Dell EMC Ready Stack: Red Hat OpenShift Container Platform 4.2

Dell EMC PowerEdge R-Series Servers and PowerSwitch Networking

January 2021

H18022.1

Deployment Guide

Abstract

This deployment guide provides a validated procedure for deploying Red Hat OpenShift Container Platform 4.2 on Dell EMC PowerEdge R-Series servers.

Dell Technologies Solutions

Copyright

Contents

Chapter 1 Introduction

This chapter presents the following topics:

Solution overview

Red Hat OpenShift Container Platform is an open-source application deployment platform that is based on Kubernetes container orchestration technology. Containers are stand-alone processes that run within their own environment, independent of the operating system and underlying infrastructure. Red Hat OpenShift Container Platform helps you develop, deploy, and manage container-based applications.

As part of Red Hat OpenShift Container Platform, Kubernetes manages containerized applications across a set of containers or hosts and provides mechanisms for the deployment, maintenance, and scaling of applications. The container run-time engine packages, instantiates, and runs containerized applications.

An OpenShift cluster consists of one or more control planes and a set of nodes. Kubernetes allocates an IP address from an internal network to each pod so that all containers within the pod behave as if they were on the same host. Each pod has its own IP address, which means the pods can be treated like physical hosts or virtual machines for port allocation, networking, naming, service discovery, load balancing, application configuration, and migration. Dell EMC recommends creating a Kubernetes service that enables your application pods to interact rather than requiring that the pods communicate directly using their IP addresses.

A fully functioning Domain Name System (DNS) is also crucial in the deployment and operation of your container ecosystem. Red Hat OpenShift Container Platform has an integrated DNS enabling the services to be found through DNS Service record (SRV) entries or through the service IP/port registrations.

Dell EMC Ready Stack for Red Hat OpenShift Container Platform is a proven design to help companies accelerate their container deployments and cloud-native adoption. Dell EMC delivers tested, validated, and documented design guidance to help customers rapidly deploy OpenShift Container Platform on Dell EMC infrastructure by minimizing time and effort. For more information, see the *Dell EMC Ready Stack: Red Hat OpenShift Container Platform 4.2 Design Guide*.

Web console

Red Hat OpenShift Container Platform 4.2 provides a web-based console for cluster and application management. After the initial cluster deployment, the web console is ready for use. For more information, see OpenShift web console. The web-based console is an extensive management interface for developers and administrators. Users can access the command line interface (CLI) from the Cluster System Admin Host (CSAH) node. For more information, see Validating the cluster operators on page 25.

Note: CSAH is Dell EMC nomenclature.

Document purpose

Note: This deployment guide may contain language from third party content that is not under Dell's control and is not consistent with Dell's current guidelines for Dell's own content. When such third party content is updated by the relevant third parties, this guide will be revised accordingly.

This guide describes the infrastructure that is required to deploy and operate OpenShift Container Platform and provides a validated process for deploying a production-ready OpenShift Container Platform cluster. The guide provides the information that you need to facilitate readiness for Day 2 operations. Topics that the guide addresses include:

- Setting up switch configuration
- Preparing the CSAH node
- Installing OpenShift and creating a cluster
- Scaling OpenShift cluster nodes
- Platform user authentication
- Deploying applications
- Setting routes for applications
- OpenShift cluster monitoring

This guide provides a reference design for deploying version 4.2 of OpenShift Container Platform on Dell EMC PowerEdge servers and Dell EMC Networking switches. Dell EMC strongly recommends that you complete the validation steps that the guide describes and ensure that you are satisfied that your application can operate smoothly before proceeding with development or production use.

For more information, see the Red Hat OpenShift Container Platform 4.2 Documentation.

Audience

This deployment guide is for system administrators and system architects. Some experience with Docker and Red Hat OpenShift Container Platform technologies is recommended. To familiarize yourself with the architecture and design of the solution before planning your deployment, see the *Dell EMC Ready Stack: Red Hat OpenShift Container Platform 4.2 Design Guide* on Dell EMC Solutions Info Hub for Containers.

We value your feedback

Dell Technologies and the authors of this document welcome your feedback on the solution and the solution documentation. Contact the Dell Technologies Solutions team by email or write to the authors of this guide with your queries.

Authors: Michael Tondee, Umesh Sunnapu

Contributors: Scott Powers, John Terpstra, Aighne Kearney, David Cain (Red Hat), Ken Holtz (Red Hat), Mark Russell (Red Hat)

Note: For links to additional documentation for this solution, see the Dell Technologies Solutions Info Hub for SQL Server.

Chapter 2 Configuring Switches

This chapter presents the following topics:

Overview

This chapter describes where to locate the sample configuration files that Dell EMC has provided and how to customize the files for your environment.

Typographical conventions

Configuration instructions in this guide use certain typographical conventions to designate commands and screen output.

Command syntax is identified by `Courier` font. Information that is specific to your environment is placed inside <> symbols. For example:

- Deployment guide command reference: `OS10(config)# hostname <S5232F>`
- On the S5248F-ON switch, enter: `OS10(config)# hostname <S5248>`

Screen output is presented in **`bold type`**.

Sample configuration files

The sample switch configuration files are available at openshift-bare-metal. These files enable you to easily configure the switch used for the OpenShift Container Platform cluster.

1. Clone the repository by running the following command:

```
git clone https://github.com/dell-esg/openshift-bare-metal.git
```

2. Change to the **examples** directory.

Note: If you use different hardware or need different configurations, modify the configuration files correspondingly.

Customizing the Dell EMC switches

The following table shows connections from each server to the switch ports with a 100 G NIC in PCI Slot2 connected to a S5232F-ON switch:

Table 1. Server-to-switch port connections in the deployment

Role	Port	S5232F_ON switch	Public IP VLAN 461 100.82.46.0/26	S3048-ON switch	iDRAC IP VLAN 34 100.82.34.0/24
csah	ens2f0	1/1/1	100.82.46.20	1/1/1	100.82.34.20
etcd-0	ens2f0	1/1/2	100.82.46.21	1/1/2	100.82.34.21
etcd-1	ens2f0	1/1/3	100.82.46.22	1/1/3	100.82.34.22
etcd-2	ens2f0	1/1/4	100.82.46.23	1/1/4	100.82.34.23
worker1	ens2f0	1/1/5	100.82.46.24	1/1/5	100.82.34.24
worker2	ens2f0	1/1/6	100.82.46.25	1/1/6	100.82.34.25
bootstrap/worker3	ens2f0	1/1/7	100.82.46.26	1/1/7	100.82.34.26

The following table shows the firmware versions that are supported for the switch models:

Table 2. Switch model and firmware specifications

Dell EMC switch model	OOB management IP	Firmware version	Default username	Default password
S3048-ON	100.82.33.46	10.5.0.2	admin	admin
S5232F-ON	100.82.33.45	10.5.0.2	admin	admin

To modify the switches for your environment:

3. Download the switch config files from GitHub by running the following command:

```
git clone https://github.com/dell-esg/openshift-bare-metal.git
```

4. Modify the sample switch config files to match your VLAN and IP schemes.

   The deployment utilizes untagged VLANs that use switchport access for nodes and tagged port-channels for switch uplinks.

   The deployment sample uses:

   - VLAN_461 configured for Public a /26 network
   - Dedicated iDRAC VLAN_34 /24 network and a dedicated /24 switch out of band (OOB) management network
   - Single port on 100 G Mellanox X5 DP NIC in PCI Slot 2

   Note: The serial port baud-rate is 115200.

Configuring the Dell EMC switches

This section describes the OS10 initial OOB management IP setup and provides sample switch configuration directions copied to `running-config`.

Follow these steps:

5. Power on the switches, connect to the serial debug port, set the hostname, and configure a static IP address for management 1/1/1.

