Configuring HyperFlex FlexVolume Storage Integration for RedHat OpenShift Container Platform

Support Matrix for HX FlexVolume Integration with OCP

The following table summarizes the Red Hat OpenShift Container Platform (OCP) software versions that are supported with each of the HX Data Platform software versions.

Table 1: Support Matrix for HX FlexVolume Integration with Red Hat OCP

HX Data Platform Version	Red Hat OCP Version 3.7	Red Hat OCP Version 3.9	Red Hat OCP Version 3.10	Red Hat OCP Version 3.11	Red Hat OCP Version 3.12	Red Hat OCP Version 3.13
3.0(1a) or later	Not Supported	Not Supported	—	—	—	—
3.5(1a) or later	—	Supported	Supported	—	—	—
3.5(2a) or later	—	Planned	Planned	—	—	—

HX Data Platform Version	Red Hat OCP Version 3.7	Red Hat OCP Version 3.9	Red Hat OCP Version 3.10	Red Hat OCP Version 3.11	Red Hat OCP Version 3.12	Red Hat OCP Version 3.13
4.0(1a) or later	—	—	Planned	Planned	—	—
4.1(1a) or later	—	—	Planned	Planned	TBD	TBD

Prerequisites

The following prerequisites must be met before configuring HyperFlex FlexVolume Storage Integration for RedHat OpenShift Container Platform.

- Cisco HyperFlex cluster is installed and running 3.5(1a) or later
- RedHat OpenShift Container Platform installed and running version 3.9 or later
- Downloaded the latest HyperFlex Kubernetes bundle (zip) file from the HyperFlex HX Data Platform section of Cisco Software Downloads.

Setting Up an Administrator Host

In the context of this document, the administrator host refers to a Linux-based host used for remotely administering the OpenShift cluster. This document does not dictate which Linux distribution should be used for the administrator host operating system, however some commands may vary slightly based on the distribution that is used. The administrator host may be a newly deployed host, or it can also be an existing host in the environment. The server used for the automated installation of the OpenShift cluster using Ansible, also known as the "bastion" node, makes a good candidate to use as the administrator host as it typically already meets most of the required prerequisites, such as OpenShift node connectivity, password-less SSH access, and so on. The examples in the following sections use the "bastion" node as the administrator host.

Important Perform the following steps on the administrator host.

Step 1 Ensure the Kubernetes command-line toolset **oc** is installed and configured to manage the target OpenShift cluster. If the toolset is not installed, you can find the procedure based on Linux distribution here: https://kubernetes.io/docs/tasks/tools/install-oc/#install-oc.

Step 2 Ensure an SSH (public and private) keypair has been generated. The SSH keypair is used to manage the remote OpenShift cluster. Ensure that password-less SSH authentication works between the administrator host and all OpenShift cluster nodes.

Step 3 Download the latest HyperFlex Kubernetes bundle (zip) file from the HyperFlex HX Data Platform section of Cisco Software Downloads.. Transfer the HyperFlex Kubernetes bundle (zip) file to the administrator host using any preferred method, such as **scp**. The remainder of this document assumes the HyperFlex Kubernetes bundle (zip) file has been copied to the following directory path `~/hxkube` on the administrator host.

Note By default, the **~/hxkube** directory does not exist and will need to be created.

Step 4 Unzip the HyperFlex Kubernetes bundle (zip) file to the **~/hxkube** directory on the administrator host. You may need to install the **unzip** package using a package manager (for example, **yum** or **apt-get**) based on the administrator host's Linux distribution.

Command Execution

The sections in this chapter are related to OpenShift and require that some commands be repeated across multiple nodes in the OpenShift cluster. You may manually run each command on all required OpenShift nodes, however this is highly repetitive. It is recommended to leverage a "while" loop in order to iterate through a list of all OpenShift nodes and execute the required commands.

Example: Using a "while" loop and a text file containing a list of OpenShift nodes

Create a file containing the IP addresses or hostnames of all OpenShift nodes.

```
administrator-host:~/hxkube$ vi ./ocp_nodes.txt
ocp-master
ocp-infra1
ocp-infra2
ocp-node1
ocp-node2
```

Iterate through the list of OpenShift nodes and run the command on each node.

```
administrator-host:~/hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n <ocpuser>@$host <command>; \
done

administrator-host:~/hxkube$
```

Note Hostnames or IP addresses can be used for each OpenShift node in the text file for the "while" loop.

Note Pay close attention to which commands should be run on which nodes, not all commands are run on all nodes.

Deploying RedHat OpenShift Container Platform

RedHat provides Ansible playbooks as a standard mechanism for automating the installation of a RedHat OpenShift Container Platform cluster. Extensive documentation and information can be found on the RedHat website for installing and configuring an OpenShift cluster using Ansible. The subsequent sections assume a running instance of RedHat OpenShift Container Platform exists or has been installed using standard RedHat methods and best practices, as found on the RedHat website.

Important Add an additional virtual machine network interface for each OpenShift node and attach it to the `k8-priv-iscsivm-network` VMware port-group. This interface is required to use the HyperFlex FlexVolume Storage Integration for OpenShift.

Note The additional interface can be added during OpenShift node deployment or can be added after deployment by editing the VMware virtual machine settings, as long as the additional interface exists prior to moving forward with the HyperFlex FlexVolume Storage Integration for OpenShift installation. There is no need to configure the added interface within the Operating System; this action will be done as part of the HyperFlex FlexVolume Storage Integration for OpenShift installation.

Distributing HyperFlex FlexVolume Software

In order to properly install and configure the HyperFlex FlexVolume Storage Integration for OpenShift, distribute the HyperFlex Kubernetes bundle (zip) file across all OpenShift cluster nodes. The following steps detail the process for distributing the HyperFlex Kubernetes bundle (zip) file to the appropriate hosts.

