ORACLE

OpenShift Cluster Setup with Agent Based Installer on Private Cloud Appliance

Step-by-step to deploy OpenShift Clusters using Agent Based Installer on Private Cloud Appliance

Version 14.0

Public

ORACLE

Purpose statement

This document provides step-by-step instructions to deploy Redhat OpenShift Clusters on Oracle Private Cloud Appliance.

Disclaimer

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle software license and service agreement, which has been executed and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement, nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

This document is for informational purposes only and is intended solely to assist you in planning for the implementation and upgrade of the product features described. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described in this document remains at the sole discretion of Oracle. Due to the nature of the product architecture, it may not be possible to safely include all features described in this document without risking significant destabilization of the code.

Introduction

In today's changing world of business IT, automation and native solutions play a crucial role in ensuring flexibility, scalability and effectiveness. As more companies embrace various Kubernetes platform as OpenShift for managing containers, the importance of deployment processes becomes clear. This guide helps to address the specific preference of using Redhat's OpenShift for Kubernetes by offering a step-by-step instructions of installing OpenShift on Private Cloud Appliance (PCA).

OpenShift offers a variety of features, including automated installation, upgrades, and life cycle management. It also provides advanced security and monitoring, integrated storage, and CI/CD pipeline management. OpenShift is powered by Kubernetes, an open-source project that streamlines the entire application lifecycle, from development to delivery to management

This content is provided for informational purposes and self-supported guidance only. Consultancy or other assistance related to the content is not covered under the Oracle Support contract or associated service requests. If you have questions or additional needs, then please do reach out to your Oracle Sales contact directly.

Access and Considerations

Before getting into the steps for setting up OpenShift on Private Cloud Appliance, it's important to make sure you have all the necessary prerequisites ready. This section outlines the elements needed to carry out the installation with minimal disruptions.

The prerequisites covered in this document encompass operational aspects, including:

- **Access to Private Cloud Appliance (PCA):** Make sure you have an active account with the appropriate permissions to create and manage resources on Private Cloud Appliance.
- **Access to Redhat Agent Based Installer Clusters Portal:** During the step-by-step deployment process, you will be taking advantage of the Redhat Hybrid Cloud Console Portal: https://console.redhat.com/openshift/install/pull-secret
- **Access Credentials:** You will need a bastion / jump host within you PCA target Compartment's Tenancy to connect with an external Terraform website to download and install Terraform. You will also use this Bastion host for OCI CLI, Running the Terraform Scripts and preparing the OpenShift Clusters.
- **Networking Considerations:** Ensure that your network setup allows for communication, which includes setting up VCN (Virtual Cloud Network) configuring subnets and managing security lists. If your PCA is disconnected, you will need to access external networks via a DRG.
- **Infrastructure Planning:** Develop a strategy outlining the resources needed capacity projections and design aspects customized to suit your organizations requirements.

By fulfilling these criteria, you will be ready to follow the instructions outlined in this paper guaranteeing a seamless and effective deployment of your Redhat OpenShift Clusters on Oracle Private Cloud Appliance.

Bastion Configuration

A Bastion Host, also referred to as a jump server, provides a server created to enable access to a private network from an external network, like the internet. For the OpenShift deployment on Oracle Private Cloud Appliance, a bastion host is utilized to host Terraform scripts utilized to configure infrastructure and build OpenShift Clusters. It will also be utilized for reaching resources and instances within a subnet in your Private Cloud Appliance environment.

Advantages of Using a Bastion Host are:

- Access Point: Acts as an entry point for administrators and users to connect to instances in the private network without exposing them directly to the internet.
- Offers a controlled gateway thereby reducing the vulnerability surface.
- Isolation of Critical Resources: Ensures that computing resources within the network are safeguarded and isolated from internet exposure. Helps in minimizing risks associated with attacks on systems.
- Monitoring: Centralizes access logging simplifying tracking and auditing access to resources. Can be integrated with security information and event management (SIEM) systems, for monitoring capabilities.

By implementing a bastion host, you can securely and efficiently manage access to your Oracle Private Cloud Appliance or Private Cloud Appliance resources, ensuring that your private instances remain protected and accessible only through a secure, controlled entry point.

The installation of the OCI CLI will be revisited in the first section of this procedure:

1. Install OCI CLI in your bastion host. For the step-by-step and how to install OCI CLI and properly setup the OCI CLI configuration file in your instance, refer to the following links below: https://docs.oracle.com/en-us/iaas/Content/API/SDKDocs/cliinstall.htm and

https://docs.oracle.com/en/engineered-systems/private-cloud-appliance/3.0-latest/user/user-usr-ce-cli.html#usr-cli-obtain-cabundle.

Overview

The installation process outlined below will involve building two hosts within the targeted compartment of the OpenShift Cluster installation.

The first instance to be installed will be a “bastion” host, within the designated compartment, and will be used to run two Terraform scripts. The first script will be used for building IAM Resources on your of the PCA (Dynamic Groups and associated Policies). The second Terraform Script will be for building the Infrastructure Resources on the PCA System to support the OpenShift Cluster (the OpenShift VCN, Public and Private Subnets, Load Balancers, Internet GW, NAT GW, and DNS Server, etc). It will include all the resources needed to allow the Control Plane and Worker Nodes to function that form a cluster. The bastion will be installed in the designated OpenShift Compartment and must be configured to communicate through a designated PCA DRG Subnet or Internet GW Subnet within the PCA’s parent tenancy.

The second instance to be installed, will be designated as the “jumpserver”. The jumpserver will be configured as a webserver to communicate to the Openshift Cluster. The jumpserver will be installed with the OpenShift Client cli for this cluster communication. The jumpserver will be installed within the OpenShift VCN built by the Terraform scripts.

Finally, three Control Plane Nodes and three Worker Nodes will be provisioned, along with configuring the external and internal Load Balancers that form the Cluster.

Bastion Server - Prerequisites

In preparation to running Terraform Scripts, this section will prepare the installation of Terraform and OCI CLI onto the PCA System.

Currently Private Cloud Appliance (PCA) does not support OCI Resource Manager Stack (RMS). To run the Terraform script, a Bastion Server (or jump server) will be installed on the PCA. Terraform and OCI CLI will then be loaded on to the Bastion Server for execution.

Public IP Addresses: You will need a minimum set of 3 Public IP Addresses in your Tenancy. Due to CIDR Rules you can configure up to a binary set of 8, 16, 32 etc – which will provide ample number of Public IP’s.

The following is an overview of installations and directories that will be placed on to the **bastion host** for the execution of Terraform scripts. You will install Terraform and the OCI CLI on the bastion host to run the needed Terraform scripts.

Installation Overview

The following procedure can be broken down into the following 3 stages.

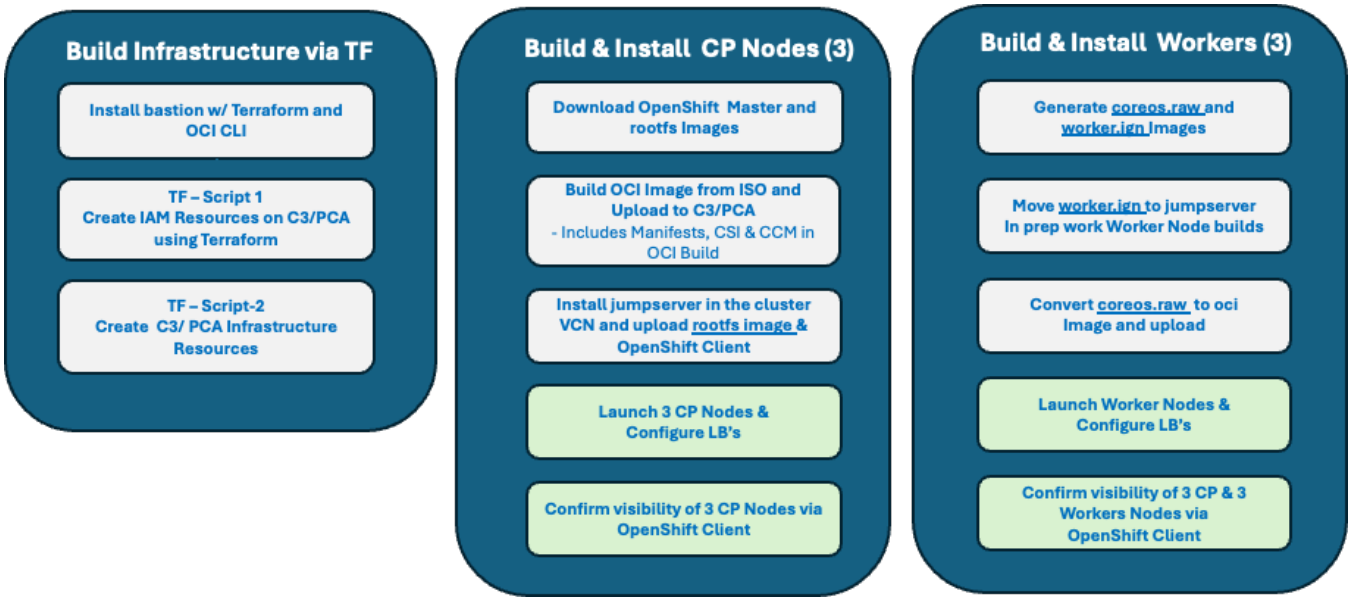


Figure-1

Bastion Installation

1. Create a Compartment in your OCI Home Region that will be managing this deployment. Allow a few minutes for this to propagate to your PCA system.
2. Login to the PCA System's Console
3. Click on "Compute" and switch to the Compartment made in Step-1
4. Create an Instance
    a. **Name:** Provide a name of your choosing (for the remaining of this procedure, we will refer to this as "bastion")
    b. **Create in Compartment:** Choose a Compartment used to build your Cluster
    c. **Fault Domain**: Automatically select …
    d. **Source Image:** Oracle Linux 9
    e. **Shape:** VM.Standard1.Flex
    f. **OCPUs:** 2
    g. **Memory (GBs):** 16
    h. Subnet:
        1. For disconnected PCA's, the VCN and Subnet chosen for VNIC attachment must have a DRG for external access
        2. Choose an appropriate **VCN** and **Subnet** for external access if you are using a DRG. You will need to have a Proxy Server installed for the DRG. The Network Addresses of the Proxy will be needed in later steps in setting the /etc/environment in both the bastion and jumpserver hosts.
        3. If your Tenancy has Public Internet access, a Public Subnet with IGW can be built for access to Terraform and OCI cli downloads needed for the bastion host.
    i. Public IP:
        1. Select the Radio Button
    j. Private IP:
        1. Keep (Optional)