   The following code sample shows an S5232F-ON switch. (Use the same process for S3048-ON switches.)

```
OS# configure terminal
OS (config)# hostname S5232F
S5232F(config)# interface mgmt 1/1/1
S5232F(conf-if-ma-1/1/1)# no shutdown
S5232F(conf-if-ma-1/1/1)# no ip address dhcp
S5232F(conf-if-ma-1/1/1)# ip address 100.82.33.45/24
S5232F(conf-if-ma-1/1/1)# exit
S5232F(config)# management route 0.0.0.0/0 100.82.33.1
```

6. Copy the modified sample switch configuration to `running-configuration` and configure the switch by running:

```
S5232F# copy scp://<user>@<hostip>/<path to downloaded
S5232F config file> running-configuration
S5232F# write memory
```

Chapter 3 Setting Up the CSAH Node

This chapter presents the following topics:

Overview

The services required to create an OpenShift Container Platform cluster are set up in the CSAH node. This chapter provides information about installing the CSAH node and running the OpenShift Container Platform cluster prerequisites.

Preparing the CSAH node

To install Red Hat Enterprise Linux 8.0 in the CSAH node:

7. Follow the guidelines in Red Hat Enterprise Linux 8.0 Installation.

   The Red Hat Enterprise Linux Installation UI is displayed, as shown in the following figure:

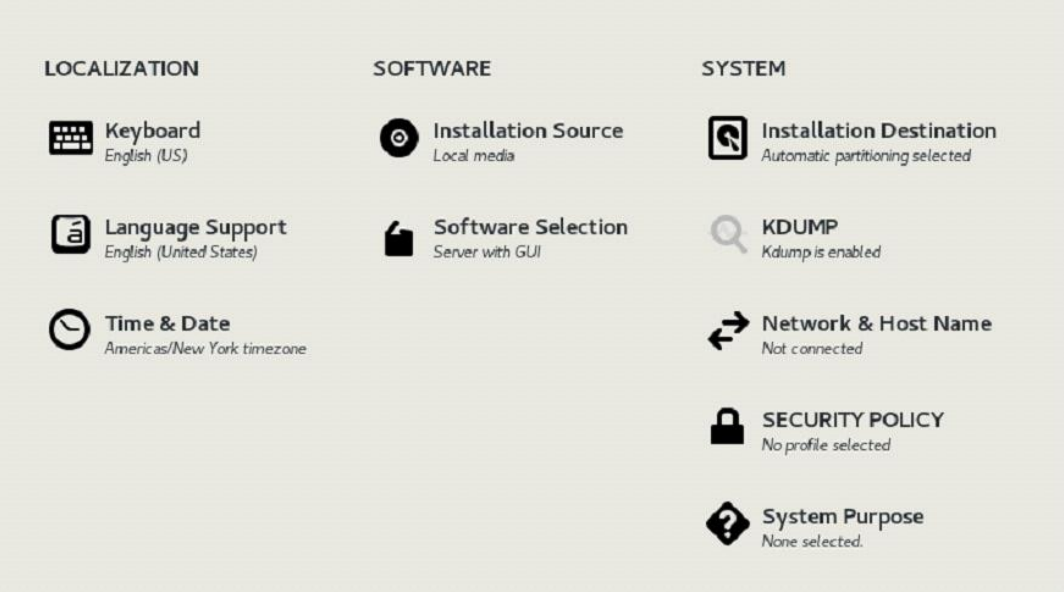

Figure 1. Operating system installation UI options

Note: Under **Software Selection**, ensure that **Server with GUI** is selected.

8. Run the following tasks as user ‘root’ unless directed otherwise.

   After the installation is complete, set the hostname to reflect the naming standards by running:

```
hostnamectl set-hostname <hostname>.<clustername>.<base
domain>
hostnamectl set-hostname csah.ocp.example.com
```

9. Assign an IP address to the interface. As part of our validation, we used `interface ens2f0`.

```
nmcli connection modify ens2f0 ipv4.method manual
ipv4.address <ipaddress/cidr> connection.autoconnect yes
ipv4.gateway <gateway> ipv4.dns <dns-server> ipv4.dns-
search <clustername.base domain>
```

Note: The assigned IP address must be able to reach the internet and the DNS must be able to resolve `subscription.rhsm.redhat.com`. For this installation process, internet access for all nodes is required. Red Hat provides an offline deployment method that is beyond the scope of this deployment guide.

10. Add the newly created hostname in the `/etc/hosts` file along with its IP address, as shown in the following command:

```
100.82.46.20  csah  csah.ocp.example.com
```

11. Enable the ansible-2.8-for-rhel-8-x86_64-rpms repository by using subscription-manager:

```
subscription-manager register --username
<subscription.user> --password <subscription.password> --
force
subscription-manager attach --pool=<pool id>
subscription-manager repos --enable=ansible-2.8-for-rhel-
8-x86_64-rpms
```

12. Install the following rpms:

```
yum install -y git jq ansible
```

13. Create a user to run playbooks by running:

```
useradd <user>
```

Note: Do not use ‘core’ as the username. User core is used as part of the OpenShift Container Platform cluster configuration. The remainder of this guide assumes that user ‘ansible’ is created to run playbooks.

14. Set up password-less access to the CSAH FQDN. As user ansible, run:

```
ssh-keygen (press enter and go by defaults for the next
set of questions)
cat .ssh/id_rsa.pub > .ssh/authorized_keys
chmod 600 .ssh/authorized_keys
```

15. As root, provide permissions to the user that you have just created to run all commands without being prompted for a password. The content in bold provides a reference.

```
visudo
# add the following line after # %wheel ALL=(ALL)
NOPASSWD: ALL
ansible ALL=(ALL) NOPASSWD: ALL
```

16. As user ansible, download the Ansible playbooks from GitHub by running:

```
git clone
https://github.com/dell-esg/openshift-bare-metal.git
```

Preparing and running the Ansible playbooks

Perform the tasks below as user ansible. The Ansible playbooks create resources that might be already configured and running in your data center, such as DNS, DHCP, and PXE. If you are not sure, consult with your network administrator before running the playbooks. You can modify the sample YAML file `ocp.yml` to remove individual roles if necessary. Dell EMC recommends implementing into your existing DNS, DHCP and PXE services whenever possible. For this deployment, all required services are configured automatically to run on the CSAH node.

Follow these steps:

1. Create a directory into which to download the following Red Hat software:
   - openshift-client-linux-4.2
   - openshift-install-linux-4.2
   - openshift-installer-kernel-4.2
   - openshift-metal-uefi-4.2
   - openshift-installer-initramfs-4.2
   - pullsecret

   A sample YAML file is provided as a hosts file in the Ansible directory.

   Note: An explanation is provided for each variable defined in the hosts YAML file. Review the sample values to gain a better understanding of the content and modify the values as necessary for your environment.

2. As user ansible, run:

   ```
   ansible-playbook -i <hosts file> ocp.yml
   ```

   After a successful Ansible playbook execution, the CSAH node is installed and configured with http, dhcp, dns, haproxy, and pxe services. In addition, the `install-config.yaml` file is generated and the `ignition config` files are created and made available over HTTP.

3. As root, modify the DNS to point to the CSAH node by running:

   ```
   $ nmcli con mod ens2f0 ipv4.dns 100.82.46.20 ipv4.dns-
   search ocp.example.com

   $ systemctl restart NetworkManager
   ```

   This example uses `ocp` as the cluster name, `example.com` as the base domain, and `100.82.46.20` as the CSAH node IP.

Chapter 4 Deploying OpenShift 4.2

This chapter presents the following topics:

Introduction

This section describes the steps for deploying OpenShift Container Platform 4.2. To create an OpenShift Container Platform cluster, you must create the following nodes in the specified order:

- Bootstrap node
- Control plane nodes
- Worker (or compute) nodes

Notes:

This guide assumes that 'NIC in Slot 2 Port 1' is used for PXE installation. Replace the interface if necessary to match the configuration.

All the nodes must be running in UEFI mode for the playbooks executed in the CSAH node to work effectively.