Step 1 Run the following command to create a directory named **hxkube** on each OpenShift cluster node.

Scope:

Run on all OpenShift nodes.

Command:

```
mkdir ~/hxkube
```

Example:

```
administrator-host:~/hxkube$
cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n @$host sudo mkdir ~/hxkube; \

done
administrator-host:~/hxkube$
```

Step 2 Copy the HX Kubernetes archive file to the `~/hxkube` directory on each OpenShift cluster node.

Scope:

Run on administrator-host, copying to all OpenShift nodes.

Command:

```
scp ~/hxkube/HX-Kubernetes-X.X.XXX.XXX.XXXXXXXX.XXXX.zip <ocpuser>@<remote-host>:<path>
```

Example:

```
administrator-host:hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do scp ~/hxkube/HX-Kubernetes-1.0.284.git.4022e8ec.hx35.zip $host:~/hxkube; \
done

HX-Kubernetes-1.0.284.git.4022e8ec.hx35.zip
                                                                       100%   37MB  74.7MB/s   00:00
```

```
HX-Kubernetes-1.0.284.git.4022e8ec.hx35.zip
                                                                      100%   37MB  85.5MB/s   00:00
HX-Kubernetes-1.0.284.git.4022e8ec.hx35.zip
                                                                      100%   37MB  81.4MB/s   00:00
HX-Kubernetes-1.0.284.git.4022e8ec.hx35.zip
                                                                      100%   37MB  82.0MB/s   00:00
HX-Kubernetes-1.0.284.git.4022e8ec.hx35.zip
                                                                      100%   37MB  75.4MB/s   00:00
administrator-host:hxkube$
```

Step 3 Install the `unzip` package on each OpenShift cluster node using the `yum` package manager.

Scope:

Run on all OpenShift nodes.

Commands:

```
sudo yum install -y unzip
```

Example:

```
administrator-host:hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo yum install -y unzip; \
done

Loaded plugins: product-id, search-disabled-repos, subscription-manager
Resolving Dependencies
--> Running transaction check
---> Package unzip.x86_64 0:6.0-19.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package      Arch          Version             Repository                 Size
================================================================================
Installing:
 unzip        x86_64        6.0-19.el7          rhel-7-server-rpms        170 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 170 k
Installed size: 365 k
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : unzip-6.0-19.el7.x86_64                                      1/1
  Verifying  : unzip-6.0-19.el7.x86_64                                      1/1

Installed:
  unzip.x86_64 0:6.0-19.el7

Complete!
Loaded plugins: product-id, search-disabled-repos, subscription-manager
Resolving Dependencies
--> Running transaction check
---> Package unzip.x86_64 0:6.0-19.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved
```

```
================================================================================
 Package      Arch          Version             Repository                 Size
================================================================================
Installing:
 unzip        x86_64        6.0-19.el7          rhel-7-server-rpms        170 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 170 k
Installed size: 365 k
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : unzip-6.0-19.el7.x86_64                                      1/1
  Verifying  : unzip-6.0-19.el7.x86_64                                      1/1

Installed:
  unzip.x86_64 0:6.0-19.el7

Complete!
Loaded plugins: product-id, search-disabled-repos, subscription-manager
Resolving Dependencies
--> Running transaction check
---> Package unzip.x86_64 0:6.0-19.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package      Arch          Version             Repository                 Size
================================================================================
Installing:
 unzip        x86_64        6.0-19.el7          rhel-7-server-rpms        170 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 170 k
Installed size: 365 k
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : unzip-6.0-19.el7.x86_64                                      1/1
  Verifying  : unzip-6.0-19.el7.x86_64                                      1/1

Installed:
  unzip.x86_64 0:6.0-19.el7

Complete!
Loaded plugins: product-id, search-disabled-repos, subscription-manager
Resolving Dependencies
--> Running transaction check
---> Package unzip.x86_64 0:6.0-19.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved
```

```
================================================================================
 Package      Arch          Version             Repository                 Size
================================================================================
Installing:
 unzip        x86_64        6.0-19.el7          rhel-7-server-rpms        170 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 170 k
Installed size: 365 k
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : unzip-6.0-19.el7.x86_64                                      1/1
  Verifying  : unzip-6.0-19.el7.x86_64                                      1/1

Installed:
  unzip.x86_64 0:6.0-19.el7

Complete!
Loaded plugins: product-id, search-disabled-repos, subscription-manager
Resolving Dependencies
--> Running transaction check
---> Package unzip.x86_64 0:6.0-19.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package      Arch          Version             Repository                 Size
================================================================================
Installing:
 unzip        x86_64        6.0-19.el7          rhel-7-server-rpms        170 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 170 k
Installed size: 365 k
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : unzip-6.0-19.el7.x86_64                                      1/1
  Verifying  : unzip-6.0-19.el7.x86_64                                      1/1

Installed:
  unzip.x86_64 0:6.0-19.el7

Complete!
administrator-host:hxkube$
```

Step 4 Unzip the Kubernetes bundle (zip) file on each OpenShift cluster node.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo unzip ~/hxkube/HX-Kubernetes-X.X.XXX.XXX.XXXXXXXX.XXXX.zip -d ~/hxkube
```

Example:

```
administrator-host:hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo unzip ~/hxkube/HX-Kubernetes-1.0.284.git.4022e8ec.hx35.zip -d ~/hxkube; \
done