k. **SSH Key:** Upload or Cut and Paste the Public Key into the dialogue box
l. **Create the Instance**
m. **Record the Public IP Address** – You will be using this to SSH into in the next step

5. SSH into the **bastion** Instance using the Public IP Address
    a. [my_laptop ~]$ ssh opc@<bastion_IP_Address>

Terraform Installation

1. Download the Terraform binary AMD64 zip file onto your laptop from: https://developer.hashicorp.com/terraform/install

   Note: The Terraform binary version will continue to be updated. Choose the latest version.

2. SSH into your bastion instance
    a. [user@my_laptop ~]$ ssh opc@<bastion_private_IP_address>

3. On the bastion instance, make a directory "prerequisite"
    a. [bastion ~]$ mkdir /home/opc/prerequisite/

4. Transfer the Terraform zip file from your laptop to the bastion host via secure copy, scp:
    a. [user@ my_laptop ~]$ scp terraform_1.8.3_linux_amd64.zip opc@<bastion_private_IP_address>:/home/opc/prerequisite/

5. Unzip the Terraform Binary and move it to the **bin** directory
    a. [opc@bastion ~ ./prerequisite]$ unzip terraform_1.8.3_linux_amd64.zip
    b. [opc@bastion ~ ./prerequisite]$ sudo mv terraform /usr/local/bin
    c. [opc@bastion ~ ./prerequisite]$ sudo chmod +x /usr/local/bin/terraform
    d. Verify the Terraform installation was a success.
        1. [opc@bastion ~ ./prerequisite]$ terraform version
        2. Note :The processor type and version number are returned

Installing and Configuring OCI CLI

1 Install the OCI CLI

a. Goto: Offline Installation documentation
    1. **Choose**: Choose an "Oracle Linux 9 Offline" version
    2. **Example**: oci-cli-3.49.2-Oracle-Linux-9-Offline
b. Download the zip file to your local machine
c. Transfer from the file from your local machine to the targeted bastion server and store in the prerequisite folder:
    1. [user@my_laptop ~]$ scp oci-cli-3.49.2-Oracle-Linux-9-Offline.zip opc@<bastion_private_IP_address>:/home/opc/prerequisite/
d. SSH into the bastion the bastion server
    1. [user@my_laptop ~]$ ssh opc@<bastion_private_IP_address>
e. Unzip the oci cli file:
    1. [opc@bastion ~ ./prerequisite]$ unzip oci-cli-3.49.2-Oracle-Linux-9-Offline.zip
f. Install oci cli
    1. [opc@bastion ~./prerequisite]$ cd oci-cli-installation
    2. [opc@bastion ~ ./prerequisite/oci-cli-installation]$ bash install.sh --offline-install

1. Return “blanks” (keep defaults) and “n” to the final question to any changes regarding paths – unless there are known criteria for these file locations that need to be set.

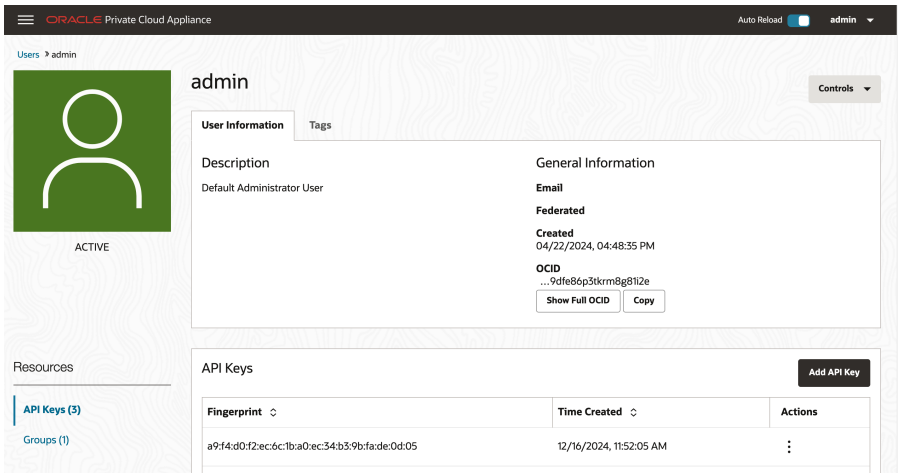
6. **Configure the OCI CLI**
   - **a.** [opc@bastion ~]$ oci setup config
   - **b.** [opc@bastion ~]$ **Enter a location for your config [/home/opc/.oci/config]:** Enter (keep path)
   - **c.** [opc@bastion ~]$ **Enter a user OCID**: Enter your User OCID (Goto your PCA Console and navigate to your User Identity and obtain your OCID)
   - **d.** [opc@bastion ~]$ **Enter your Tenancy OCID**: Enter your Tenancy OCID (This can be found on the PCA Console in the upper right corner, Profile.)
   - **e.** [opc@bastion ~]$ **Enter a Region by index or name**: Select the Region name’s representative number and enter it (for this example, we are using 58: us-ashburn-1)
   - **f.** [opc@bastion ~]$ **Do you want to generate a new API Signing RSA key pair?** Y
     1. You can opt for N, if you already have an API signing key on the PCA.
   - **g.** [opc@bastion ~]$ **Enter a directory for your keys to be created [/home/opc/.oci]:** Enter (keep default)
   - **h.** [opc@bastion ~]$ **Enter a name for your key [oci_api_key]: Enter** (keep default)
   - **i.** [opc@bastion ~]$ **Enter a passphrase for your private key:** select N/A (or choose a passphrase if desired)

     - This completes the initial configuration.
     - A directory will have been made: /home/opc/.oci
     - **It will contain the following files:**
       - config
       - oci_api_key.pem
       - oci_api_key_public.pem

Note: If you declined a new api key pair, place your keys in this directory with the file names shown above.

3. **Make a Copy of your oci_api_key_public.pem that you will use in your PCA User profile’s API Keys:**
   a. [opc@bastion ~]$ cat oci_api_key_public.pem
   b. Cut and paste this key and save it for the next step

4. **Navigate to your PCA User Profile to update your User API Keys**
   a. Identity
   b. Select Users
   c. Select your User identity
   d. Scroll down on left nav to “API Keys” and select
   e. Select “Add API”
   f. Cut and Paste the oci_api_key_public.pem
   g. Select “Add”
   h. **Copy the Configuration profile View presented** (It will be used in the next step)
   i. **Note the API key fingerprint.** This should be the same fingerprint found in the oci config file: /home/opc/.oci/config



5. **In this section, you will be making changes to the /.oci/config file. This file provides the necessary tenancy and security keys for accessing to your PCA System**

   j. [opc@bastion ~]$ vi ~/.oci/config
      1. You can use the copied file from the previous step or use the examples below)
      2. Change the [DEFAULT] to [pca]
      3. Cut and paste the complete section and place beneath it. This will be the section for the PCA Region
      4. Change the second section from [home] to [PCA]. These names can change to your choosing. They will be referenced in Terraform Scripts later in this document.

Example provided:

[pca]
user=ocid1.user.oc1..aaaaaaaaqlnen3dpgyqn6xi6hwl7vuertngidgncd7rbgt2x2ga2rmyawvya
fingerprint=[redacted]
key_file=/home/opc/.oci/oci_api_key.pem
tenancy=ocid1.tenancy.oc1..aaaaaaaalmxrgmsqoou26rdybgthdwtwdwhn7km5oq3u25f6bdpoxihqef6q
region= scasg02.us.oracle.com #example region only.

Note: For the **region** of your PCA, simply use the URL that you log into your PCA with. Everything after "console" is your Region. Example: https://console.**scasg02.us.oracle.com/** : scasg02.us.oracle.com is the Region.

7. For the next step, you will need a Certificate Authority Bundle. The instructions are provided in the link below. The details have already been provided starting with Step 7a.