Creating a bootstrap node

To create a bootstrap node:

1. Connect to the iDRAC of the bootstrap node and open the virtual console.
2. Power on the bootstrap node from the iDRAC console.
3. To ensure that the ens2f0 interface is set for PXE boot:
    a. Press **F2** to enter **System Setup**.
    b. Select Device Settings > NIC in Slot 2 Port 1 > NIC Configuration.
    c. From the **Legacy Boot Protocol** menu, select **PXE**.
    d. Select **Finish** to return to **System Setup**.
    e. Select **System BIOS** > **Network Settings**.
    f. Under **UEFI PXE Settings**, select **PXE Device1 Settings**.
    g. From the I**nterface** menu, select **NIC in Slot2 Port1 Partition 1**.
    h. Save your changes and reboot the node.

    The system boots automatically into the PXE network and the PXE menu is displayed, as shown in the following figure:

Figure 2. iDRAC console PXE menu

4. Select **bootstrap**.

   After the node installation is complete, the system reboots automatically.

5. Before the node boots again into the PXE, ensure that the hard disk is placed above the PXE interface in the boot order:

   a. Press **F2** to enter System Setup.

   b. Select **Boot Settings** > **UEFI Boot Settings** > **UEFI Boot Sequence**.

   c. Select **PXE Device 1** and click the - sign.

   d. Repeat the preceding step until **PXE Device 1** is at the bottom of the boot menu.

   e. Click **OK**, and then click **Back**.

   f. Click **Finish** and save your changes.

6. Let the node boot into the hard drive where the OS is installed.

7. After the node restarts, check that the hostname in the iDRAC console is displaying `bootstrap` and that the correct IP is address assigned, as shown in the following figure:

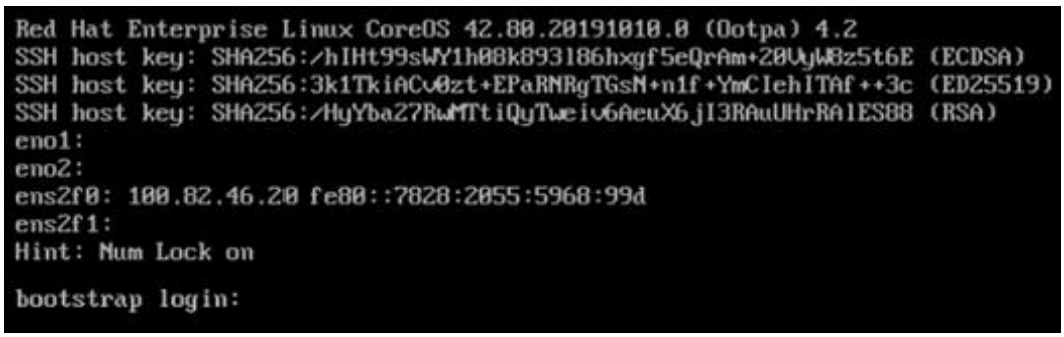

Figure 3. iDRAC console bootstrap node

Creating control plane (etcd-*) nodes

To create control plane nodes:

1. Connect to the iDRAC of the control plane node and open the virtual console.
2. Power on the control plane node.
3. To ensure that the `ens2f0` interface is set for PXE boot:

   a. Press **F2** to enter **System Setup**.

   b. Select **Device Settings** > **NIC in Slot 2 Port 1** > **NIC Configuration**.

   c. From the **Legacy Boot Protocol** menu, select **PXE**.

   d. Click **Finish** to go back to **System Setup**.

   e. Select **System BIOS** > **Network Settings**.

   f. Under **UEFI PXE Settings**, select **PXE Device1 Settings**.

   g. From the **Interface** menu, select **NIC in Slot2 Port1 Partition 1**.

h. Save the changes and reboot the node.

The system automatically boots into PXE network and the PXE menu is displayed, as shown in the following figure:

Figure 4. iDRAC console: PXE menu

4. Select **etcd-0** (because this is the first node). Let the system reboot after installation.
5. Before the node boots again into the PXE, ensure that the hard disk is placed above the PXE interface in the boot order:
   a. Press **F2** to enter **System Setup**.
   b. Select **Boot Settings** > **UEFI Boot Settings** > **UEFI Boot Sequence**.
   c. Select **PXE Device 1** and click the **–** icon.
   d. Repeat the preceding step until **PXE Device 1** is at the bottom of the boot menu.
   e. Click **OK**, and then click **Back**.
   f. Click **Finish** and save the changes.
6. Let the node boot into the hard drive where the operating system is installed.
7. After the node restarts, ensure that the hostname and IP address are aligned, as shown in the following figure:

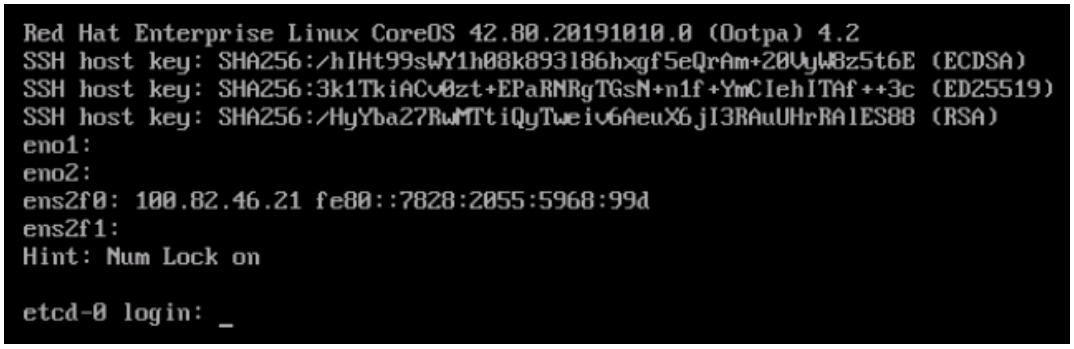

Figure 5. Control plane (etcd-0) iDRAC Consol

After the installation is complete, the node reboots to fetch the control plane configuration file.

8. Repeat steps 1-7 for the remaining two control plane nodes. For the second control plane node, select **etcd-1** and for the third control plane node, select **etcd-2**.

Creating worker nodes

To create worker nodes:

1. Connect to the iDRAC of the worker node and open the virtual console.
2. Power on the worker node.
3. To ensure that the `ens2f0` interface is set for PXE boot:
    a. Press **F2** to enter **System Setup**.
    b. Select Device Settings > NIC in Slot 2 Port 1 > NIC Configuration.
    c. From the **Legacy Boot Protocol** menu, select **PXE**.
    d. Select **Finish** to go back to **System Setup**.
    e. Select **System BIOS** > **Network Settings**.
    f. Under **UEFI PXE Settings**, select **PXE Device1 Settings**.
    g. From the **Interface** menu, select **NIC in Slot2 Port1 Partition 1**.
    h. Save your changes and reboot the node.

The system automatically boots into the PXE network, as shown in the following figure:

Figure 6. iDRAC console: PXE menu

4. Select **worker1** (because this is the first worker node). Let the system reboot after the installation.
5. Before the node boots again into the PXE, ensure that the hard disk is placed above the PXE interface in the boot order:
    a. Press **F2** to enter **System Setup**.
    b. Select **Boot Settings** > **UEFI Boot Settings** > **UEFI Boot Sequence**.
    c. Select **PXE Device 1** and click the **-** sign.
    d. Repeat the preceding step until **PXE Device 1** is at the bottom of the boot menu.
    e. Click **OK** and then click **Back**.
    f. Click **Finish** and save the changes.
6. Let the node boot into the hard drive where the OS is installed.
7. When the node restarts, ensure that the hostname and IP address are aligned, as shown in the following figure:

Figure 7. iDRAC console: PXE menu

After the installation is complete, the node reboots to fetch the worker configuration file.

8. Repeat steps 1-7 for the second worker node. This time, select **worker2** from the PXE menu in Step 4.

Bootstrap steps

Perform the following tasks as user **core** after the bootstrap, control plane, and worker nodes are installed in the CSAH node.