Archive:  /root/hxkube/HX-Kubernetes-1.0.284.git.4022e8ec.hx35.zip
 extracting: /root/hxkube/RELEASE.txt
 extracting: /root/hxkube/hx-provisioner-setup
 extracting: /root/hxkube/hx-provisioner.tar.gz
 extracting: /root/hxkube/hxkube-collect-logs
 extracting: /root/hxkube/hxprovisioner-deploy.yaml
 extracting: /root/hxkube/hxvolume
 extracting: /root/hxkube/hxvolume-plugin-1.0.284.git.4022e8ec.hx35-1.x86_64.rpm
 extracting: /root/hxkube/hxvolume-plugin_1.0.284.git.4022e8ec.hx35_amd64.deb
 extracting: /root/hxkube/istgttool
Archive:  /root/hxkube/HX-Kubernetes-1.0.284.git.4022e8ec.hx35.zip
 extracting: /root/hxkube/RELEASE.txt
 extracting: /root/hxkube/hx-provisioner-setup
 extracting: /root/hxkube/hx-provisioner.tar.gz
 extracting: /root/hxkube/hxkube-collect-logs
 extracting: /root/hxkube/hxprovisioner-deploy.yaml
 extracting: /root/hxkube/hxvolume
 extracting: /root/hxkube/hxvolume-plugin-1.0.284.git.4022e8ec.hx35-1.x86_64.rpm
 extracting: /root/hxkube/hxvolume-plugin_1.0.284.git.4022e8ec.hx35_amd64.deb
 extracting: /root/hxkube/istgttool
Archive:  /root/hxkube/HX-Kubernetes-1.0.284.git.4022e8ec.hx35.zip
 extracting: /root/hxkube/RELEASE.txt
 extracting: /root/hxkube/hx-provisioner-setup
 extracting: /root/hxkube/hx-provisioner.tar.gz
 extracting: /root/hxkube/hxkube-collect-logs
 extracting: /root/hxkube/hxprovisioner-deploy.yaml
 extracting: /root/hxkube/hxvolume
 extracting: /root/hxkube/hxvolume-plugin-1.0.284.git.4022e8ec.hx35-1.x86_64.rpm
 extracting: /root/hxkube/hxvolume-plugin_1.0.284.git.4022e8ec.hx35_amd64.deb
 extracting: /root/hxkube/istgttool
Archive:  /root/hxkube/HX-Kubernetes-1.0.284.git.4022e8ec.hx35.zip
 extracting: /root/hxkube/RELEASE.txt
 extracting: /root/hxkube/hx-provisioner-setup
 extracting: /root/hxkube/hx-provisioner.tar.gz
 extracting: /root/hxkube/hxkube-collect-logs
 extracting: /root/hxkube/hxprovisioner-deploy.yaml
 extracting: /root/hxkube/hxvolume
 extracting: /root/hxkube/hxvolume-plugin-1.0.284.git.4022e8ec.hx35-1.x86_64.rpm
 extracting: /root/hxkube/hxvolume-plugin_1.0.284.git.4022e8ec.hx35_amd64.deb
 extracting: /root/hxkube/istgttool
Archive:  /root/hxkube/HX-Kubernetes-1.0.284.git.4022e8ec.hx35.zip
 extracting: /root/hxkube/RELEASE.txt
 extracting: /root/hxkube/hx-provisioner-setup
 extracting: /root/hxkube/hx-provisioner.tar.gz
 extracting: /root/hxkube/hxkube-collect-logs
 extracting: /root/hxkube/hxprovisioner-deploy.yaml
 extracting: /root/hxkube/hxvolume
 extracting: /root/hxkube/hxvolume-plugin-1.0.284.git.4022e8ec.hx35-1.x86_64.rpm
 extracting: /root/hxkube/hxvolume-plugin_1.0.284.git.4022e8ec.hx35_amd64.deb
 extracting: /root/hxkube/istgttool
administrator-host:hxkube$
```

Managing HyperFlex FlexVolume Plug-in

Installing HyperFlex FlexVolume Plug-in

Step 1 The Ansible installation playbooks for OpenShift should by default install the `iscsi-initiator-utils` package. Verify that the package is installed.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo yum list installed iscsi-initiator-utils
```

Example:

```
administrator-host:hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo yum list installed iscsi-initiator-utils; \
done

Loaded plugins: product-id, search-disabled-repos, subscription-manager
Installed Packages
iscsi-initiator-utils.x86_64        6.2.0.874-10.el7         @rhel-7-server-rpms
Loaded plugins: product-id, search-disabled-repos, subscription-manager
Installed Packages
iscsi-initiator-utils.x86_64        6.2.0.874-10.el7         @rhel-7-server-rpms
Loaded plugins: product-id, search-disabled-repos, subscription-manager
Installed Packages
iscsi-initiator-utils.x86_64        6.2.0.874-10.el7         @rhel-7-server-rpms
Loaded plugins: product-id, search-disabled-repos, subscription-manager
Installed Packages
iscsi-initiator-utils.x86_64        6.2.0.874-10.el7         @rhel-7-server-rpms
Loaded plugins: product-id, search-disabled-repos, subscription-manager
Installed Packages
iscsi-initiator-utils.x86_64        6.2.0.874-10.el7         @rhel-7-server-rpms
administrator-host:hxkube$
```

Step 2 If, for some reason, the `iscsi-initiator-utils` package is not installed, install it using the yum package manager.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo yum install -y iscsi-initiator-utils
```

Example:

```
administrator-host:~/hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo yum install -y iscsi-initiator-utils; \
done

administrator-host:~/hxkube$
```

Step 3 Install the `avahi-autoipd` package using the yum package manager.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo yum install -y avahi-autoipd
```