   Reference Doc: https://docs.oracle.com/en/engineered-systems/private-cloud-appliance/3.0-latest/user/user-usr-ce-cli.html#usr-cli-obtain-cabundle

   a. Type into your browser: https://iaas.<your_PCA_domain>/cachain

   b. Copy/Cut this output for the next step to paste

      Note: Your PCA "Domain" is the same as the "Region" in the oci config file and noted as an example on the previous step.

8. Create the pca.pem file
    a. Create a file under the /home/opc/.oci directory and name it “pca.pem”
    b. [opc@bastion /home/opc/.oci~]$ vi pca.pem
        1. Paste the Cert File from the previous step and save.

9. Create the oci_cli_rc file
    a. [opc@bastion /home/opc/.oci~]$ vi oci_cli_rc
    b. Cut and paste and update with your tenancy id :

```
[pca]
tenancy=<your tenancy ocid>
cert-bundle=/home/opc/.oci/pca.pem
```

10. Next, modify the firewall for CLI access to PCA
    a. [bastion /home/opc/.oci]$ sudo systemctl stop firewalld
    b. [bastion /home/opc/.oci]$ sudo rm -f /etc/firewalld/direct.xml
    c. [bastion /home/opc/.oci]$ sudo systemctl start firewalld

11. Confirm CLI access **on PCA**
    a. [bastion /home/opc/.oci]$ oci os ns get --profile pca

```
{
  "data": "axpu7uokzjmj" #example data set only.
}
```

9. Set up a proxy on your bastion host before running the terraform script for systems that do not have direct internet access
    a. [bastion /home/opc/.oci]$ sudo vi /etc/environment
    b. Edit the following examples for http_proxy, https_proxy and no_proxy into the /etc/environment file.

Example:

```
http_proxy=http://www-proxy.<your_domain>.com:80
https_proxy=http://www-proxy.<your_domain>.com:80
no_proxy=localhost,127.0.0.1,1,2,3,4,5,6,7,8,9,0,.<your_domain>.com,.ie.oracle.com,.oraclecorp.com
```

Note: You will need to contact your system administrator for proxy and port settings for your environment.

10. When the /etc/environment file is saved, ensure to set the source to /etc/environment
    a. [bastion /home/opc/.oci]$ source /etc/environment

Terraform Script Execution – Part 1 (Provisioning IAM Resources)

All the Identity related resources, including Dynamic Groups, Policies and Tags, will be created on the PCA in this initial step.

Two Terraform scripts will be run on the bastion server created in previous steps. This first script will be to create the needed Two Dynamic Groups (one for control plane, one for compute nodes), Policies for the Dynamic Groups and Tags for the Control and Worker Nodes. The second script will be to create infrastructure resources on PCA.

Important Note:

The PCA System uses the same Terraform Scripts as C3. As of this writing, Jan 22nd, 2025, the Engineering team is making updates to naming conventions on github to reflect naming for the PCA. This is simply a minor update. These changes will be complete by Feb 7th, 2025.

You will see notes during this installation to point out the references. These notes will change once github changes are made.

1. SSH into the Bastion Server created in previous steps
   a. [user@my_laptop ~]$ ssh opc@<bastion_Private_IP_Address>

2. Make a directory, "openshift"
   a. [opc@bastion /home/opc]$ mkdir openshift

3. Make a directory, "createResourceOnHomeRegion" in the openshift directory
   a. [opc@bastion /home/opc/openshift]$ mkdir createResourceOnHomeRegion

4. Copy the two files, createInfraResources.tf, and terraform.tfvars from the following github repo as files into the directory, createResourceOnHomeRegion.
   a. https://github.com/oracle-quickstart/oci-openshift/tree/c3-la/edge/c3 into the directory, createResourceOnHomeRegion.
   b. Replace the variables in **terraform.tfvars**
   c. Replace the default values of "compartment_ocid", "cluster_name" and "home_region_profile_name" (found during OCI CLI Configuration)

```
tenancy_ocid             = "<replace with C3 or PCA tenancy_ocid>"
compartment_ocid         = "<replace with the compartment_ocid>"
home_region_profile_name = "<replace with the oci config profile name of the home region or
PCA region>"
cluster_name = "<replace with the OpenShift cluster name>"
```

Note-1: For the **home_region_profile_name,** this is simply the designation you had provided in the oci-cli config file. In this example we designated it [pca]. You are free to designate any name you like.

5. Before launching the TF scripts, determine if you have tenancy privileges to create Dynamic Groups for specific compartment. **If you do not, please see Step-7 before applying Terraform Scripts**

6. Apply the Terraform Script:

   a. **Run these two, always, before performing terraform steps:**
      1. Ensure your Network source is set:
         a. [Bastion createResourceOnPCA]$ **source /etc/environment**
      2. Ensure your PCA Certification location is set
         a. [Bastion createResourceOnPCA]$ **export custom_cert_location=/home/opc/.oci/pca.pem**

   b. [Bastion createResourceOnHomeRegion~]$ terraform init
   c. [Bastion createResourceOnHomeRegion~]$ terraform plan

d. [Bastion createResourceOnHomeRegion~]$ terraform apply

Note1: If you receive an error during the "terraform init" step, "Error: Failed to query available provider packages", simply exit the bastion and ssh again into it.

Note2: If you received an error, showing a Dynamic Group error, you will need to contact your sysadmin to make the following **changes on your behalf shown in Step7.**

Note3: As mentioned at the outset, we are reusing C3 Scripts. Hence the name "createResourceOnHomeRegion". For C3, IAM changes need to be made on it's federated OCI Home Region's Tenancy. In reality, it's creating IAM Resources. The terraform variables that were updated, will point it to your PCA instead of OCI Home Region.

7. If you do not have administrative rights to the tenancy in your home region to create Dynamic Groups and Policies. This provisioning will be done manually and at **the tenancy** level by an Admin:
   a. **Create a Dynamic Group: "<your-cluster-name>_compute_nodes"**
      1. On the Left Nav, navigate to Identity and Security and choose Domains and the choose Default
      2. On the Left Nav, choose Dynamic Groups and Create a Dynamic Group with the name above.
      3. In the Matching Rules section, copy the following.
      4. Replace the ${var.compartment_ocid} , in Rule-1, with the target compartment ocid
      5. Replace the ${var.cluster_name} ,in Rule-2, with the target compartment cluster name

      - Rule 1:
        - instance.compartment.id= '${var.compartment_ocid}'
      - Rule-2:
        - tag.openshift- ${var.cluster_name}.instance-role.value='compute'

   b. **Create another Dynamic Group**: **"<your-cluster-name>_control_plane_nodes>"**
      1. In the Matching Rules section, copy the following.
      2. Replace the `${var.compartment_ocid}` in Rule-1 with the target compartment ocid
      3. Replace the `${var.cluster_name}` in Rule-2 with the target compartment cluster name

      - Rule 1:
        - instance.compartment.id= '${var.compartment_ocid}'
      - Rule-2:
        - tag.openshift- ${var.cluster_name}.instance-role.value='compute'

   c. **Create a Policy** for the control plane nodes in your compartment and name it **: "<your-cluster-name>_control_plane_nodes>".**
      1. In the Policy Builder, using manual editor, copy the following section below and replace the `${oci_identity_dynamic_group.openshift_control_plane_nodes.name}` with the **control plane DG** name created in the previous step.
      2. Replace the `${oci_identity_dynamic_group.openshift_control_plane_nodes.name}` with the target compartment's **cluster name**

      1. Allow dynamic-group ${oci_identity_dynamic_group.openshift_control_plane_nodes.name} to **manage volume-family** in compartment id ${var.compartment_ocid}

2. Allow dynamic-group ${oci_identity_dynamic_group.openshift_control_plane_nodes.name} to **manage instance-family** in compartment id ${var.compartment_ocid}
3. Allow dynamic-group ${oci_identity_dynamic_group.openshift_control_plane_nodes.name} to **manage security-lists** in compartment id ${var.compartment_ocid}
4. Allow dynamic-group ${oci_identity_dynamic_group.openshift_control_plane_nodes.name} to **use virtual-network-family** in compartment id ${var.compartment_ocid}
5. Allow dynamic-group ${oci_identity_dynamic_group.openshift_control_plane_nodes.name} to **manage load-balancers** in compartment id ${var.compartment_ocid}

d. **Comment out lines 79-107 in the createIdentityResources.tf script**
   1. This will include:
      a. resource "oci_identity_dynamic_group" "openshift_control_plane_nodes"
      b. resource "oci_identity_policy" "openshift_control_plane_nodes"
      c. resource "oci_identity_dynamic_group" "openshift_compute_nodes"

Terraform Script Execution – Part 2 ("Creating PCA Resources")

In this section, we will create all the infrastructure resources (LB's, VCN's, Instances, etc) on the PCA System.

1. SSH into the Bastion image from your laptop
   a. [my_laptop ~]$ ssh opc@<bastion_Private_IP_Address>
2. Export the PCA Certificate location. 2a provides an example of the path and pem file name only. If you followed this installation procedure, you can cut and paste it into your command line.
   a. [opc@bastion ~] export custom_cert_location=/home/opc/.oci/pca.pem
3. Check that the variable is set properly
   a. [opc@bastion ~] **`echo $custom_cert_location`**

   Note-1: During the execution of the next TF script, and see an x509 Cert issue, copy the above command in 2a and re-run the scripts. Ensure that you are NOT logged into the bastion with two ssh sessions.