Note: Cluster administration is performed on the CSAH node only.

The `hosts` file shows the value specified for the `install_dir` variable. The following steps use `openshift` for this variable:

1. In the `.bash_profile` file, under `/home/core`, add the following entry:

```
export KUBECONFIG=/home/core/openshift/auth/kubeconfig
export PATH=$PATH:/home/core
```

   Run `cd && . .bash_profile` to ensure that the profile changes are reflected immediately.

2. Switch to the `core` home directory and run:

```
cd /home/core/openshift
./openshift-install --dir=openshift wait-for bootstrap-
complete --log-level debug
DEBUG OpenShift Installer v4.2.0DEBUG Built from commit
90ccb37ac1f85ae811c50a29f9bb7e779c5045fbINFO Waiting up
to 30m0s for the Kubernetes API at
https://api.ocp.example.com:6443...INFO API
v1.14.6+2e5ed54 upINFO Waiting up to 30m0s for
bootstrapping to complete...DEBUG Bootstrap status:
completeINFO It is now safe to remove the bootstrap
resources
```

3. As root, remove the bootstrap node entry for ports 6443 and 22623 from the `haproxy.cfg` file under `/etc/haproxy`.

4. As root, restart the haproxy service to reflect the changes by running:

```
systemctl restart haproxy
```

5. To ensure that the configuration is successful, run the following command as user core:

```
oc whoami
system:admin
```

Validating and approving Certificate Signing Requests

Unless directed otherwise, run all the following commands as user core.

Note: Cluster administration is performed on the CSAH node only.

To validate the Certificate Signing Requests (CSRs):

1. Ensure that all the control plane nodes are visible and in READY status by running:

```
oc get nodes
NAME                          STATUS   ROLES    AGE   VERSION
etcd-0.ocp.example.com   Ready    master   54m
v1.14.6+c07e432da
etcd-1.ocp.example.com   Ready    master   54m
v1.14.6+c07e432da
etcd-2.ocp.example.com   Ready    master   54m
v1.14.6+c07e432da
```

2. Check for any pending certificates and approve them to add the worker nodes as part of the cluster. The following output is displayed:

```
oc get csr
NAME        AGE     REQUESTOR
CONDITION
csr-7nrxg   34m     system:serviceaccount:openshift-
machine-config-operator:node-bootstrapper   Pending
csr-7rr6w   47m     system:serviceaccount:openshift-
machine-config-operator:node-bootstrapper   Pending
csr-fdt85   60m     system:serviceaccount:openshift-
machine-config-operator:node-bootstrapper
Approved,Issued
csr-fwtcq   59m     system:node:etcd-1.ocp.example.com
Approved,Issued
csr-gcz5g   4m26s   system:serviceaccount:openshift-
machine-config-operator:node-bootstrapper   Pending
csr-h7w54   60m     system:serviceaccount:openshift-
machine-config-operator:node-bootstrapper
Approved,Issued
```

Note: It is sometimes necessary to run the command twice to ensure that the worker nodes are seen as part of `oc get nodes`.

3. Approve all pending CSRs by running a single command:

```
oc get csr -ojson | jq -r '.items[] | select(.status ==
{} ) | .metadata.name' | xargs oc adm certificate approve
certificatesigningrequest.certificates.k8s.io/csr-7nrxg
approved
certificatesigningrequest.certificates.k8s.io/csr-7rr6w
approved
certificatesigningrequest.certificates.k8s.io/csr-gcz5g
approved
certificatesigningrequest.certificates.k8s.io/csr-q942b
approved
certificatesigningrequest.certificates.k8s.io/csr-tddzj
approved
certificatesigningrequest.certificates.k8s.io/csr-w8fxd
approved
certificatesigningrequest.certificates.k8s.io/csr-x76lp
approved
```

4. Ensure that the worker nodes are part the cluster nodes by running the following command:

```
oc get nodes
NAME                          STATUS   ROLES    AGE
VERSION
etcd-0.ocp.example.com    Ready    master   65m
v1.14.6+c07e432da
etcd-1.ocp.example.com    Ready    master   64m
v1.14.6+c07e432da
etcd-2.ocp.example.com    Ready    master   65m
v1.14.6+c07e432da
worker1.ocp.example.com   Ready    worker   3m12s
v1.14.6+c07e432da
worker2.ocp.example.com   Ready    worker   3m9s
v1.14.6+c07e432da
```

5. For a nonproduction configuration, set up a temporary image registry by running the following command:

```
oc patch configs.imageregistry.operator.openshift.io
cluster --type merge --patch
'{"spec":{"storage":{"emptyDir":{}}}}'
config.imageregistry.operator.openshift.io/cluster
patched
```

Validating the cluster operators

The OpenShift cluster consists of multiple cluster operators. For more information about operators, see the *Dell EMC Ready Stack: Red Hat OpenShift Container Platform 4.2 Design Guide*.

All the operators must be in the 'available' state. To verify this:

1. Confirm that the cluster operators are all displayed as TRUE in the AVAILABLE column, as shown in the following figure:

```
[core@csah ~]$ oc get clusteroperators
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED   SINCE
authentication                             4.2.0     True        False         False      8m8s
cloud-credential                           4.2.0     True        False         False      75m
cluster-autoscaler                         4.2.0     True        False         False      70m
console                                    4.2.0     True        False         False      10m
dns                                        4.2.0     True        False         False      74m
image-registry                             4.2.0     True        False         False      5m40s
ingress                                    4.2.0     True        False         False      12m
insights                                   4.2.0     True        False         False      75m
kube-apiserver                             4.2.0     True        False         False      73m
kube-controller-manager                    4.2.0     True        False         False      72m
kube-scheduler                             4.2.0     True        False         False      72m
machine-api                                4.2.0     True        False         False      75m
machine-config                             4.2.0     True        False         False      74m
marketplace                                4.2.0     True        False         False      70m
monitoring                                 4.2.0     True        False         False      12m
network                                    4.2.0     True        False         False      73m
node-tuning                                4.2.0     True        False         False      72m
openshift-apiserver                        4.2.0     True        False         False      71m
openshift-controller-manager               4.2.0     True        False         False      72m
openshift-samples                          4.2.0     True        False         False      68m
operator-lifecycle-manager                 4.2.0     True        False         False      74m
operator-lifecycle-manager-catalog         4.2.0     True        False         False      74m
operator-lifecycle-manager-packageserver   4.2.0     True        False         False      71m
service-ca                                 4.2.0     True        False         False      75m
service-catalog-apiserver                  4.2.0     True        False         False      72m
service-catalog-controller-manager         4.2.0     True        False         False      72m
storage                                    4.2.0     True        False         False      71m
```

Figure 8. Cluster operators in OpenShift Cluster 4.2

2. To complete the cluster installation, run:

```
./openshift-install --dir=openshift wait-for install-complete --
log-level debug
DEBUG OpenShift Installer v4.2.0
DEBUG Built from commit 90ccb37ac1f85ae811c50a29f9bb7e779c5045fb
INFO Waiting up to 30m0s for the cluster at
https://api.ocp.example.com:6443 to initialize...
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/core/openshift/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-
openshift-console.apps.ocp.example.com
INFO Login to the console with user: kubeadmin, password: xxxxx-
xxxxx-xxxxx-xxxx-xxxx
```

OpenShift web console

To access OpenShift through a browser, you must obtain the URL of the routes console. Accessing the cluster using the web console provides all functionalities including, but not limited to, pod creation and application deployment.

To obtain the existing routes in all namespaces:

1. Run the following command:

```
oc get routes --all-namespaces

NAMESPACE                  NAME
HOST/PORT
SERVICES               PORT    TERMINATION
WILDCARD

openshift-authentication   oauth-openshift       oauth-
openshift.apps.ocp.example.com
oauth-openshift        6443    passthrough/Redirect   None

openshift-console          console               console-
openshift-console.apps.ocp.example.com
console                https   reencrypt/Redirect     None
```

Note: Obtain the **HOST/PORT** value for the console in the `openshift-console` namespace.