Example:

```
administrator-host:hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo yum install -y avahi-autoipd; \
done

Loaded plugins: product-id, search-disabled-repos, subscription-manager
Resolving Dependencies
--> Running transaction check
---> Package avahi-autoipd.x86_64 0:0.6.31-19.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package           Arch       Version              Repository             Size
================================================================================
Installing:
 avahi-autoipd     x86_64     0.6.31-19.el7        rhel-7-server-rpms      40 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 40 k
Installed size: 44 k
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : avahi-autoipd-0.6.31-19.el7.x86_64                           1/1
  Verifying  : avahi-autoipd-0.6.31-19.el7.x86_64                           1/1

Installed:
  avahi-autoipd.x86_64 0:0.6.31-19.el7

Complete!
Loaded plugins: product-id, search-disabled-repos, subscription-manager
Resolving Dependencies
--> Running transaction check
---> Package avahi-autoipd.x86_64 0:0.6.31-19.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package           Arch       Version              Repository             Size
================================================================================
Installing:
 avahi-autoipd     x86_64     0.6.31-19.el7        rhel-7-server-rpms      40 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 40 k
```

```
Installed size: 44 k
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : avahi-autoipd-0.6.31-19.el7.x86_64                          1/1
  Verifying  : avahi-autoipd-0.6.31-19.el7.x86_64                          1/1

Installed:
  avahi-autoipd.x86_64 0:0.6.31-19.el7

Complete!
Loaded plugins: product-id, search-disabled-repos, subscription-manager
Resolving Dependencies
--> Running transaction check
---> Package avahi-autoipd.x86_64 0:0.6.31-19.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package           Arch       Version              Repository              Size
================================================================================
Installing:
 avahi-autoipd     x86_64     0.6.31-19.el7        rhel-7-server-rpms      40 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 40 k
Installed size: 44 k
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : avahi-autoipd-0.6.31-19.el7.x86_64                          1/1
  Verifying  : avahi-autoipd-0.6.31-19.el7.x86_64                          1/1

Installed:
  avahi-autoipd.x86_64 0:0.6.31-19.el7

Complete!
Loaded plugins: product-id, search-disabled-repos, subscription-manager
Resolving Dependencies
--> Running transaction check
---> Package avahi-autoipd.x86_64 0:0.6.31-19.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package           Arch       Version              Repository              Size
================================================================================
Installing:
 avahi-autoipd     x86_64     0.6.31-19.el7        rhel-7-server-rpms      40 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 40 k
```

```
Installed size: 44 k
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : avahi-autoipd-0.6.31-19.el7.x86_64                           1/1
  Verifying  : avahi-autoipd-0.6.31-19.el7.x86_64                           1/1

Installed:
  avahi-autoipd.x86_64 0:0.6.31-19.el7

Complete!
Loaded plugins: product-id, search-disabled-repos, subscription-manager
Resolving Dependencies
--> Running transaction check
---> Package avahi-autoipd.x86_64 0:0.6.31-19.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package            Arch       Version             Repository             Size
================================================================================
Installing:
 avahi-autoipd      x86_64     0.6.31-19.el7       rhel-7-server-rpms      40 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 40 k
Installed size: 44 k
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : avahi-autoipd-0.6.31-19.el7.x86_64                           1/1
  Verifying  : avahi-autoipd-0.6.31-19.el7.x86_64                           1/1

Installed:
  avahi-autoipd.x86_64 0:0.6.31-19.el7

Complete!
administrator-host:hxkube
```

Step 4 In the Deploying RedHat OpenShift Container Platform section, you were instructed to add an additional virtual machine network interface to each OpenShift node. The interface name of the added virtual machine network interface is now required in order to proceed with the installation.

Use the `ifconfig -a` command to find the name of the interface on one of the OpenShift nodes. The interface should not have an IP address assigned and it is recommended to cross reference the MAC address with VMware vCenter in order to ensure the correct interface. In this particular environment the added interface is named *ens224*.

Note The `ifconfig -a` command only needs to be run on a single OpenShift node.

Scope:

Run on a single OpenShift node.

Command:

```
sudo ifconfig -a
```

Example:

```
administrator-host:hxkube$ ssh ocp-master sudo ifconfig -a
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 0.0.0.0
        ether 02:42:e8:b1:c0:1e  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens192: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.2.17.150  netmask 255.255.255.0  broadcast 10.2.17.255
        ether 00:50:56:98:ad:b3  txqueuelen 1000  (Ethernet)
        RX packets 799305  bytes 992373512 (946.4 MiB)
        RX errors 0  dropped 73  overruns 0  frame 0
        TX packets 275804  bytes 142731217 (136.1 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens224: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether 00:50:56:98:05:fd  txqueuelen 1000  (Ethernet)
        RX packets 654  bytes 217464 (212.3 KiB)
        RX errors 0  dropped 2  overruns 0  frame 0
        TX packets 629  bytes 215118 (210.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 3120877  bytes 1376730600 (1.2 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3120877  bytes 1376730600 (1.2 GiB)
              TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
administrator-host:hxkube$
```

Step 5 Ensure there is no network configuration file (`/etc/sysconfig/network-scripts/ifcfg-<interface_name>`) for the additional virtual machine network interface that was added for the HyperFlex Storage Integration for OpenShift.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo rm /etc/sysconfig/network-scripts/ifcfg-<interface_name>
```

Example:

```
administrator-host:hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo rm /etc/sysconfig/network-scripts/ifcfg-ens224; \
done

administrator-host:hxkube$
```

Step 6 Restart the Network Manager service.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo systemctl restart NetworkManager.service
```

Example:

```
administrator-host:hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo systemctl restart NetworkManager.service; \
done
```