3. Make a directory **"createResourceOnPCA"** and copy following two files into the directory: https://github.com/oracle-quickstart/oci-openshift/tree/c3-la/edge/c3

Note: Again, when you visit the github location above, it's referencing C3. Updates will be made in naming by Feb 7th, 2025.

   a. [Bastion /home/opc/openshift]$ mkdir createResourceOnPCA
   b. Make a file for each respective file from the above github link:
      1. createInfraResources.tf
      2. terraform.tfvars
   c. Update all the variables in the PCA Region's **terraform.tfvars** file

```
c3_or_pca_region              = "<replace with C3 region or PCA region>"
compartment_ocid              = "<replace with the compartment_ocid>"
```

```
c3_or_pca_region_profile_name = "<replace with the oci config profile name of the C3
region or PCA region>"
cluster_name                  = "<replace with the OpenShift cluster name>"
zone_dns                      = "<replace with the name of cluster's DNS zone>"
image_id_manually_created_on_C3 = "<replace with the image ocid which manually created
on C3>"
control_plane_shape           = "<replace with the Compute shape of the control_plane
nodes>"
compute_shape                 = "<replace with the Compute shape of the compute
nodes>"
create_openshift_instance_pools = false
```

Note-1: The C3 or PCA region is the FQDN of the console you logged into. Following are two examples:
(1) C3 Lab Console: console.scasg03.us.oracle.com. Region = scasg03.us.oracle.com
(2) PCA Lab Console: console.scasg02.us.oracle.com Region = scasg02.us.oracle.com

Note-2:
- Keep **image_id_manually_created_on_PCA** : to the default value (no edits).
- Change the **create_openshift_instance_pools**: to false

Note-3: Compute Shapes
(1) For C3, insert and use (for both Control Plane & Compute): VM.PCAStandard.E5.Flex
(2) For PCA, insert and use (for both Control Plane & Compute): VM.PCAStandard1.Flex

4. Ensure your Network source is set:

   a. `[Bastion createResourceOnPCA]$ source /etc/environment`

5. Ensure your PCA Certification location is set

   a. `[Bastion createResourceOnPCA]$ export custom_cert_location=/home/opc/.oci/pca.pem`

6. Apply the Terraform Script

```
[Bastion createResourceOnPCA]$ terraform init
[Bastion createResourceOnPCA]$ terraform plan
[Bastion createResourceOnPCA]$ terraform apply
```

Note: If "terraform init" causes the following error: "**Error**: Failed to query available provider packages", exit the bastion host and ssh back in.

5. **Save the Terraform output ! in the same directory**, as it will be used it in later steps when building the Master CP Images in the Cluster Installation section. Below is a generic example:

```
useInstancePrincipals: true
compartment: <compartment-ocid>
vcn: <vcn-ocid>
loadBalancer:
  subnet1: <subnet-ocid>
  securityListManagementMode: Frontend
  securityLists:
    <subnet-ocid>: <security-ocid>
```

```
rateLimiter:
  rateLimitQPSRead: 20.0
  rateLimitBucketRead: 5
  rateLimitQPSWrite: 20.0
  rateLimitBucketWrite: 5
```

Cluster Installation

In this section, you will prepare OpenShift images, create master/worker nodes accordingly and verify the installation.

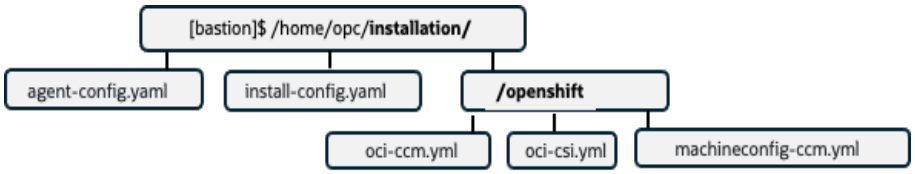
Install Master Nodes

Prepare the Openshift Master Images

Generate iso image and rootfs.img

In this section you will be copying files from github into a created installation directory.

Important: The construct of this directory, subdirectory and files will need to reflect the structure below:



1. You will need the contents in https://github.com/oracle-quickstart/oci-openshift/tree/c3-la/edge/c3 directory under the edge/PCA/agentBasedInstallation.
    a. You will be copy and pasting then editing 4 files.
        i. agent-config.yaml
        ii. install-config.yaml
        iii. oci-ccm.yml
        iv. oci-csi.yml
        v. machineconfig-ccm.yml – DOES NOT require any edits
2. Create the installation directory:
    a. [opc@bastion /home/opc]$ mkdir installation
3. Config the files agent-config.yaml and install-config.yaml in the installation directory shown above.
    a. For the installation, you can use the default configuration copied from the repo. Just replace some values.
4. Create the file **agent-config.yaml**, under the /home/opc/**installation** directory
    a. [opc@bastion /home/opc/installation]$ vi agent-config.yaml
    b. Replace the metadata: **name** and **namespace** with the cluster name.

```
apiVersion: v1alpha1
metadata:
```

```
  name: <replace with the OpenShift cluster name>
  namespace: <replace with the OpenShift cluster name>
rendezvousIP: 10.0.16.20 # or replace with custom value of your configuration
bootArtifactsBaseURL: http://10.0.0.5  # or replace with custom value of your
configuration
```

5. Create the file **install-config.yaml,** under the /home/opc/**installation** directory
    a. [opc@bastion /home/opc/installation]$ vi install-config.yaml
    b. Replace the **install-config.yaml** section of the install-config.yaml**,** with the following:
        i. **Metadata, name:**
        ii. **baseDomain:** this should match the "zone_dns" naming in the createResourceOnPCA terraform.tfvars file.
        iii. **sshKey:** Specify your SSH public key.
        iv. **pullSecret:** The pull secret that you need for authenticate purposes when downloading container images for OpenShift Container Platform components and services, such as Quay.io. See Install OpenShift Container Platform 4 from the Red Hat Hybrid Cloud Console.
        v. **proxy information:** replace with contents found in the previously configured /etc/environment customized for your environment, **while retaining the IP Address of 10.0.0.0/16** (or the custom machine Network CIDR intended to use)

```
metadata:
  name: <replace with the OpenShift cluster name>
baseDomain: <replace with the name of cluster's DNS zone>
sshKey: '<replace with your publish ssh key>'
pullSecret: '<replace with your pull secret key generated from RH>'
proxy:
  httpProxy: http://www-proxy.us.oracle.com:80
  httpsProxy: http://www-proxy.us.oracle.com:80
  noProxy:
localhost,127.0.0.1,1,2,3,4,5,6,7,8,9,0,.us.oracle.com,.ie.oracle.com,.oracleco
rp.com,10.0.0.0/16
```

shh key: Ensure that this is the ssh key used when building the jump-host instance.
proxy :
You will need to have consulted your Network admin for proper /etc/environment settings in previous steps. See Reference document in the following step for more information.

6. Next, we will be making changes to the files to be located under the /home/opc/installation/**openshift** directory
    a. Identical changes will be made to both the **oci-ccm.yml** and **oci-csi.yml files.**

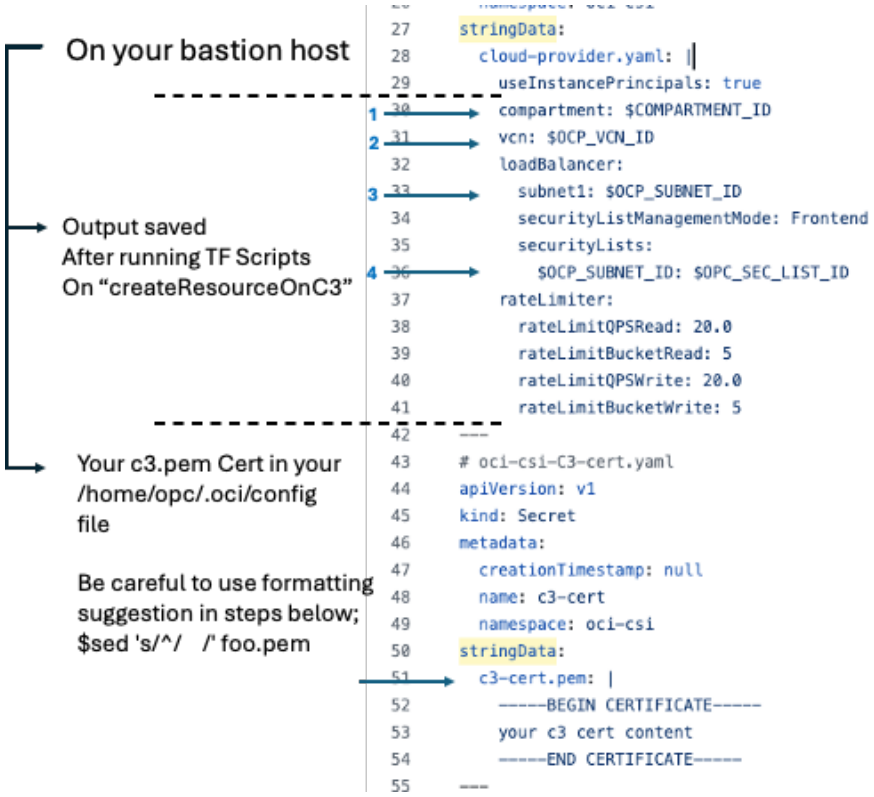
b. No changes will be required for the **machine-config.yml** that will be installed in this same directory.