2. Open a browser and enter the following URL: `https://console-openshift-console.apps.ocp.example.com` (this URL is derived from the output in Step 1).
3. Log in as user kubeadmin using the password that is located in: `/home/core/<install dir>/auth/kubeadmin-password`.

Authenticating platform users

OpenShift supports different authentication methods. For more information, see the Red Hat document Understanding authentication.

Note: This deployment guide explains how to configure identity providers for `htpasswd` and Active Directory, but only one method is needed.

Htpasswd authentication

Unless otherwise directed, run the following commands in the CSAH node as user core.

To set up the prerequisites for user authentication using the OpenShift cluster:

1. Create a htpasswd file on the CSAH node by running the following command:

```
cd /home/core/<install directory>/
htpasswd -c -B -b htpasswd dellemc1 Password1
htpasswd -b htpasswd mike Password2
htpasswd -b htpasswd umesh Password3
htpasswd -b htpasswd john Password4
htpasswd -b htpasswd user1 Password5
```

2. Create a secret (containing a username and passwords) for `htpasswd` using the `htpasswd` file you created in the preceding step:

```
oc create secret generic htpass-secret --from-
file=htpasswd=/home/core/openshift/htpasswd -n openshift-
config
```

3. Create a custom resource (CR). Save the following contents in a file:

```
apiVersion: config.openshift.io/v1
kind: OAuth
metadata:
  name: cluster
spec:
  identityProviders:
  - name: htpasswd
    mappingMethod: claim
    type: HTPasswd
    htpasswd:
      fileData:
        name: htpass-secret
```

4. Apply the CR by running:

```
oc apply -f <file name>
```

5. Log in as a user created with `htpasswd`:

```
oc login -u <username>
Authentication required for
https://api.ocp.example.com:6443 (openshift)
Username: <username>
Password: <password>
```

```
Login successful.
You don't have any projects. You can try to create a new
project, by running    oc new-project <projectname>
```

Windows Active Directory authentication

The prerequisites for user authentication using Windows Active Directory (AD) are:

- OpenShift Cluster
- AD
- Sample users created in AD

Note: Unless otherwise directed, run the commands in the CSAH node as user core.

Perform the following steps to integrate OpenShift and AD for user authentication:

1. Create a secret containing a password that can connect to AD—typically, an admin password—and search the tree by running the following command:

```
oc create secret generic <secret-name> –from-
literal=bindPassword=<password> -n openshift-config
secret/<secret-name> created
```

If you are not using certificates for authentication, skip to Step 5.

2. Obtain the certificate used by AD. The certificate is displayed at the end of the output from:

```
--BEGIN CERTIFICATE---- to ---------END CERTIFICATE--
```

3. Copy the lines from `BEGIN CERTIFICATE to END CERTIFICATE` to a new file:

```
openssl s_client -connect <AD ip>:<SSL Port> 2>/dev/null
| openssl x509 -text
```

For reference, `ad.cert` is provided as the name of the new file.

4. Create a ConfigMap referencing the AD certificate file path by running the following command:

```
oc create configmap <config-map-name> --from-file=ca.crt=<path to
file ad.cert> -n openshift-config
```

For reference, `ca-config-map` is provided as the name of the configmap.

5. To create a CR for that AD identity provider, use the following sample CR and replace all the values that are specified in <>:

```
apiVersion: config.openshift.io/v1
kind: OAuth
metadata:
    name: cluster
spec:
   identityProviders:
   - name: <ip address of active directory>
     mappingMethod: claim
```

```
     type: LDAP
ldap:
  attributes:
     id:
      - dn
    email:
      - mail
    name:
    - cn
    preferredUsername:
    - sAMAccountName
  bindDN: <provide the bindDN of the user who can query
root dn referenced in url>
  bindPassword:
     name: <provide the secret name for the user
specified in bindDN>
  ca:
    name: <provide the name of the configmap created>
  insecure: false
  url: "ldap://<AD FQDN>/<root dn>?sAMAccountName"
```

If you are skipping Steps 2 to 4, delete `ca`, `name` and set `insecure` to `true`. Ensure that the **AD FQDN** entry is added to the DNS config in the CSAH node, and then save the file.

6. Apply the CR by running:

```
oc apply -f <CR file name>
```

7. Log in to the cluster as a user from AD. Enter the user password when prompted:

```
oc login -u <username>
Authentication required for
https://api.ocp.example.com:6443 (openshift)
Username: <username>
Password: <password>
Login successful.
You don't have any projects. You can try to create a new
project, by running
       oc new-project <projectname>
```

Assigning a cluster admin role to the AD user

To assign a cluster-admin role to the AD user:

1. Ensure that the user is listed by running the following command:

```
oc get users
NAME       UID                                    FULL
NAME          IDENTITIES
```

```
ocpadmin    e5fd24c8-f504-11e9-a918-0a580a820029
openshift redhat
100.82.46.10Q049b3BlbnNoaWZ0IHJlZGhhdCxDTj1Vc2
```

The AD IP address is 100.82.46.10.

2. To get a list of all available cluster roles, run

   `oc get clusterrole --all-namespaces`.

3. Assign a `cluster-admin` cluster role to the user `ocpadmin` by running:

```
oc adm policy add-cluster-role-to-user cluster-admin
ocpadmin
clusterrole.rbac.authorization.k8s.io/cluster-admin
added: "ocpadmin"
```

4. Verify that the `cluster-admin` role is assigned to that user by running:

```
oc get clusterrolebindings -o json | jq '.items[] |
select(.subjects[0].name=="ocpadmin")' | jq
'.roleRef.name'
"cluster-admin"
```

Image registry storage recommendations

Take account of the following recommendations for the container image registry:

- Dell EMC recommends using the Dell EMC Unity 380F All Flash array for image-registry storage.
- Red Hat does not recommend using the Red Hat Enterprise Linux-backed NFS server for image-registry storage, although this is possible for POC implementations.
- Although POC implementations can use “EmptyDir” for `image-registry` storage, images pushed to the registry are not saved after a reboot.
- The OC commands in this deployment guide configure an NFS-backed Persistent Volume (PV) that is attached to a Unity 380F All-Flash array for image registry storage.

For information about configuring image-registry in bare-metal installations, see the Red Hat OpenShift 4.2 Install documentation.

For information about registry values and configuration, see the Red Hat OpenShift 4.2 Image Registry documentation.

Configuring Unity NFS volume for the image registry

We validated the deployment that is described in this guide using Unity 380F software version 5.0.0.0.5.116. The following code snippet shows the Unity 380F NAS server and NFS share details:

```
Unitynfs
Description:
NAS Server:
   unitynas
File System:
   unitynfs
Local Path:
/unitynfs/
Default Access:
Read/Write, allow Root
Hosts:
0
All Export paths:
100.82.46.8:/unitynfs
```

Prerequisites

To configure the Unity NFS volume for the image registry, you must:

- Identify the configured provider
- Obtain an OpenShift cluster login with administrator permissions

Note: Unless directed otherwise, run the following commands in the CSAH node as user core.

1. Create a PV file `nfsimageregpv.yml` with the following content. Modify the path and server values for your environment:

```
apiVersion: v1
kind: PersistentVolume
metadata:
    name: nfs-image-registry
    namespace: openshift-image-registry
spec:
   capacity:
      storage: 100Gi
   accessModes:
   - ReadWriteMany
   nfs:
         path: /unitynfs
server: 100.82.46.8
   persistentVolumeReclaimPolicy: Retain
```

2. Apply the CR you created in Step 3 of Htpasswd authentication:

```
oc apply -f nfsimageregpv.yml
```

3. Edit the registry configuration to use the newly created PV:

```
oc edit configs.imageregistry.operator.openshift.io
```

4. Scroll down to Storage and add the following lines:

```
Storage:
    pvc:
        claim:
```

5. Save the file, and then quit by running `:wq`

   The image-registry cluster operator is updated and a new PV claim is created using the PV that we created using Unity NFS storage.