Step 7 Restart the Atomic OpenShift Node service.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo systemctl restart atomic-openshift-node.service
```

Example:

```
administrator-host:hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo systemctl restart atomic-openshift-node.service; \
done
```

Step 8 Install the `hxvolume-plugin` RPM package to update the existing HyperFlex FlexVolume plug-in on each OpenShift cluster node.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo rpm -i ~/hxkube/hxvolume-plugin_X.X.XXX.XXX.XXXXXXXX.XXXX.rpm
```

Example:

```
administrator-host:~/hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo rpm -i ~/hxkube/hxvolume-plugin-1.0.284.git.4022e8ec.hx35-1.x86_64.rpm; \
done

administrator-host:~/hxkube$
```

Step 9 Edit the `/etc/kubernetes/hxflexvolume.json` configuration file to change the target IP address from 169.254.1.1 to 169.254.254.1.

Command:

```
 sudo sed -i -e s/169.254.1.1/169.254.254.1/  /etc/kubernetes/hxflexvolume.json.example
```

Example:

```
administrator-host:~/hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \

do ssh -n $host sudo sed -i -e s/169.254.1.1/169.254.254.1/
/etc/kubernetes/hxflexvolume.json.example; \

done
```

Step 10 Create the HyperFlex FlexVolume Plugin configuration file. The file will be copied from an example file found in the `/etc/kubernetes/` directory.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo cp /etc/kubernetes/hxflexvolume.json.example /etc/kubernetes/hxflexvolume.json
```

Example:

```
administrator-host:~/hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo cp /etc/kubernetes/hxflexvolume.json.example /etc/kubernetes/hxflexvolume.json;
 \
done

administrator-host:~/hxkube$
```

Step 11 Edit the `/etc/kubernetes/hxflexvolume.json` configuration file and update with the appropriate name for the interface to be used for the HyperFlex Storage Integration for OpenShift. In the example in this document the interface is named *ens224*.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo sed -i -e s/ens[[:digit:]+]/<interface_name>/ /etc/kubernetes/hxflexvolume.json
```

Example:

```
administrator-host:hxkube$cat ~/hxkube/ocp_nodes.txt | while read host; \
dossh -n $host sudo sed -i -e s/ens[[:digit:]+]/ens224/ /etc/kubernetes/hxflexvolume.json;\
done

administrator-host:hxkube$
```

Step 12 Initialize the FlexVolume plugin.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo /usr/libexec/kubernetes/kubelet-plugins/volume/exec/hyperflex~hxvolume/hxvolume init
```

Example:

```
administrator-host:hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo /usr/libexec/kubernetes/kubelet-plugins/volume/exec/hyperflex~hxvolume/hxvolume
 init; \
done

{"capabilities":{"attach":false},"status":"Success"}
{"capabilities":{"attach":false},"status":"Success"}
{"capabilities":{"attach":false},"status":"Success"}
{"capabilities":{"attach":false},"status":"Success"}
{"capabilities":{"attach":false},"status":"Success"}
administrator-host:hxkube$
```

Step 13 Restart the Atomic Openshift Node service.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo systemctl restart atomic-openshift-node.service
```

Example:

```
administrator-host:hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo systemctl restart atomic-openshift-node.service; \
done

administrator-host:hxkube$
```

Checking HyperFlex FlexVolume Plug-in Version

Important The following steps must be performed on one OpenShift node only.

Step 1 Log into one of the OpenShift cluster nodes using SSH.

Step 2 Change directories to the `/usr/libexec/kubernetes/kubelet-plugins/volume/exec/hyperflex-hxvolume/` directory.

Scope:

Run on a single OpenShift node.

Command

```
cd /usr/libexec/kubernetes/kubelet-plugins/volume/exec/hyperflex~hxvolume/
```

Example:

```
ocpuser@openshift-master:~$ cd /usr/libexec/kubernetes/kubelet-plugins/volume/exec/hyperflex~hxvolume/
ocpuser@openshift-master:/usr/libexec/kubernetes/kubelet-plugins/volume/exec/hyperflex~hxvolume$
```

Step 3 Run the `hxvolume version` command as the root user (with sudo) to view the HyperFlex FlexVolume plug-in version.

Scope:

Run on a single OpenShift node.

Command:

```
sudo hxvolume version
```

Example:

```
ocpuser@openshift-master:/usr/libexec/kubernetes/kubelet-plugins/volume/exec/hyperflex~hxvolume$
sudo ./hxvolume version
hxvolume version: 1.0.284.git.4022e8ec.hx35
ocpuser@openshift-master:/usr/libexec/kubernetes/kubelet-plugins/volume/exec/hyperflex~hxvolume$
```

Updating HyperFlex FlexVolume Plug-in

The following steps must be performed from the administrator host.

Step 1 Install the `hxvolume-plugin` Debian package to update the existing HyperFlex FlexVolume plug-in on each OpenShift cluster node.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo rpm -i ~/hxkube/hxvolume-plugin_X.X.XXX.XXX.XXXXXXXX.XXXX.rpm
```

Example:

```
administrator-host:hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo rpm -i ~/hxkube/hxvolume-plugin-1.0.284.git.4022e8ec.hx35-1.x86_64.rpm; \
done

administrator-host:hxkube$
```

Step 2 Restart the Atomic OpenShift Node service.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo systemctl restart atomic-openshift-node.service
```

Example:

```
administrator-host:hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo systemctl restart atomic-openshift-node.service; \
done

administrator-host:hxkube$
```

Managing HyperFlex FlexVolume Provisioner

Installing HyperFlex FlexVolume Provisioner

Step 1 Load the HyperFlex FlexVolume Provisioner docker image on each OpenShift cluster node.