c. Remember, the placement of these files within the directory structure is important.

d. Once the **openshift-install** script is run, all the files within the installation directory will be consumed and no longer be present.

e. The content you will need for changes in both these files can be found in the following sections pasted below.

f. Open another terminal window and ssh into your bastion to obtain this needed information:



g. Create a file under /home/opc/installation/openshift for the first file: **oci-ccm.yml**

i. [opc@bastion ~ ./installation]$ mkdir openshift
ii. [opc@bastion ~ ./installation/openshift]$ vi oci-ccm.yml
iii. Copy the respective file contents found on github into this new file
iv. Make updates to the #oci-ccm-04-cloud-controller-manager-config.yaml with the **output saved after running the terraform script on PCA**. Starting at line 193….

```
# oci-ccm-04-cloud-controller-manager-config.yaml
apiVersion: v1
kind: Secret
metadata:
  creationTimestamp: null
  name: oci-cloud-controller-manager
  namespace: oci-cloud-controller-manager
stringData:
  cloud-provider.yaml: |
    useInstancePrincipals: true
    compartment: $COMPARTMENT_ID
    vcn: $OCP_VCN_ID
    loadBalancer:
      subnet1: $OCP_SUBNET_ID
```

```
    securityListManagementMode: Frontend
    securityLists:
      $OCP_SUBNET_ID: $OPC_SEC_LIST_ID
  rateLimiter:
    rateLimitQPSRead: 20.0
    rateLimitBucketRead: 5
    rateLimitQPSWrite: 20.0
    rateLimitBucketWrite: 5
```

v. Next , **staying in the same oci-ccm.yml**, update the # oci-ccm-PCA-cert.yaml with your PCA.pem found in your /home/opc/.oci/PCA.pem file. Save the file after this update.

```
# oci-ccm-PCA-cert.yaml
apiVersion: v1
kind: Secret
metadata:
  creationTimestamp: null
  name: PCA-cert
  namespace: oci-cloud-controller-manager
stringData:
  PCA-cert.pem: |

    -----BEGIN CERTIFICATE-----
    your PCA cert content
    -----END CERTIFICATE-----
```

Important Note: The formatting of this pem within the two yml files are sensitive to justification.

- **To replicate the formatting required for the oci-ccm.yml & oci-csi.yml, use the following command on your pem file:**
  - [opc@test-ssh .oci]$sed 's/^/ /' foo.pem
  - where “foo.pem” is the name of your pem file
  - the spaced between the two forward slashes is equal to 4 spaces
    - this is equal to the justification in the two files
- The above command will create an output to cut and paste to place in both the oci-ccm.yml & oci-csi.yml files

h. Repeat the previous 2 updates for the **oci-csi.yml file,** starting at line 20.
   i. [opc@bastion ~ installation/openshift]$ vi oci-csi.yml
   ii. Copy the oci-csi.yml into the editor
   iii. Within the **oci-csi.yml file**, update the **# oci-csi-01-config.yaml** with the same content/process found above in 4.a.iii
   iv. Within the **oci-csi.yml file**, update the **# oci-csi-PCA-cert.yaml** with the same content/process found above in 4.a.iv
   v. Save and Quit

i. Create the final file, machineconfig-ccm.yml, under the same openshift directory
   i. [opc@ bastion ~ installation/openshift]$ vi machineconfig-ccm.yml
   ii. Save and Quit (Note: This file requires no modifications.)

7. Download openshift-install binary in <installation-directory> directory.

   a. Download OpenShift Install with following command, or download the version on demand from link in (b) that follows: https://mirror.openshift.com/pub/openshift-v4/clients/ocp/4.15.21/openshift-install-linux.tar.gz

b. [opc@ bastion ~./installation]$ wget https://mirror.openshift.com/pub/openshift-v4/clients/ocp/4.15.21/openshift-install-linux.tar.gz

c. Extract the openshift installation:

i. [opc@ bastion ~./installation]$ tar -xvf openshift-install-linux.tar.gz

8. Run command "./openshift-install agent create image --log-level debug" under the <**installation-directory**>. The command completes the following actions below.

a. [opc@ bastion ./installation]$ ./openshift-install agent create image --log-level debug

b. You will observe the install has consumed the previously configured files and has introduced the changes below:

1. Created a subdirectory, ./<installation-directory>/**auth directory**, and places **kubeadmin-password** and **kubeconfig** files in the subdirectory.
2. Created a **rendezvousIP** file based on the IP address that you specified in the agent-config.yaml configuration file.
3. Created an **.iso image** file
4. Created a subdirectory, ./<installation-directory>/**boot-artifacts** directory, and places a **rootfs image** file in it.

Important Note: In the boot-artifacts directory, you will find the **agent.x86_64-rootfs.img**. This file will need to be transferred to the **/var/www/html when starting a webserver** on your jump host, which will be built in the steps below.

9. Create a subdirectory ./<installation-directory>/prepareImage, place the .iso image file in this subdirectory and then run following commands to convert the .iso file to an .oci file

a. [opc@ bastion ~./ installation]$ mkdir prepareImage

b. [opc@ bastion ~./ installation/]$ mv <file_name>.iso ./prepareImage

10. Convert image iso to qcow2

a. [opc@ bastion ~ ./prepareImage]$ brew install qemu

b. [opc@ bastion ~ /prepareImage]$ qemu-img convert -O qcow2 <file_name>.iso output.QCOW2

```
Note: We need to set firmware as "UEFI" and launch mode as "PARAVIRTUALIZED" for the
openshift image, before we import it on PCA. Following are specific steps.
```

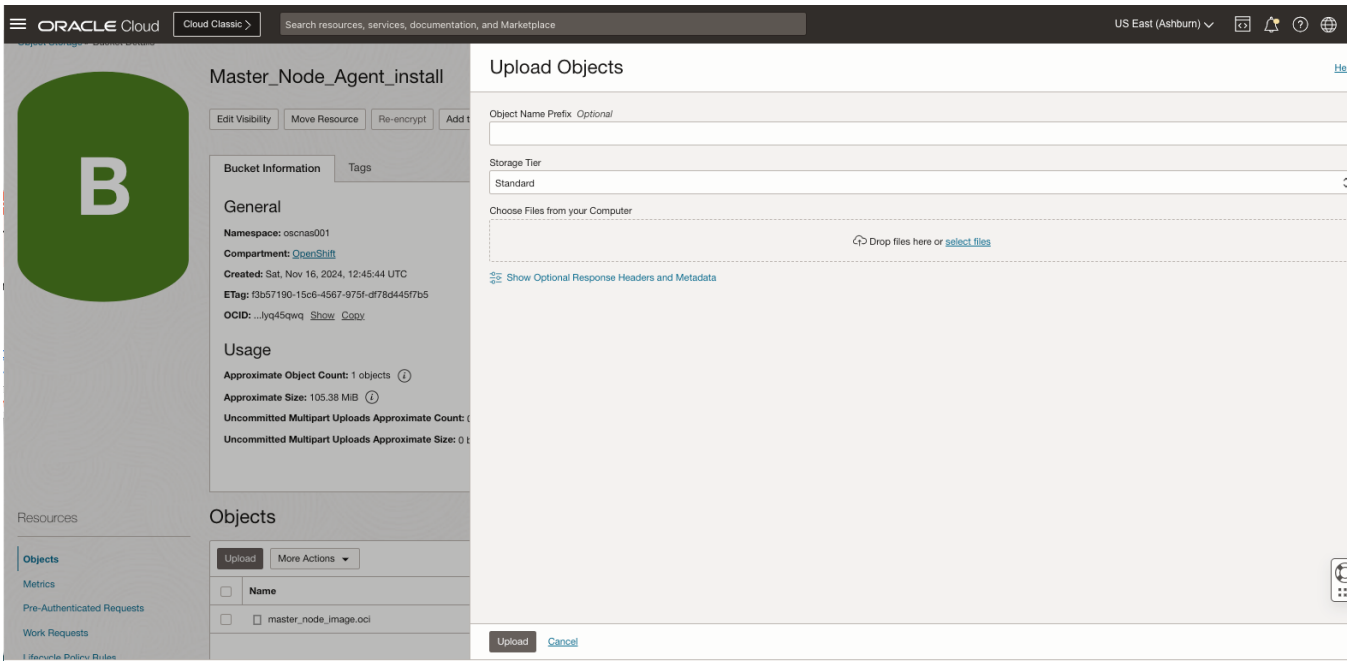
11. Create image_metadata.json as follows

- [opc@ bastion ~ prepareImage]$ vi image_metadata.json
- Paste the following contents into the editor, save and quit

```
{
"version": 2,
"externalLaunchOptions": {
```

```
"firmware": "UEFI_64",
"networkType": "PARAVIRTUALIZED",
"bootVolumeType": "PARAVIRTUALIZED",
"remoteDataVolumeType": "PARAVIRTUALIZED",
"localDataVolumeType": "PARAVIRTUALIZED",
"launchOptionsSource": "PARAVIRTUALIZED",
"pvAttachmentVersion": 1,
"pvEncryptionInTransitEnabled": false,
"consistentVolumeNamingEnabled": false
},
"imageCapabilityData": null,
"imageCapsFormatVersion": null,
"operatingSystem": "Custom",
"operatingSystemVersion": "Custom"
}
```

12. Tar the output.QCOW image and metadata.json to create an oci image
    a. [opc@ bastion ~ prepareImage]$ tar cf master_node_image.oci image_metadata.json output.QCOW2
13. Upload the "master_node_image.oci" to the OCI Home Region bucket, and generate the Pre-Authenticated Request(PAR) URL
    a. Upload the "master_node_image.oci" to OCI bucket.
    b. Select the hotdog (dot) menu next to it and **Create Pre-Authenticated Request** for the object, allowing object reads and extending the expiration date to your needs. Copy the PAR URL and keep it. We will use it in the next step.