Chapter 5 Adding Worker Nodes

This chapter presents the following topics:

Removing the bootstrap node

We created a bootstrap node as part of the deployment procedure. You can remove this node now that the OpenShift Container Platform cluster is up and running. Perform these steps as user ansible.

To convert the bootstrap node to a worker node:

1. Modify the hosts file to delete the entries under `bootstrap_node`. The following example shows the entries that you must remove:

```
# Provide bootstrap node details below
bootstrap_node:
      - name: bootstrap
        mac: 3C:FD:FE:B8:DD:00
        ip: 100.82.46.26
```

2. To remove the references to the bootstrap node from haproxy, dhcp, dns, pxe:menu, and so on, run the following command:

```
ansible-playbook -i hosts ocp.yml
```

Adding a worker node

Perform the following tasks to add a worker (compute) node. Unless directed otherwise, perform the tasks as user ansible.

To add a node as worker3 (because there are already two worker nodes):

1. Obtain the MAC and IP address of the node.
2. Add this information in the `worker nodes` section of the `hosts` file, as shown in the following example:

```
worker_nodes:
      - name: worker1
        mac: 3C:FD:FE:B8:DD:10
        ip: 100.82.46.21
      - name: worker2
        mac: 3C:FD:FE:B8:E1:A0
        ip: 100.82.46.22
      - name: worker3
        mac: 3C:FD:FE:B8:DD:00
        ip: 100.82.46.23
```

3. Run the playbooks to ensure that the worker3 node details are added to the dhcpd, haproxy, dns and pxe configuration files.

```
ansible-playbook -i hosts ocp.yml
```

4. Connect to the iDRAC of the worker node and open the virtual console.
5. Power on the worker node.

6. To ensure that the `ens2f0` interface is set for PXE boot:
   a. Press F2 to enter System Setup.
   b. Select Device Settings > NIC in Slot 2 Port 1 > NIC Configuration.
   c. From the Legacy Boot Protocol menu, select PXE.
   d. Select Finish to go back to System Setup.
   e. Select System BIOS > Network Settings.
   f. Under UEFI PXE Settings, select PXE Device1 Settings.
   g. From the Interface menu, select NIC in Slot2 Port1 Partition 1.
   h. Save the changes and reboot the node.

   When the system boots, it automatically boots into the PXE network, as shown in the following figure:

Figure 9. iDRAC console: PXE menu

7. Select worker3. Let the system reboot after installation.
8. After the node has restarted, ensure that the hostname and IP address are aligned, as shown in the following figure:

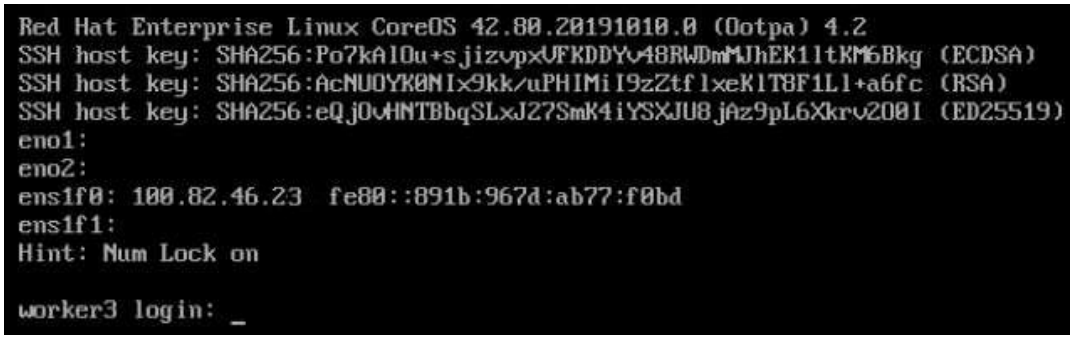

Figure 10. iDRAC console: Worker node

After the installation is complete, the node reboots to fetch the worker configuration file.

9. Ensure that the certificates are approved for the new worker node. As user **core**, run the following command in the CSAH node:

```
oc get csr -ojson | jq -r '.items[] | select(.status ==
{} ) | .metadata.name' | xargs oc adm certificate approve
certificatesigningrequest.certificates.k8s.io/csr-2jsd9
approved
```

```
certificatesigningrequest.certificates.k8s.io/csr-7cv46
approved
certificatesigningrequest.certificates.k8s.io/csr-7wslk
approved
certificatesigningrequest.certificates.k8s.io/csr-bpdzf
approved
certificatesigningrequest.certificates.k8s.io/csr-pzllb
approved
certificatesigningrequest.certificates.k8s.io/csr-rh5zs
approved
certificatesigningrequest.certificates.k8s.io/csr-z9m5w
approved
```

10. Verify that the new node is listed by running the following command as user core:

```
oc get nodes
NAME                          STATUS   ROLES    AGE
VERSION
etcd-0.ocp.example.com    Ready    master   2d
v1.14.6+c07e432da
etcd-1.ocp.example.com    Ready    master   2d
v1.14.6+c07e432da
etcd-2.ocp.example.com    Ready    master   2d
v1.14.6+c07e432da
worker1.ocp.example.com   Ready    worker   47h
v1.14.6+c07e432da
worker2.ocp.example.com   Ready    worker   47h
v1.14.6+c07e432da
worker3.ocp.example.com   Ready    worker   4m18s
v1.14.6+c07e432da
```

Chapter 6 Deploying Applications

This chapter presents the following topics:

Overview

This chapter provides examples of how to deploy applications in an OpenShift cluster. For more information, see the Red Hat Applications document.

Deploying images

OpenShift supports applications deployment using an image that is stored in an external image registry. Images have the necessary packages and program tools to run the applications by default.

To deploy an application that is already part of an image:

11. Log in to the OpenShift cluster:

    ```
    oc login -u <user name>
    ```

12. Create a new project by running:

    ```
    oc new-project <project name>
    ```

13. Create a new application by running:

    ```
    oc new-app <image-name>
    ```

    This guide uses `openshift/hello-openshift` for the image name.

14. After the image is deployed, you can identify all the objects that are created as part of the deployment by running the `oc get all` command. The following figure shows the command output:

```
[core@csah yaml]$ oc get all
NAME                           READY   STATUS              RESTARTS   AGE
pod/hello-openshift-1-deploy   1/1     Running             0          13s
pod/hello-openshift-1-f9gtr    0/1     ContainerCreating   0          6s

NAME                                     DESIRED   CURRENT   READY   AGE
replicationcontroller/hello-openshift-1   1         1         0       14s

NAME                      TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)             AGE
service/hello-openshift   ClusterIP   172.30.170.96   <none>        8080/TCP,8888/TCP   15s

NAME                                                REVISION   DESIRED   CURRENT   TRIGGERED BY
deploymentconfig.apps.openshift.io/hello-openshift   1          1         1         config,image(hello-openshift:latest)

NAME                                         IMAGE REPOSITORY                                                                  TAGS     UPDA
TED
imagestream.image.openshift.io/hello-openshift   image-registry.openshift-image-registry.svc:5000/image-deployment/hello-openshift   latest   14 s
econds ago
```

Figure 11. Sample application deployment status

Deploying S2I

OpenShift supports application deployment by using a source from GitHub and specifying an image. A build configuration file is generated for the S2I deployment in a new pod called Build Pod. In the build configuration file, you can configure the triggers needed to automate the new build process every time a condition meets the specifications you defined. After the deployment is complete, a new image with injected source code is created automatically.

Perform the following steps to deploy an application. The sample deployment uses httpd-ex as the GitHub source.

1. Log in to the OpenShift cluster:

   ```
   oc login -u <user name>
   ```

2. Create a new project by running:

   ```
   oc new-project <project name>
   ```

3. Use the GitHub source and specify the image of which the application will be a part to create the application:

   ```
   oc new-app centos/httpd-24-centos7~https://github.com/sclorg/httpd-ex.git
   ```

   Note: The image is **centos/httpd-24-centos7** and the GitHub source is `https://github.com/sclorg/httpd-ex.git`. You can obtain build logs by running `oc logs -f bc/httpd-ex` for this example.