Scope:

Run on all OpenShift nodes.

Command

```
sudo docker image load --input ~/hxkube/hx-provisioner.tar.gz
```

Example

```
administrator-host:hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo docker image load -i ~/hxkube/hx-provisioner.tar.gz; \
done

Loaded image: hx-provisioner:1.0.284.git.4022e8ec.hx35
Loaded image: hx-provisioner:1.0.284.git.4022e8ec.hx35
Loaded image: hx-provisioner:1.0.284.git.4022e8ec.hx35
Loaded image: hx-provisioner:1.0.284.git.4022e8ec.hx35
Loaded image: hx-provisioner:1.0.284.git.4022e8ec.hx35
administrator-host:hxkube$
```

Step 2 Run the `hx-provisioner-setup` script to generate the required YAML file for deploying the HyperFlex Provisioner pod on the OpenShift cluster. Provide the following information as parameters when running the `hx-provisioner-setup` script.

Parameters:

- `-cluster-name`—Name of the OpenShift cluster (must be unique across the HyperFlex cluster).
- `-url`—URL to reach the HyperFlex API. This URL is equivalent to the https://<hyperFlex_cluster_management_IP_address.
- `-username`—The username that is used to authenticate to the HyperFlex cluster. Typically, a vCenter SSO account such as administrator@vsphere.local.
- `-password`—Name of the resulting output file generated by the `hx-provisioner-setup` script.

Scope:

Run on administrator host.

Commands:

```
hx-provisioner-setup -cluster-name <ocp_cluster_name> -url
https://<hx_cluster_mgmt_ip> -username
<hx_cluster_username> -output <output_file>.yml
```

Example:

```
administrator-host:hxkube$ ~/hxkube/hx-provisioner-setup -cluster-name tc1 -url https://172.0.13.32
 -username administrator@vsphere.local -output tc1.yml

password for [administrator@vsphere.local] at [https://172.0.13.32]: <password>
wrote config to tc1.yml

administrator-host:hxkube$
```

Step 3 Ensure the output file was created successfully. In this example, the output file was named `tc1.yml`.

Command:

```
ls ~/hxkube
```

Example:

```
administrator-host:hxkube$ ls -l ~/hxkube

total 76376
-rwxr-xr-x. 1 root root      371 Jan  1  2008 hxkube-collect-logs
-rw-r--r--. 1 root root 39078464 Nov  6 16:05 HX-Kubernetes-1.0.284.git.4022e8ec.hx35.zip
-rw-r--r--. 1 root root     2374 Jan  1  2008 hxprovisioner-deploy.yaml
-rwxr-xr-x. 1 root root  8193670 Jan  1  2008 hx-provisioner-setup
-rw-r--r--. 1 root root  7024699 Jan  1  2008 hx-provisioner.tar.gz
-rw-------. 1 root root     2356 Nov 16 13:26 hx-provisioner.yaml
-rwxr-xr-x. 1 root root  4540037 Jan  1  2008 hxvolume
-rw-r--r--. 1 root root  8807897 Jan  1  2008 hxvolume-plugin-1.0.284.git.4022e8ec.hx35-1.x86_64.rpm
-rw-r--r--. 1 root root  8726200 Jan  1  2008 hxvolume-plugin_1.0.284.git.4022e8ec.hx35_amd64.deb
-rw-r--r--. 1 root root  1780928 Jan  1  2008 istgttool
-rw-r--r--. 1 root root       53 Nov 18 21:57 ocp_nodes.txt
-rw-r--r--. 1 root root     1146 Jan  1  2008 RELEASE.txt
-rw-------. 1 root root     2356 Nov 27 11:31 tc1.yml
administrator-host:hxkube$
```

Step 4 Run the `oc create -f <file>` command to deploy the HyperFlex Provisioner pod.

Note This procedure schedules the HyperFlex Provisioner pod to run on one of the OpenShift cluster nodes nodes. While there is no technical reason for it, if you require to run the HyperFlex Provisioner pod on a specific OpenShift cluster node (perhaps the infra nodes or even the master node), edit the `<output>` file created by the `hx-provisioner-setup` script to include a `Toleration` to overcome to include a `Toleration` to overcome any configured taints. For more information see https://kubernetes.io/docs/concepts/configuration/taint-and-toleration/.

Scope:

Run on administrator host.

Command:

```
oc create -f ~/hxkube/<file>
```

Example:

```
administrator-host:hxkube$ oc create -f ~/hxkube/tc1.yml

secret "hxprovisioner" created
configmap "hxprovisioner-config" created
serviceaccount "hxprovisioner" created
clusterrolebinding.rbac.authorization.k8s.io "hxprovisioner-binding" created
deployment.apps "hx-provisioner" created
administrator-host:hxkube$
```

Step 5 Verify that the HyperFlex Provisioner pod is running.

Scope:

Run on administrator host.

Command:

```
oc get pods -n kube-system
```

Example:

```
administrator-host:hxkube$ oc get pods -n kube-system

NAME                             READY     STATUS    RESTARTS   AGE
hx-provisioner-cdf64fc5f-ghvrw   1/1       Running   0          10s
master-api-ocp-master            1/1       Running   0          18h
master-controllers-ocp-master    1/1       Running   0          18h
master-etcd-ocp-master           1/1       Running   0          18h
administrator-host:hxkube$
```

Checking HyperFlex FlexVolume Provisioner Version

Step 1 Run `et pods -n kube-system` command to get the complete name of the deployed HyperFlex FlexVolume Provisioner pod.

Scope:

Run on administrator host.