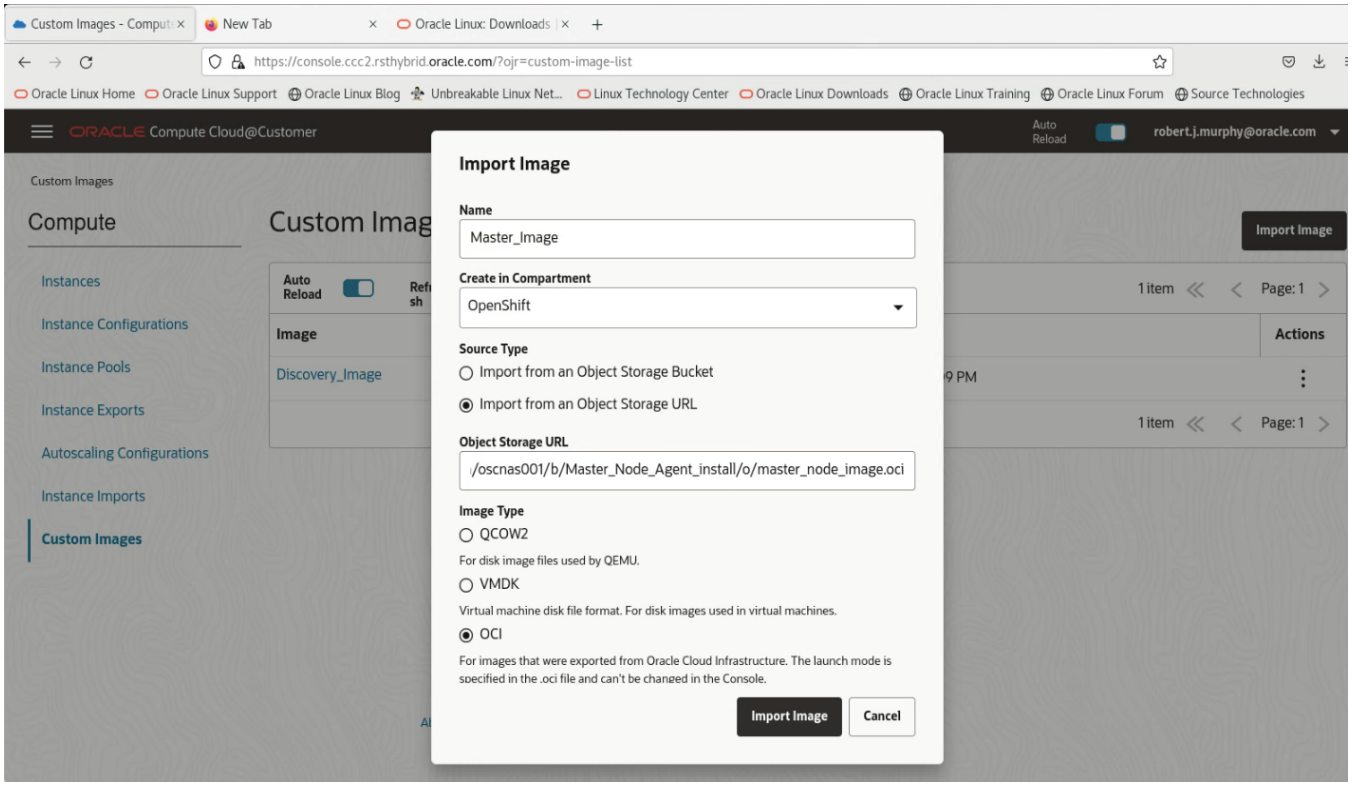


11. Import master node image on PCA portal.

a. Login to your PCA System

b. Click Compute → select your compartment → click custom images

c. Click import image → give it a name → Source Type choose "Import from an Object Storage URL" → Paste the PAR URL we generated in previous step in the "Object Storage URL" → Image type select "OCI" → Launch Mode select "Paravirtualized Mode" → click import image → make sure the image states become available; we will use it later.



Start a web server to upload rootfs

If your PCA System is in a disconnected environment, you need to upload the rootfs manually for booting the iso image, i.e., start a preferred web server, such as any Hypertext Transfer Protocol daemon (httpd), to upload rootfs to the location stated in the bootArtifactsBaseURL parameter in the agent-config.yaml configuration file. (More background info refers to Step8 in the 20.2.3 part of RH)
doc https://docs.redhat.com/en/documentation/openshift_container_platform/4.15/html/installing/installing-on-oci#creating-config-files-cluster-install-oci_installing-oci-agent-based-installer)

If you used the default config files copied from the repo in the previous steps, you can follow steps below.

1. Create an instance on PCA console, in your openshift test compartment
    a. Navigate to Compute/Instances and click on Create Instance
        1. **Name:** jumphost
        2. **Create in Compartment:** Use the pull-down menu to choose your compartment
        3. **Fault Domain**: Leave to default (auto select)

4. **Source Image** / Source Type: Oracle Linux 9
5. **Shape:** VM.PCAStandard1.Flex
6. **OCPU's:** 2
7. **Memory:** 16
8. **Boot Volume:** Click radio and leave at Default
9. **Subnet: VCN =** cluster_name**, Subnet:** Public
10. **Public IP Address:** Click radio button, Assign Public IP Address
11. **Assign Private IP:** 10.0.0.5
    1. **Assign the Private IP as 10.0.0.5.** This value should match the bootArtifactsBaseURL in the https://github.com/oracle-quickstart/oci-openshift/tree/c3-la/edge/c3
12. **Upload your ssh key**
13. **Create the instance**

2. SSH into the newly created "jump-host" on PCA and run following command to setup the webserver.
3. Set up system-wide proxy by adding three lines in environment file.
    a. [opc@jump-host ~]$ sudo vi /etc/environment
    b. If you have implemented a DRG into your design, you will need to set-up specific for the proxy server being used in conjunction with your DRG. Below is an example:

Example:

```
http_proxy=http://www-proxy.us.oracle.com:80
https_proxy=http://www-proxy.us.oracle.com:80
no_proxy=localhost,127.0.0.1,1,2,3,4,5,6,7,8,9,0,.us.oracle.com,.ie.oracle.com,.oraclecorp.com
```

Note: As in previously setting up the bastion host, you will need to create the proper settings for your specific environment.

c. Set your systemwide variables

i. [opc@jump-host ~]$ source /etc/environment

3. Install webserver and tar
    a. [opc@jump-host ~]$ sudo dnf -y install httpd tar
    b. [opc@jump-host ~]$ sudo systemctl enable --now httpd.service
    c. [opc@jump-host ~]$ sudo firewall-cmd --add-service=http --permanent
    d. [opc@jump-host ~]$ sudo firewall-cmd --add-port=80/tcp --permanent
    e. [opc@jump-host ~]$ sudo firewall-cmd --reload
    f. [opc@jump-host ~]$ sudo setenforce 0

4. Check the webservers serving directory.
    a. [opc@jump-host ~]$ sudo grep -i documentroot /etc/httpd/conf/httpd.conf
    b. From the output of 4a, verify the DocumentRoot:

```
DocumentRoot "/var/www/html"

    # access content that does not live under the DocumentRoot.
```

5. Upload rootfs file from the machine which has the openshift installation. This was completed in previous steps: (/home/opc/**installation**/boot-artifacts/**agent.x86_64-rootfs.img** to the jump-host.)
    a. [opc@bastion ~]$ scp agent.x86_64-rootfs.img opc@<jump-host-public-ip>:/home/opc
6. Move the rootfs file to webservers serving directory.
    a. [opc@jump-host ~]$ sudo mv agent.x86_64-rootfs.img /var/www/html
7. Download and Install OpenShift OC client
    a. [opc@jump-host ~]$ wget https://mirror.openshift.com/pub/openshift-v4/clients/ocp/4.15.21/openshift-client-linux.tar.gz
    b. [opc@jump-host ~]$ tar -xvf openshift-client-linux.tar.gz
    c. [opc@jump-host ~]$ sudo mv oc /usr/local/bin/.

