
  "gitVersion": "v1.19.0+9f84db3",
  "gitCommit": "9f84db336d1a77cba52684ecb51bfb197e9b4533",
  "gitTreeState": "clean",
  "buildDate": "2020-10-30T09:33:51Z",
  "goVersion": "go1.15.2",
  "compiler": "gc",
  "platform": "linux/amd64"
```

9.6. Check the availability of the OpenShift console.

```
[lab@utility ~]$ curl -kIs \
> https://console-openshift-console.apps.ocp4.example.com
HTTP/1.1 200 OK
...output omitted...
```

9.7. Verify the availability of the image registry from the cluster nodes.

```
[lab@utility ~]$ oc debug node/master01
Creating debug namespace/openshift-debug-node-zjb2h ...
Starting pod/master01-debug ...
To use host binaries, run chroot /host
Pod IP: 192.168.50.10
If you don't see a command prompt, try pressing enter.
sh-4.4# chroot /host
sh-4.4# curl -kIs https://image-registry.openshift-image-registry.svc:5000/healthz
HTTP/2 200
cache-control: no-cache
date: Tue, 02 Feb 2021 12:01:49 GMT

sh-4.4# exit
exit
sh-4.4# exit
exit

Removing debug pod ...
Removing debug namespace/openshift-debug-node-zjb2h ...
```

Repeat this step for the master02 and master03 control plane nodes.

10. Deploy a rails-postgresql-example application in the test project to perform a functional test of the cluster.

- 10.1. Deploy a test application to verify the proper function of the build process and the registry storage.

```
[lab@utility ~]$ oc new-project test
Now using project "test" on server "https://api.ocp4.example.com:6443".
```

You can add applications to this project with the 'new-app' command. For example, use the following command to build a new example application in Ruby.

```
oc new-app rails-postgresql-example
```

Or, you can use kubectl to deploy a simple Kubernetes application.

```
kubectl create deployment hello-node --image=k8s.gcr.io/serve_hostname
```

```
[lab@utility ~]$ oc new-app rails-postgresql-example
--> Deploying template "openshift/rails-postgresql-example" to project test
```

Rails + PostgreSQL (Ephemeral)

An example Rails application with a PostgreSQL database. For more information about using this template, including OpenShift considerations, see <https://github.com/sclorg/rails-ex/blob/master/README.md>.

WARNING: Any data stored will be lost upon pod destruction. Only use this template for testing.

The following service(s) have been created in your project: rails-postgresql-example, postgresql.

For more information about using this template, including OpenShift considerations, see <https://github.com/sclorg/rails-ex/blob/master/README.md>.

* With parameters:

...output omitted...

--> Creating resources ...

```
secret "rails-postgresql-example" created
service "rails-postgresql-example" created
route.route.openshift.io "rails-postgresql-example" created
imagestream.image.openshift.io "rails-postgresql-example" created
buildconfig.build.openshift.io "rails-postgresql-example" created
deploymentconfig.apps.openshift.io "rails-postgresql-example" created
service "postgresql" created
deploymentconfig.apps.openshift.io "postgresql" created
```

--> Success

Access your application via route 'rails-postgresql-example-test.apps.ocp4.example.com'

Build scheduled, use 'oc logs -f buildconfig/rails-postgresql-example' to track its progress.

Run 'oc status' to view your app.

Chapter 7 | Comprehensive Review

10.2. Watch the build process logs.

```
[lab@utility ~]$ oc logs -f buildconfig/rails-postgresql-example
Cloning "https://github.com/sclorg/rails-ex.git" ...
...output omitted...
Successfully pushed image-registry.openshift-image-registry.svc:5000/test/rails-
postgresql-example@sha256:b97b...ff82
Push successful
```

The image is built and pushed successfully to the internal registry, using the persistent storage.

10.3. Verify that all the pods in the namespace `test` are in the `Running` or `Completed` status.

```
[lab@utility ~]$ oc get pods -n test
NAME                           READY   STATUS    RESTARTS   AGE
postgresql-1-deploy           0/1     Completed  0          2m50s
postgresql-1-pxg5x             1/1     Running   0          2m47s
rails-postgresql-example-1-build 0/1     Completed  0          2m50s
rails-postgresql-example-1-c4ljh 1/1     Running   0          31s
rails-postgresql-example-1-deploy 0/1     Completed  0          52s
rails-postgresql-example-1-hook-pre 0/1     Completed  0          47s
```

10.4. Get the route for the `rails-postgres-example` test application.

```
[lab@utility ~]$ oc get routes -n test
NAME            HOST/PORT
PATH  SERVICES      PORT  TERMINATION  WILDCARD
rails-postgresql-example  rails-postgresql-example-test.apps.ocp4.example.com
                         rails-postgresql-example <all>        None
```

10.5. On the workstation machine, open the Chromium web browser and navigate to the route `rails-postgresql-example-test.apps.ocp4.example.com`.

Welcome to your Rails application on OpenShift

How to use this example application

For instructions on how to use this application with OpenShift, start by reading the [Developer Guide](#).

Deploying code changes

The source code for this application is available to be forked from the [OpenShift GitHub repository](#). You can configure a webhook in your repository to make OpenShift automatically start a build whenever you push your code:

- From the Web Console homepage, navigate to your project
- Click on Browse > Builds
- Click the link with your BuildConfig name
- Click the Configuration tab
- Click the "Copy to clipboard" icon to the right of the "GitHub webhook URL" field
- Navigate to your repository on GitHub and click on repository settings > webhooks > Add webhook
- Paste your webhook URL provided by OpenShift
- Leave the defaults for the remaining fields — that's it!

After you save your webhook, if you refresh your settings page you can see the status of the ping that GitHub sent to OpenShift to verify it can reach the server.

Managing your application

Documentation on how to manage your application from the Web Console or Command Line is available at the [Developer Guide](#).

Web Console

You can use the Web Console to view the state of your application components and launch new builds.

Command Line

With the [OpenShift command line interface \(CLI\)](#), you can create applications and manage projects from a terminal.

Development Resources

- [OpenShift Documentation](#)
- [Openshift Origin GitHub](#)
- [Source To Image GitHub](#)
- [Getting Started with Ruby on OpenShift](#)
- [Stack Overflow questions for OpenShift](#)
- [Git documentation](#)

Evaluation

As the student user on the **workstation** machine, use the **lab** command to grade your work. Correct any reported failures and rerun the command until successful.

```
[lab@utility ~]$ exit
...output omitted...
[student@workstation ~]$ lab comprehensive-review grade
```

Finish

As the student user on the **workstation** machine, use the **lab** command to complete this exercise.

```
[student@workstation ~]$ lab comprehensive-review finish
```



Note

The **lab comprehensive-review finish** command does not destroy the cluster. Running the **lab comprehensive-review grade** script after the **lab comprehensive-review finish** command shows that the three-node cluster exists.

If you wish to start over, reset the **bootstrap**, **master01**, **master02**, and **master03** machines, and then start again at the step that instructs you to install RHCOS in the **bootstrap**, **master01**, **master02**, and **master03** machines from the PXE menu.

This concludes the lab.