Command:

```
oc get pods -n kube-system
```

Example:

```
administrator-host:hxkube$ oc get pods -n kube-system

NAME                              READY     STATUS    RESTARTS   AGE
hx-provisioner-cdf64fc5f-ghvrw    1/1       Running   0          <invalid>
master-api-ocp-master             1/1       Running   0          18h
master-controllers-ocp-master     1/1       Running   0          18h
master-etcd-ocp-master            1/1       Running   0          18h
administrator-host:hxkube$
```

Step 2 Run `describe pods` command to get the complete details of the deployed HyperFlex FlexVolume Provisioner pod. Look for the `hx-provisioner` container image name which includes the version as a tag (that is, after the colon in the container name).

Scope:

Run on administrator host.

Command:

```
oc describe pods <pod_name> -n kube-system
```

Example:

```
administrator-host:hxkube$ oc describe pods hx-provisioner-cdf64fc5f-ghvrw -n kube-system

Name:            hx-provisioner-cdf64fc5f-ghvrw
Namespace:       kube-system
Node:            ocp-node2/10.2.17.154
Start Time:      Tue, 27 Nov 2018 11:43:55 -0500
Labels:          app=hx-provisioner
                 pod-template-hash=789209719
Annotations:     <none>
Status:          Running
IP:              10.128.2.2
Controlled By:   ReplicaSet/hx-provisioner-cdf64fc5f
Containers:
  hx-provisioner:
    Container ID:  docker://b5cf5c79a9fec30221ec1f27c09fe51c138a371e474854577f1743baad343f45
    Image:         hx-provisioner:1.0.284.git.4022e8ec.hx35
    Image ID:      docker://sha256:e275592fd478b06b4e60c40eb42fc2321a15546d6e4eb9a8b7aee818073852b0
    Port:          443/TCP
    Host Port:     0/TCP
    Args:
      -hxapi-url=$(HX_API_URL)
      -hxapi-token-file=/secrets/hxapi/token
      -hxapi-hxclusteruuid=$(HX_CLUSTERUUID)
    State:          Running
administrator-host:hxkube$
```

Updating HyperFlex FlexVolume Provisioner

Step 1 Download the latest HX Kubernetes release package from Cisco Software Downloads.

Step 2 Follow the steps in Distributing HyperFlex FlexVolume Software to copy the latest HX Kubernetes release package to each OpenShift node. Unzip the file on each OpenShift node.

Scope:

Run on all OpenShift nodes.

Command:

```
sudo docker image load --input ~/hxkube/hx-provisioner.tar.gz
```

Example:

```
administrator-host:hxkube$ cat ~/hxkube/ocp_nodes.txt | while read host; \
do ssh -n $host sudo docker image load -i ~/hxkube/hx-provisioner.tar.gz; \
done

Loaded image: hx-provisioner:1.0.284.git.4022e8ec.hx35
Loaded image: hx-provisioner:1.0.284.git.4022e8ec.hx35
Loaded image: hx-provisioner:1.0.284.git.4022e8ec.hx35
Loaded image: hx-provisioner:1.0.284.git.4022e8ec.hx35
Loaded image: hx-provisioner:1.0.284.git.4022e8ec.hx35
administrator-host:hxkube$
```

Step 3 Note the full name of the newly loaded Docker image from the output of the previous command.

Example:

```
Loaded image: hx-provisioner:1.0.284.git.4022e8ec.hx35
```

Step 4 Run the `oc set image` command to update the HyperFlex FlexVolume Provisioner container deployment to use the updated container image.

Scope:

Run on administrator host.

Command:

```
oc set image pod/<pod_name> hx-provisioner=<new_image_name>
```

Example:

```
administrator-host:hxkube$ oc set image pod/hx-provisioner-cdf64fc5f-ghvrw
hx-provisioner=hx-provisioner:1.0.284.git.4022e8ec.hx35 -n kube-system

administrator-host:hxkube$
```

Configuring Storage Classes

Step 1 Create a file called `hx-storageclass.yml` in the `~/hxkube` directory

Scope:

Run on administrator host.

Command:

```
sudo touch ~/hxkube/hx-storageclass.yml
```

Example:

```
administrator-host:hxkube$ touch ~/hxkube/hx-storageclass.yml

administrator-host:hxkube$
```

Step 2 Edit the `hx-storageclass.yml` file using your choice of file editor (for example, vi.) and insert the following text. Be sure to copy the text exactly as shown below, including indentations. Save the file once complete.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: hyperflex
  annotations:
    storageclass.kubernetes.io/is-default-class: "true"
provisioner: hyperflex.io/hxvolume
```

Note If you do not wish for the hyperflex storage class to be the default storage class, you can remove the following two lines from the above `hx-storageclass.yml` file before creating the storage class:

```
annotations:
    storageclass.kubernetes.io/is-default-class: "true"
```

Step 3 Use the `oc create -f` command to create the hyperflex storage class in the OpenShift cluster.

Scope:

Run on administrator host.

Command:

```
oc create -f <file>
```

Example:

```
administrator-host:hxkube$ oc create -f ./hx-storageclass.yml

storageclass.storage.k8s.io "hyperflex" created
administrator-host:hxkube$
```

Step 4 Use the `get sc` command to view the new hyperflex storage classes on the OpenShift cluster.

Scope:

Run on administrator host.

Command:

```
oc get sc
```

Example:

```
administrator-host:hxkube$ oc get sc

NAME                   PROVISIONER              AGE
hyperflex (default)    hyperflex.io/hxvolume    23s
administrator-host:hxkube$
```

Provisioning Persistent Volumes

At this point, the HyperFlex Storage Integration for OpenShift has been fully deployed and can now be leveraged to provide persistent storage to OpenShift workloads. Developers and users can now simply submit Persistent Volume Claim requests, and if the hyperflex storage class is configured as the default storage class, the requested storage will be automatically provisioned by HyperFlex and provided to the OpenShift environment. This results in a new Persistent Volume Claim bound to a new Persistent Volume object as well. The Persistent Volume Claim can then be used when deploying workloads within OpenShift.

Note When creating a Persistent Volume Claim request, the `storageClassName:hyperflex`line is required only if you decide not to create the `hyperflex` storage class as the default storage class. If the hyperflex storage class is the default storage class, you can remove that line from any Persistent Volume Claim requests.

The following steps provide a sample workflow of deploying a simple “Cisco Message Board” application using persistent storage from HyperFlex.

Step 1 Create a new project (namespace) for the sample application.

Scope:

Run on administrator host.

Command:

```
oc create namespace <name>
```

Example:

```
administrator-host:hxkube$ oc create namespace message-board

namespace "message-board" created
administrator-host:hxkube$
```

Step 2 Create a file named `pvc.yaml` in the `~/hxkube` directory.

Scope:

Run on administrator host.

Command:

```
touch ~/hxkube/pvc.yml
```

Example:

```
administrator-host:hxkube$ touch ~/hxkube/pvc.yml

administrator-host:hxkube$
```

Step 3 Edit the `pvc.yaml` file and insert the following text. Save the file once complete.

Example:

```
apiVersion: v1
kind: PersistentVolumeClaim metadata:
name: message-board-pvc
 namespace: message-board
spec:
 storageClassName: hyperflex
 accessModes:
    -ReadWriteOnce
 resources:
   requests:
      storage: 100Gi
```

Step 4 Run the `oc create <file>` command to submit the `pvc.yaml` file and create the Persistent Volume Claim object in the OpenShift cluster. In parallel, as part of the operation, HyperFlex will create a Persistent Volume object to complement the Persistent Volume Claim object and bind the two together in OpenShift.

Scope:

Run on administrator host.

Command:

```
oc create -f ~/hxkube/pvc.yaml
```

Example:

```
aadministrator-host:hxkube$ oc create -f ~/hxkube/pvc.yml

persistentvolumeclaim "message-board-pvc" created
administrator-host:hxkube$
```

Step 5 Check the status of the Persistent Volume Claim object with the `oc get pvc` command to make sure it was created successfully and is “Bound” to a Persistent Volume object.

Scope:

Run on administrator host.

Command:

```
oc get pvc
```

Example:

```
administrator-host:hxkube$ oc get pvc

NAME                STATUS    VOLUME
CAPACITY   ACCESS MODES   STORAGECLASS   AGE
message-board-pvc   Bound     hx-default-message-board-pvc-c54defc5-f26b-11e8-8aff-00505698adb3
100Gi      RWO,ROX        hyperflex      <invalid>
administrator-host:hxkube$
```

Step 6 Now that the Persistent Volume Claim and the supporting Persistent Volume (from HyperFlex) have been successfully created, you can deploy the Message Board Pod. Start the process by creating a file called `message-board.yml` in the `~/hxkube` directory.

Scope:

Run on administrator host.

Command:

```
touch ~/hxkube/message-board.yml
```

Example:

```
administrator-host:hxkube$ touch ~/hxkube/message-board.yml

administrator-host:hxkube$
```

Step 7 Edit the `message-board.yml` file and insert the following text. Save the file once complete.

Note For purposes of this example, the Pod definition also includes a simple Service definition so the application can be reached outside the OpenShift cluster.

Note The Persistent Volume Claim `message-board-pvc` is referenced in the Pod definition.

Example:

```
apiVersion: v1
kind: Pod
metadata:
    name: message-board
    labels:
        app: message-board
        name: message-board
    namespace: message-board
spec:
    containers:
    - name: message-board
      image: michzimm/message_board:version1
      volumeMounts:
      - name: demovolume1
        mountPath: /sqldb
      ports:
      - containerPort: 5000
    volumes:
    - name: demovolume1
      persistentVolumeClaim:
        claimName: message-board-pvc
---
apiVersion: v1
kind: Service
metadata:
  name: message-board
  labels:
    name: message-board
  namespace: default
spec:
  type: NodePort
  ports:
  - port: 5000
    nodePort: 30002
  selector:
    name: message-board
```

Step 8 By default, OpenShift runs containers as the default user within a project (namespace). In order to successfully deploy the Message Board Pod, it is required that you add the `privileged` security context constraint (scc) to the `default` user in the message-board project (namespace).

Important This step is not recommended in production as this presents a security risk.

Scope:

Run on administrator host.

Command:

```
oc adm policy add-scc-to-user privileged -z default -n <namespace>
```

Example:

```
administrator-host:hxkube$ oc adm policy add-scc-to-user privileged -z default -n message-board

scc "privileged" added to: ["system:serviceaccount:message-board:default"]
administrator-host:hxkube$
```

Step 9 Use the `oc create -f <file>` command to deploy the Message Board Pod definition.

Scope:

Run on administrator host.

Command:

```
oc create -f <file>
```

Example:

```
administrator-host:hxkube$ oc create -f ~/hxkube/message-board.yml

pod "message-board" created
service "message-board" created
administrator-host:hxkube$
```

Step 10 Use the `oc get pods` command to check the status of the deployed Pod and ensure it is running.

Scope:

Run on administrator host.

Command:

```
oc get pods -n <namespace>
```

Example:

```
administrator-host:hxkube$ oc get pods -n message-board

NAME            READY     STATUS              RESTARTS   AGE
message-board   1/1       Running             0          2m
administrator-host:hxkube$
```