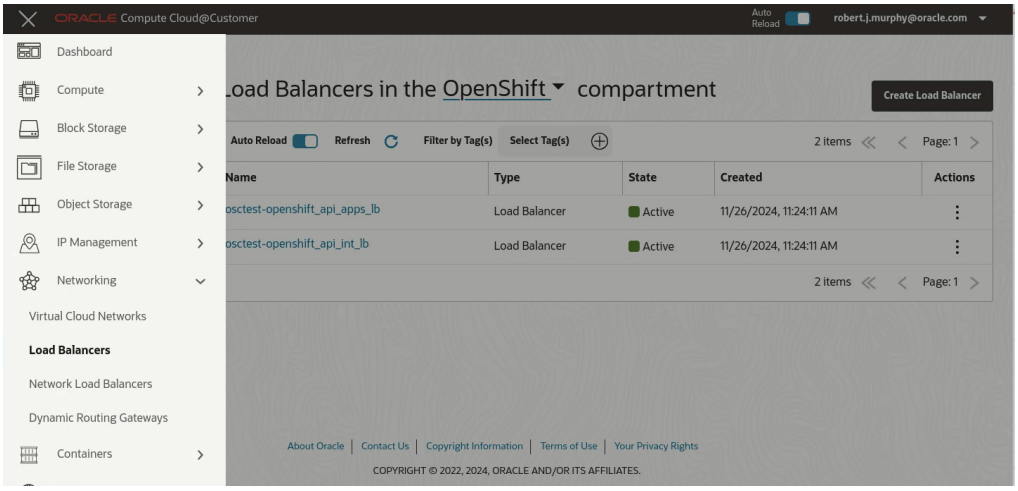
Create control plane instances on PCA and Master Node LB Backend Sets

1. Navigate to the Compute service, then select Instances
2. Select Create instance.
3. Enter a Name for the instance.
4. Select the compartment that the OpenShift cluster is being created in.
5. In the Image and shape section, select Change image and choose My images.
6. Select Custom images. Ensure the Compartment is set to the compartment where your custom image is stored.
7. Select the checkbox beside the name of the master node image we imported on PCA in the previous step, then click Select image.
8. You must select the VM.PCAStandard1.Flex
9. Select Change shape.
10. In the Browse all shapes panel, choose a shape and ensure that the **Number of OCPUs value is 4** or more, and the **Amount of Memory (GB) value is 16 or more**.
11. Click Select shape to continue.
12. In the Boot volume section, select Specify a custom boot volume size and **only specify the boot volume size of 1024, and keep VPU as default.** See Block Volume Performance for information on volume performance units (VPUs)
13. **Choose the VCN** that you specified when editing the oci-ccm.yml file. This VCN should also be referenced in the cidr value of the machineNetwork parameter in the install-config.yaml file.
14. **For the Subnet**, choose Select existing subnet, then choose the **private subnet** for the VCN

15. **First master node only:** In the Primary VNIC IP addresses section, select Manually assign private IPv4 address. In the IPv4 address field, **enter the rendezvous IP address**. For extra master nodes, and for all worker nodes, skip this step.

16. In the Add SSH keys section, select Paste public keys and enter the public key from the SSH key pair discussed in Prerequisites.
17. In the **Network Security Group** section, select **Enable Network Security Group** and select NSG as cluster-controlplane-nsg
18. Click Show advanced options to add a tag to the instance . Using the defined tag option, specify the openshift_tags namespace and the instance_role tag. Then assign control_plane as the tag value

19. **Select Create to provision the compute instance.**
20. **Repeat Steps 1-18 above for the remaining two CP Nodes.**
21. **Add backend sets in LB for master nodes:**
    a. After creation of 3 master nodes, you need to add backend sets in Load Balancers.
        1. Login to your PCA System
        2. Click Networking → click Load Balancers → select your compartment. You should see two load balancers here.



        a. Click the "<cluster-name>-openshift_api_**apps**_lb"

        → click **Backend Sets (3)** (on left nav)

→ click "openshift_cluster_api_backend"

→ click Create Backend

→ click IP Addresses

→ **IP Address 1 Only:** type in the **private rendezvous IP address** and the port **6443** (same as provisioned during instance launch)

→ click Add IP Addresses twice

→ **IP Address 2 & 3**: type in the private IP addresses of the other two master nodes and the port **6443**.

b. Click the "<cluster-name>-openshift_api_**int**_lb"

→ click Backend sets.

→ click "**openshift_cluster_api_backend"**

→ click Create Backend

→ click IP Addresses

→ type in the **private rendezvous IP address** and the port **6443**

→ click Add IP Addresses twice → type in the private IP addresses of the other two master nodes and the port **6443**.

→ click "**openshift_cluster_infra-mcs"**

→ click Create Backend

→ click IP Addresses

→ type in the private rendezvous IP address andthe port **22623**

→ click Add IP Addresses twice

→ type in the private IP addresses of the other two master nodes and the port **22623**.

→ click "openshift_cluster_infra-mcs_2"

→ click Create Backend

→ click IP Addresses

→ type in the private rendezvous IP address andthe port **22624**

→ click Add IP Addresses twice

→ type in the private IP addresses of the other two master nodes and the port **22624**.

22. There should be 3 master nodes that are running in the cluster. SSH into the jump host and run the following commands to **verify** the cluster installation.
23. Copy and paste the kubeconfig file from the ./<installation-directory>/auth directory to the jump-host instance,and export it as env variable KUBECONFIG
    a. [opc@jump-host ~]$ export KUBECONFIG=~/auth/kubeconfig
24. Set up system-wide proxy by adding three lines in environment file. Append the server domain in the no_proxy.

a. As an example, the server in kubeconfig file is "server: https://api.**osctest.osctest.rsthybrid.oracle.com:6443** ", capture the suffix " **osctest.osctest.rsthybrid.oracle.com:6443** " and append it in the no_proxy config.

b. Make the following changes to the /etc/environment

1. [jump-host ~]$ sudo vi /etc/environment

- http_proxy=http://10.61.2.219:3128
- https_proxy= http://10.61.2.219:3128
- no_proxy=localhost,127.0.0.1,1,2,3,4,5,6,7,8,9,0,.rsthybrid.oracle.com, .**osctest.osctest.rsthybrid.oracle.com:6443** ← **Add this to no_proxy.**

c. Set the source path

1. [jump-host ~]$ source /etc/environment

Please Note: The installation will take some time, please check periodically until you see the outputs that follow.

You may notice that CP's ROLES here include "worker", which is incorrect. We will fix it in a later step, after adding compute (worker) nodes to the cluster.

d. Query the node status using the openshift client installed

1. [jump-host ~]$ oc get nodes

```
NAME    STATUS    ROLES                          AGE     VERSION
cp1     Ready     control-plane,master,worker    5m19s   v1.28.11+add48d0
cp2     Ready     control-plane,master,worker    20m     v1.28.11+add48d0
cp3     Ready     control-plane,master,worker    21m     v1.28.11+add48d0
```

If you see these error messages, it may mean that the installation is still in progress.

e. [opc@jump-host ~]$ oc get nodes -A

```
E0910 00:01:19.245859  180079 memcache.go:265] couldn't get current
server API group list:
Get "https://api.jzabtest2.jzabtest2.PCA.dfosterdev.com:6443/api?timeout
=32s": EOF
```

e. [opc@jump-host~]$ oc get nodes

```
Unable to connect to the server: EOF
```

Add worker nodes

Prepare the Openshift worker image

Generate coreos-rawdisk.raw image and worker.ign file

1. SSH into the "jump-host" instance. Run this script https://github.com/davidc-dev/oci-ocp-disconnected-tweaks/blob/main/00-extract-ignition-create-worker-image.sh
2. This script will generate two files, "coreos-rawdisk.raw" and "worker.ign".
3. Copy and paste script to jump host instance and add the permission to execute the file

a. [opc@jump-host ~]$ vi 00-extract-ignition-create-worker-image.sh

1. Copy and paste the content from the file referenced

2. Save and Quit

b. [opc@jump-host ~]$ chmod +x 00-extract-ignition-create-worker-image.sh

Note: <server IP or hostname> is the bootArtifactsBaseURL parameter in the agent-config.yaml configuration file**. If you are using the default config file copied from the github repo, you can use 10.0.0.5.**

3. Define the Webserver IP Address. <server IP or hostname> is the bootArtifactsBaseURL parameter in the agent-config.yaml configuration file. If you are using the default config file copied from the github repo, **you can use 10.0.0.5.**
4. [opc@jump-host ~]$ WEBSERVER=<**server IP** or hostname>
5. [opc@jump-host ~]$ ./00-extract-ignition-create-worker-image.sh $WEBSERVER
    a. Two images have been built:
        i. (1) coreos-rawdisk.raw and
        ii. (2) worker.ign files.
    b. The coreos-rawdisk.raw file will be converted to a worker image in oci format then uploaded to your Home OCI Bucket Storage to be fetched by your PCA system with a PAR. In order to upload the worker oci image to OCI Bucket Storage, you will need to move it to your PC/MAC. **The coreos-rawdisk.raw is a large file and will take up to 30m for each transfer.** Plan your time accordingly.
    c. The worker.ign file will be moved to the webservers html file in later steps
6. [opc@jump-host ~]$ WEBSERVER=<**server IP** or hostname>
7. [opc@jump-host ~]$ ./00-extract-ignition-create-worker-image.sh $WEBSERVER
8. Convert the ***coreos-rawdisk.raw*** file to worker_node_image.oci file and upload the image to a bucket and create a pre-authenticated URL. Then import the custom image by the URL. (Same procedure as for the master node image. You can refer to Step 6.7.8 in the “Generate iso image and rootfs.img section”.) Note, the custom image is large (~ 2.4GB), so importing will take some time.
9. Convert image raw to qcow2
    a. [opc@jump-host ~]$ brew install qemu
    b. [opc@jump-host ~]$ qemu-img convert -O qcow2 coreos-rawdisk.raw output.QCOW2

10. We need to set firmware as "UEFI" and launch mode as "PARAVIRTUALIZED" **for** the openshift image, before we **import** it on PCA. Following are specific steps.
    a. create image_metadata.json as follows
    b. [opc@jump-host ~]$ vi image_metadata.json
    c. Cut and paste the following, save and quit

```
{
"version": 2,
"externalLaunchOptions": {
"firmware": "UEFI_64",
"networkType": "PARAVIRTUALIZED",
"bootVolumeType": "PARAVIRTUALIZED",
"remoteDataVolumeType": "PARAVIRTUALIZED",
"localDataVolumeType": "PARAVIRTUALIZED",
"launchOptionsSource": "PARAVIRTUALIZED",
```

```
"pvAttachmentVersion": 1,
"pvEncryptionInTransitEnabled": false,
"consistentVolumeNamingEnabled": false
},
"imageCapabilityData": null,
"imageCapsFormatVersion": null,
"operatingSystem": "Custom",
"operatingSystemVersion": "Custom"
}
```

9. Tar the two files to build an .oci image
   a. [opc@jump-host ~]$ tar cf worker_node_image.oci image_metadata.json output.QCOW2
10. Upload the ***worker_node_image.oci*** image to a OCI Home Region bucket and then transfer the image to PCA, in similar fashion as was done for the Master Node Image

Upload the worker.ign file to the web server

The worker.ign file needs to be copied to the webserver serving directory (same place the rootfs is from the original installation.).