4. After the image is deployed, identify all the objects that were created as part of the deployment by running the `oc get all` command.

   The following output is displayed:

```
[core@csah yaml]$ oc get all
NAME                        READY   STATUS      RESTARTS   AGE
pod/httpd-ex-1-5pnl7        1/1     Running     0          61s
pod/httpd-ex-1-build        0/1     Completed   0          106s
pod/httpd-ex-1-deploy       0/1     Completed   0          70s

NAME                                  DESIRED   CURRENT   READY   AGE
replicationcontroller/httpd-ex-1      1         1         1       70s

NAME                TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)             AGE
service/httpd-ex    ClusterIP   172.30.234.191   <none>        8080/TCP,8443/TCP   107s

NAME                                          REVISION   DESIRED   CURRENT   TRIGGERED BY
deploymentconfig.apps.openshift.io/httpd-ex   1          1         1         config,image(httpd-ex:latest)

NAME                                    TYPE     FROM   LATEST
buildconfig.build.openshift.io/httpd-ex   Source   Git    1

NAME                                  TYPE     FROM          STATUS     STARTED              DURATION
build.build.openshift.io/httpd-ex-1   Source   Git@0ac6da9   Complete   About a minute ago   37s

NAME                                                  IMAGE REPOSITORY                                                                TAGS     UPDATED
imagestream.image.openshift.io/httpd-24-centos7   image-registry.openshift-image-registry.svc:5000/s2i/httpd-24-centos7   latest   About a minute
ago
imagestream.image.openshift.io/httpd-ex           image-registry.openshift-image-registry.svc:5000/s2i/httpd-ex           latest   About a minute
ago
```

Figure 12. Sample S2I deployment status

5. Obtain triggers for this deployment by checking the YAML template of the build config:

   ```
   oc get buildconfig httpd-ex -o yaml
   ```

Application routes

To access deployed applications in the OpenShift cluster that are using images or source code from GitHub, you can use the service IP that is associated to the deployments. External access to the applications is not available by default.

To enable access to the applications from an external network:

1. Log in to the OpenShift cluster:

```
oc login -u <user name>
```

2. Switch to the project under which the application is running:

```
oc project <project name>
Now using project "<project name>" on server
"https://api.ocp.example.com:6443".
```

3. Identify the service that is associated with the application:

```
oc get svc
NAME              TYPE         CLUSTER-IP      EXTERNAL-IP
PORT(S)             AGE
hello-openshift   ClusterIP   172.30.92.106   <none>
8080/TCP,8888/TCP   23h
```

Note: Typically, the name of the service is the same as the name of the deployment.

4. Expose the route for service of your application:

```
oc expose svc/hello-openshift
route.route.openshift.io/hello-openshift exposed
```

5. Obtain the routes that were created:

```
oc get routes
NAME              HOST/PORT
PATH   SERVICES          PORT       TERMINATION
WILDCARD
hello-openshift   hello-openshift-
ocp42.apps.ocp.example.com          hello-openshift
8080-tcp
```

6. Open a browser, specify the content under HOST/PORT, and press Enter.

Note: The URL to connect to the application is `hello-openshift-ocp42.apps.ocp.example.com`.

7. Repeat the preceding steps to expose the service for the S2I deployment.

Application scaling

Applications are designed and created to meet the demands of customers. They can be scaled up or down based on business needs.

Perform the following steps to scale an application. This example uses `hello-openshift`.

1. Log in to the OpenShift cluster:

```
oc login -u <user name>
```

2. Switch to the project under which application is running:

```
oc project <project name>
Now using project "<project name>" on server
"https://api.ocp.example.com:6443".
```

3. Identify the deployment configuration that is associated with the application:

```
oc get dc
NAME              REVISION   DESIRED   CURRENT
TRIGGERED BY
hello-openshift   1                  1                  1
config,image(hello-openshift:latest)
```

4. Increase the desired count to 3 by running the following command:

```
oc scale --replicas=3 dc/hello-openshift
deploymentconfig.apps.openshift.io/hello-openshift scaled

oc get dc
NAME              REVISION   DESIRED   CURRENT
TRIGGERED BY
hello-openshift   1          3         3
config,image(hello-openshift:latest)
```

Note: OpenShift Container Platform supports the autoscaling of pods if cluster metrics are installed. After installing cluster metrics, run `oc autoscale dc/hello-openshift -min=1 --min=10 --cpu-percent=80`. The cluster metrics feature is under **Technical Preview**. For more information, see the Red Hat Custom Metrics documentation.

Chapter 7 Provisioning Storage

This chapter presents the following topics:

Overview

Administrators of OpenShift Container Platform clusters can map storage to containers. For more information, see Types of PVs. This chapter describes how to use working NFS/iSCSI storage to create a PV, claim the PV, and map the storage claim to the pod.

Prerequisites for using NFS

Before you start, ensure that:

- OpenShift Container Platform 4.2 cluster is up and running
- NFS server is set up, as described in NFS setup
- Worker nodes in the cluster can reach the NFS server and access the NFS share

---

Note: Run the steps for PV and Persistent Volume Claim (PVC) as user core unless directed otherwise.

Creating a PV using NFS

To create a PV:

5. Gather the following information:
   - NFS server IP or hostname
   - Path to the share
   - Storage capacity of the NFS share
6. Create an `nfspv.yaml` file using the following code. Modify the values of the variables defined between <> as necessary.

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: <nfs pv name>
spec:
  capacity:
    storage: <capacity>
  accessModes:
  - ReadWriteOnce
  nfs:
    path: <nfs share path>
    server: <nfs server ip or hostname>
```

7. Create the PV by running:

```
oc apply -f nfspv.yaml
```

8. Verify that the PV exists by running:

```
oc get pv nfspv
```

```
NAME    CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS
STORAGECLASS   REASON   AGE
nfspv   50Gi       RWO            Retain
Available                                  36m
```

Creating a PVC using NFS

To create a PVC using NFS:

1. Create an `nfspvc.yaml` file using the following content:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: nfspvc
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
storage: 50Gi
```

2. Create a PVC using the PV you created in the preceding step:

```
oc apply -f nfspvc.yaml
```

Note: A PVC created using the YAML file looks for an available PV. When it finds a PV that meets the needs of the claim, the PV and the PVC bind.

3. Verify that the PVC is created and that the status of PV is bound to the newly created PVC:

```
oc get pv nfspv
NAME    CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS
CLAIM            STORAGECLASS   REASON   AGE
nfspv   50Gi       RWO            Retain           Bound
default/nfspvc                           42m

oc get pvc
NAME     STATUS   VOLUME   CAPACITY   ACCESS MODES
STORAGECLASS   AGE
nfspvc   Bound    nfspv    50Gi       RWO
16s
```

Using iSCSI LUN

Before you start using iSCSI LUN, ensure that:

- OCP 4.2 cluster is up and running
- iSCSI server is set up (see generic iSCSI setup)
- Worker nodes in the cluster can reach the iSCSI server and access the iSCSI LUN

Creating a PV using iSCSI LUN

To create a PV:

1. Gather the following details:
   - iSCSI server IP or hostname
   - Target iqn—To obtain `iqn`, run `targetcli` as root in a Red Hat Enterprise Linux 7 or 8 system and then run `ls /iscsi`. The first iSCSI value is `iqn`.
   - Set the type to ext4.

```
targetcli
targetcli shell version 2.1.fb49
Copyright 2011-2013 by Datera, Inc and others.
For help on commands, type 'help'.

/> ls /iscsi
o- iscsi
.....................................................
.....................................................
[Targets: 1]
  o- iqn.2003-01.org.linux-
iscsi.nfs.x8664:sn.aa06c8c9ac41 ............. [TPGs: 1]
```

2. Create an `iscsipv.yaml` file with the following content. Modify the values of the variables in <> as necessary:

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: <iscsi pv name>
spec:
  capacity:
    storage: <capacity>
  accessModes:
  - ReadWriteOnce
  iscsi:
    targetPortal: <ip address of iscsi server>:3260
    iqn: <target iqn of the iscsi server>
    lun: 0
    fsType: <file system type>
```

```
    readOnly: false
```

3. Create a PV by running the following command:

```
oc apply -f iscsipv.yaml
```

4. Verify that the PV exists:

```
 oc get pv
NAME                 CAPACITY   ACCESS MODES   RECLAIM
POLICY   STATUS      CLAIM
STORAGECLASS   REASON   AGE
iscsipv              10Gi       RWO            Retain
Available
```

Creating a PVC using iSCSI

1. Create an `iscsipvc.yaml` file using the following content:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: iscsipvc
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 25Gi
```

2. Create a PVC using the PV you created in the preceding step:

```
oc apply -f iscsipvc.yaml
```

Note: A PVC created using the YAML file defined in Step 1 looks for PVs. When it finds an available PV that meets the claim needs, the PVC and the PV bind.

