Openshift 4 Disconnected Collection

Overview

The why

- The goal of this repository is to provide a standardized workflow for deploying OpenShift 4 in a disconnected environment using an agent-based installer.

The how

- The idea is to have all variables set in group_vars/all, with some defaults set in roles. The defaults in roles are only applied in case they are not set in group_vars.

Variables

- cluster_version: This variable controls the version of the cluster and allows for the pulling of operators and additional images (for the cluster) via oc-mirror. This requires an active subscription with a pull secret in ~/.docker/config.json. It will pull all the required binaries along with some additional nice-to-haves (such as Butane).
- cluster_deployment: This variable controls how the cluster is deployed. It will generate the install-config and agent-config based on configured variables.
- static this is used to set things that very rarely change or set variables defined in other vars, that are built ontop of other variables

Some assumptions include:

- SSH keys will not exist.
- SSL certificates for the hosted registry will be generated, which will run on the disconnected host. Post-Install Configuration

Goals

- The current goal for post-install configuration is to deploy ArgoCD and have it manage the configuration and enforcement.
- To assist with this, a local Git repository will be configured as described in this article.

Status

- This is a work in progress, and we will continue to make enhancements over time.

Requirements (disconnected system)

- Packages:
  - nmstate
  - openssl
  - httpd-tools
- Ansible-collections
  - dellemc.openmanage

Additional Info:

View notes on recommended procedures for vm migrations

Thanks:

This started as as a fork from hyperkineticnerd/ansible-ocp4-agent, but we have since diverged quite a bit away, but still would like to give credit.

Red Hat OpenShift Installation Guide

This guide covers the process for installing Red Hat OpenShift 4.15+ on Red Hat Enterprise Linux 9 with options for FIPS and STIG configurations. The instructions support deployment on both connected and disconnected environments.

Prerequisites

RHEL Install

RHEL 9 Installation: Install Red Hat Enterprise Linux 9 and register it with your Red Hat account (if connected to the internet).

> **NOTE:** If the goal is to have a FIPS-enabled cluster, the bastion host has to be FIPS-enabled as well. If you don't need FIPS, you can ignore the following configurations.

Environment Configurations:

> **NOTE:** These configurations can be done post install. Changes to usbguard/sysctl.conf will require a reboot, while fapolicyd will only require restart on the service.

1. Ensure **FIPS** mode is enabled for RHEL 9.

   - Enable FIPS during installation; or
   - Enable FIPS post-installation

2. Configure **STIG** compliance as needed

3. Configure **fapolicyd** for Ansible Playbooks:

   - Allow regular users to run Ansible playbooks by creating a new file at `/etc/fapolicyd/rules.d/22-ansible.rules` with the following contents:

   ```
   allow perm=any uid=1000 : dir=/home/user/.ansible
   allow perm=any uid=1000 : dir=/home/user/.cache/agent
   allow perm=any uid=1000 : dir=/usr/share/git-core/templates/hooks
   allow perm=any uid=1000 : dir=/pods
   allow perm=any uid=1000 : dir=/usr/bin
   allow perm=any uid=0,1000 : dir=/tmp
   ```

   > **NOTE:** This was what was used for testing, you can use Group Identifier (GID) or User Ieentifer (ID). I used UID because it more narrow scoped that GID. Policy and best jugement should dictate which route you do

4. Adjust User Namespace Limits for Registry Pod:

   - Increase the `user.max_user_namespaces` setting to enable the registry pod to run as a non-root user. Update `/etc/sysctl.conf` as follows:

   ```
   # Per CCE-83956-3: Set user.max_user_namespaces = 0 in /etc/sysctl.conf
   user.max_user_namespaces = 5
   ```

5. Enable Access to External USB Devices (for Disconnected Environments):

   - Add the following commands to the `%post` section in your kickstart file:

```
systemctl disable usbguard
sed -i 's/black/\#black/g' /etc/modprobe.d/usb-storage.conf
sed -i 's/install/\#install/g' /etc/modprobe.d/usb-storage.conf
```

6. Install Ansible/Podman:

```
sudo dnf install -y ansible-core container-tools
```

Running the Automation

Getting Collection installed (connected side or connected cluster)