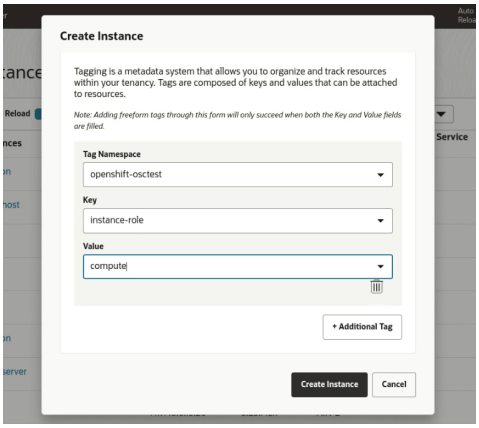
1. Move the ign file to webservers serving directory.
   a. [opc@jump-host ~]$ sudo mv worker.ign /var/www/html
2. Verify the ign file is in the serving directory alongside the previously moved file, agent.x86_64-rootfs.img, for the Control Plane Master Nodes:
   a. [opc@jump-host ~]$ ls /var/www/html/

      agent.x86_64-rootfs.img **worker.ign**

Create compute instances and Worker Node LB Backend Sets

1. Navigate to the Compute service, then select Instances
2. Select Create instance.
3. Enter a Name for the instance.
4. Select the compartment that the OpenShift cluster is being created in.
5. In the Image and shape section, select Change image and choose My images.
6. Select Custom images. Ensure the Compartment is set to the compartment where your custom image is stored.
7. Select the checkbox beside the name of the **worker node image** we imported on PCA in the previous step, then click Select image.
8. Select Change shape.
9. Click Select shape to continue.
10. In the Browse all shapes panel, choose a shape and ensure that the **Number of OCPUs value is 4 or more, and the Amount of Memory (GB) value is 16 or more.**
11. In the Boot volume section, select Specify a custom boot volume size and **only specify the boot volume size of 100, and keep VPU as default (10 VPUs).** See Block Volume Performance for information on volume performance units (VPUs)

12. Choose the VCN that you specified when editing the oci-ccm.yml file. This VCN should also be referenced in the cidr value of the machineNetwork parameter in the install-config.yaml file.
13. For the Subnet, choose Select existing subnet, then **choose the private subnet for the VCN**.
14. **First master node only**: In the **Primary VNIC IP addresses** section, select Manually assign private IPv4 address. In the IPv4 address field, **enter the rendezvous IP address**. For extra master nodes, **and for worker nodes, skip this step.**
15. In the Add SSH keys section, select Paste public keys and enter the public key from the SSH key pair discussed in Prerequisites.
16. In the **Network Security Group** section, select **Enable Network Security Group** and select NSG as **cluster-compute-nsg**
17. Click Show advanced options to add a tag to the instance . Using the defined tag option, specify the **openshift_tags namespace** (ie: <openshift-cluster_name>) and **the instance_role** tag. Then assign **compute** as the tag value. (See figure on following page)



18. Select Create to provision the compute instance.
19. Repeat Steps 1-18 for the other two worker nodes
20. After the creation of 3 worker nodes, you need to add backend sets in Load Balancers.
21. **Add backend sets in LB for worker nodes:**
    a. Login to your PCA System
    b. Click Networking → click Load Balancers → select your compartment. You should see two load balancers here. **Click the "<cluster-name>-openshift_api_apps_lb"** → click Backend sets.
    c. Click "**openshift_cluster_ingress_http**" → click Create Backend → click IP Addresses → click Add IP Addresses twice → type in the private IP addresses of all the worker nodes and the port **80**.
    d. Click "**openshift_cluster_ingress_https**" → click Create Backend → click IP Addresses → click Add IP Addresses twice → type in the private IP addresses of all the worker nodes and the port **443**.
22. There should be 3 worker nodes now configured with Backend Sets along with 3 Control Plane nodes in the previous sections.
23. Verify all Control Plane nodes and all Worker Nodes are running
    a. Copy and paste the kubeconfig file from the ./<installation-directory>/auth directory to the jump-host instance, and export it as env variable KUBECONFIG.
        1. `[opc@jump-host ~]$ export KUBECONFIG=~/auth/kubeconfig`
    b. **If you're worker nodes to not provide a status, you will need to run the following two tasks**

1. `[opc@jump-host ~]$ oc get csr | grep Pending`

c. A list of “kubernetes.io/kube-apiserver-client-kubelet” will be provided

1. You will need to execute the following command for each output from the last request where `csr-wr9ff` is an example of one of the output lines

1. `[opc@jump-host ~]$ oc adm certificate approve csr-wr9ff`

d. `[jump-host ~]$ oc get nodes -A`

```
NAME      STATUS   ROLES                         AGE     VERSION
cp1       Ready    control-plane,master,worker   3h1m    v1.28.11+add48d0
cp2       Ready    control-plane,master,worker   3h16m   v1.28.11+add48d0
cp3       Ready    control-plane,master,worker   3h17m   v1.28.11+add48d0
worker1   Ready    worker                        10m     v1.28.11+add48d0
worker2   Ready    worker                        4m50s   v1.28.11+add48d0
worker3   Ready    worker                        2m49s   v1.28.11+add48d0
```

Note: The installation will take some time. Perhaps up to 10 minutes to provide worker status.

24. You may notice that CP's ROLES here include “worker”, which is incorrect. We will fix it now refers to the instruction in the link https://github.com/davidc-dev/oci-ocp-disconnected-tweaks/blob/main/README.md#2-node-role-issues.

a. [opc@jump-host~]$

oc patch scheduler cluster --type merge -
p '{"spec":{"mastersSchedulable":false}}'scheduler.config.openshift.io/cluster patched

oc rollout -n openshift-image-registry restart deploy/image-registry

oc rollout -n openshift-image-registry restart deploy/image-registry

oc rollout -n openshift-monitoring restart statefulset/alertmanager-main

oc rollout -n openshift-monitoring restart statefulset/prometheus-k8s

oc rollout -n openshift-monitoring restart deployment/Grafana

oc rollout -n openshift-monitoring restart deployment/kube-state-metrics

oc rollout -n openshift-monitoring restart deployment/telemeter-client

oc rollout -n openshift-monitoring restart deployment/thanos-querier

Output:

```
deployment.apps/router-default restarted
Error from server (NotFound): deployments.apps "image-registry" not found
statefulset.apps/alertmanager-main restarted
statefulset.apps/prometheus-k8s restarted
Error from server (NotFound): deployments.apps "grafana" not found
deployment.apps/kube-state-metrics restarted
deployment.apps/telemeter-client restarted
deployment.apps/thanos-querier restarted
```

25. Verify the ROLES of the CP nodes have been correctly fixed, i.e., the “Worker” role has been removed. All the CP and worker nodes show as Ready.
    a. [opc@jump-host ~]$ oc get nodes -A

```
NAME      STATUS   ROLES                  AGE     VERSION
cp1       Ready    control-plane,master   3h5m    v1.28.11+add48d0
cp2       Ready    control-plane,master   3h21m   v1.28.11+add48d0
cp3       Ready    control-plane,master   3h21m   v1.28.11+add48d0
worker1   Ready    worker                 14m     v1.28.11+add48d0
worker2   Ready    worker                 9m21s   v1.28.11+add48d0
worker3   Ready    worker                 7m20s   v1.28.11+add48d0
```

26. If you do not see your all you Worker Nodes come up, complete the following procedure under paragraph 4:
    a. https://github.com/davidc-dev/oci-ocp-disconnected-tweaks/blob/main/README.md#2-node-role-issues

References

1. https://docs.oracle.com/en-us/iaas/Content/openshift-on-oci/agent-installer.htm
2. https://github.com/davidc-dev/oci-ocp-disconnected-tweaks
3. https://docs.redhat.com/en/documentation/openshift_container_platform/4.15/html/installing/installing-on-oci#installing-oci-agent-based-installer

Author: Anderson Souza, Amardeep Dhillon