3. Confirm that the PVC is created and that the PV status is bound to the newly created PVC:

```
oc get pv iscsipv
NAME      CAPACITY   ACCESS MODES   RECLAIM POLICY
STATUS   CLAIM               STORAGECLASS   REASON   AGE
iscsipv   25Gi       RWO            Retain
Bound    default/iscsipvc                           46m

oc get pvc iscsipvc
NAME       STATUS   VOLUME    CAPACITY   ACCESS MODES
STORAGECLASS   AGE
iscsipvc   Bound    iscsipv   25Gi       RWO
46m
```

Creating a pod using NFS PVC

To create a pod using NFS PVC:

1. Create a file using the following sample YAML file:

```
apiVersion: v1
kind: Pod
metadata:
  name: <pod name>
spec:
  containers:
  - name: <container name>
    image: <image name>
    volumeMounts:
    - mountPath: "<mount point>"
      name: <volume name>
  volumes:
  - name: <volume name>
    persistentVolumeClaim:
      claimName: <nfs persistent volume claim>
```

   We used the following `nfspod.yaml` file:

```
apiVersion: v1
kind: Pod
metadata:
  name: nfspod
spec:
  containers:
  - name: myfrontend
    image: nginx
    volumeMounts:
    - mountPath: "/var/www/html"
      name: nfsshare
  volumes:
  - name: nfsshare
    persistentVolumeClaim:
      claimName: nfspvc
```

2. Create the pod by using the YAML file:

```
oc apply -f nfspod.yaml
```

3. Verify that the pod is created and that the NFS share is used for the mount point:

```
oc exec -it nfspod -- df -h /var/www/html
Filesystem              Size  Used Avail Use% Mounted on
100.82.46.61:/nfspv      50G  4.1G   46G   9% /var/www/html
```

   100.82.46.61 is the NFS server.

Creating a pod using an iSCSI PVC

To create a pod using an iSCSI PVC:

1. Create a file using this sample YAML file content. Modify the values for your environment:

```
apiVersion: v1
kind: Pod
metadata:
  name: <pod name>
spec:
  containers:
  - name: <container name>
    image: <image name>
    volumeMounts:
    - mountPath: "<mount point>"
      name: <volume name>
  volumes:
  - name: <volume name>
    persistentVolumeClaim:
      claimName: <iscsi persistent volume claim>
```

   We used the `iscsipod.yaml` YAML file:

```
apiVersion: v1
kind: Pod
metadata:
  name: iscsipod
spec:
  containers:
  - name: iscsi
    image: nginx
    volumeMounts:
    - mountPath: "/var/www/html"
      name: iscsivol
  volumes:
  - name: iscsivol
    persistentVolumeClaim:
      claimName: iscsipvc
```

2. Create the pod using the YAML file:

```
oc apply -f iscsipod.yaml
```

3. Verify that the pod is created and ensure that the iSCSI LUN is used for the mount point:

```
oc get pod iscsipod
NAME       READY   STATUS    RESTARTS   AGE
iscsipod   1/1     Running   0          7m21s
oc exec -it iscsipod -- df -h /var/www/html
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb         25G   45M   25G   1% /var/www/html
```

Chapter 8 Monitoring the Cluster

This chapter presents the following topics:

Introduction

By default, OpenShift Container Platform includes a monitoring cluster operator that is based on the Prometheus open source project. Multiple pods run in the cluster to monitor the state of the cluster and raise any alerts immediately in the OpenShift web console. You can create dashboards using Grafana pods by collecting all the necessary cluster metrics.

For more information, see the Red Hat document About cluster monitoring.

Viewing the Grafana dashboard

Note: Unless directed otherwise, run the following commands as user core.

To view dashboards using Grafana pods:

1. Log in to the CSAH node.
2. Obtain the Grafana route by running the following command:

```
oc get routes --all-namespaces | grep -i grafana
openshift-monitoring       grafana                    grafana-
openshift-monitoring.apps.ocp.example.com
grafana                    https    reencrypt/Redirect     None
```

3. Open a browser and enter the URL you obtained. In our sample output, the URL is: **grafana-openshift-monitoring.apps.ocp.example.com**
4. Log in using kubeadmin credentials or as an AD user.

   A list of the available components in the cluster is displayed.

5. Click one of the lists that is designated as `etcd`.

   The dashboard shows the active streams, the number of `etcd` nodes that are running, and other details, as shown in the following figure:

Figure 13. Grafana dashboard

Viewing alerts

To view the alerts in the OpenShift web console:

1. Log in to the CSAH node.
2. Get the Alert Manager route by running:

```
oc get routes --all-namespaces | grep -I alertmanager
openshift-monitoring       alertmanager-main
alertmanager-main-openshift-
monitoring.apps.ocp.example.com          alertmanager-
main     web     reencrypt/Redirect     None
```

3. Open a browser and enter the URL. In our sample output, the URL is:

   `alertmanager-main-openshift-monitoring.apps.ocp.example.com`

4. Log in as `kubeadmin` or an AD user.

Note: It is possible to silence an existing alert, temporarily muting notifications. For more information, see Silencing Alerts.

Viewing cluster metrics

To view cluster metrics in the OpenShift web console:

1. Log in to the CSAH node.
2. Obtain the cluster metrics route by running the following command:

```
oc get routes --all-namespaces | grep -i prometheus
openshift-monitoring       prometheus-k8s
prometheus-k8s-openshift-monitoring.apps.ocp.example.com
prometheus-k8s         web     reencrypt/Redirect     None
```

3. Open a browser and enter the URL. In our sample output, the URL is:
   `prometheus-k8s-openshift-monitoring.apps.ocp.example.com`
4. Log in as `kubeadmin` or an AD user.
5. From the **Execute** drop-down menu, select one of the **available queries** and click **Execute**.

   A graph is displayed for the selected query

Chapter 9 References

This chapter presents the following topics:

Dell Technologies documentation

The following Dell Technologies documentation provides additional and relevant information. Access to these documents depends on your login credentials. If you do not have access to a document, contact your Dell Technologies representative.

- Dell EMC InfoHub for Red Hat OpenShift Container Platform
- Dell EMC Ready Stack Converged Infrastructure
- Dell EMC PowerEdge R640 Technical Guide
- Dell EMC PowerEdge R740 and R740xd Technical Guide

Red Hat documentation

The following Red Hat resources provide additional relevant information:

- OpenShift Container Platform 4.2 Documentation
- Understanding the Operator Lifecycle Manager
- Red Hat Container Security Guide
- Understanding Red Hat OpenShift Service Mesh
- About cluster monitoring
- About Metering
- Silencing Alerts

Other resources

The following resources provide additional relevant information:

- Intel Xeon Gold Processors
- Kubernetes Guideposts for 2019
- Kubeflow: The Machine Learning Toolkit for Kubernetes