1. Clone this Repository:

```
git clone https://github.com/cjnovak98/ocp4-disconnected-config
```

2. Navigate to the Playbooks Directory: `bash        cd ocp4-disconnected-config/playbooks`
3. Install Required Ansible Collections(1st time/connected system):

> **NOTE:** This will install disconnected-collection and should be run on a fresh system, before anything else has been run.

```
ansible-playbook initial-ansible-collection.yml
```

or

```
./initial-ansible-collection.yml
```

Directory Structure Assumptions

Ensure that the content resides in `/pods/content` (definable in `playbooks/group_vars/all/cluster-deployment.yml`). It is assumed this is the transferable media when running connected side for a disconnected cluster and disconnected config is /pods/content/ansible

Run Content Gathering Playbook to Prepare Disconnected Environments:

> **NOTE:** Ensure a valid pull secret exists. You can get your pull secret from https://console.redhat.com/openshift/create/local and store it in `~/.docker/config` on the host where you're running the automation.

Update Environment Variables Modify `group_vars/all/cluster-version.yml` to customize it for your environment. Key variables include:

- common_openshift_release: Target cluster version
- common_previous_openshift_release: Previous version, used for updates
- common_trident_version: trident target version
- content_generation_disconnected_operators: Array of all Catalog sources Openshift will use
  - redhat-operator-index: Name of this catalog source
    - name: Name of any operator to pull within this catalog source
      - channel: specific version of operator to pull on
  - redhat-marketplace-index:
  - community-operator-index:
  - certified-operator-index:
- content_generation_additionalImages: an array of addtinal images to pull

```
# Example group_vars/all/cluster-version.yml

common_openshift_release: 4.16.4
common_previous_openshift_release: 4.16.3
common_trident_version: v24.06.0
```

```
# Disconnected Images

#content_generation_pull_mirror: true

## Operators To be mirrored, based on index they are in
content_generation_disconnected_operators:
  - redhat-operator-index:
    - name: kubernetes-nmstate-operator
    - name: kubevirt-hyperconverged
  - redhat-marketplace-index:
  - community-operator-index:
  - certified-operator-index:

## additionalImages for oc-mirror to pull. Include version
content_generation_additionalImages:
  - name: registry.redhat.io/ubi8/ubi:latest
  - name: registry.redhat.io/ubi9/ubi:latest
  - name: quay.io/noseka1/toolbox-container:full
```

> **NOTE:** for the operator portion of cluster-versions, if nothing is definded as an operator for a catalog source, the entire source is omitied, it is the same design for channel, however the latest is pulled vs being omitied

Run the automation If you are deploying on a disconnected system then you will first need to gather all of the openshift content on a machine that has internet connection and transfer it over. There is a playbook that you can run which will gather the appropriate content:

```
ansible-playbook -K gather-content.yml
```

or

```
./gather-content.yml
```

Once you have the content downloaded, transfer it to your disconnected machine and put in the content directory (i.e. /pods/content), you will also need the `ocp4-disconnected-config` folder used to run this playbook

Install the Collection on a Disconnected Machine

You can either transfter the content from the media used to get it to the disconnected side, or update `playbooks/group_vars/all/cluster-deployment.yml` to point to the mount point of the media

> **Example:** For testing, im running from a 1TB external drive, mounted in `/pods/content` and the dirctory structure was:

```
ls /pods/content
bin   downloads   images   ocp4-disconnected-config
```

Install required Ansible Collection from the previous step:

> **NOTE:** This will install need the media to be mounted, the group vars set, and be in the `ocp4-disconnected-config/playbooks` directory

```
ansible-playbook disconnected-ansible-collection.yml
```

or

```
./disconnected-ansible-collection.yml
```

Run the Deployment Playbook

Update Environment Variables

- Modify `group_vars/all/cluster-deployment.yml` to customize it for your environment. Key variables include:
  - `common_openshift_dir`: Directory for pulled content to live.
  - `common_connected_cluster`: Set to `true` if the cluster itself is connected.
  - `mirror_content_pull_mirror`: Set to `true` to pull content (set to `false` for disconnected environments).
  - `common_fips_enabled`: `true` if the host is stigged and FIPS is enabled.
  - `common_cluster_domain`: Top-Level Domain (TLD) for the cluster.
  - `common_ip_space`: First three octets of the IP address range for both the bastion and cluster.
  - `common_nodes`: Details of nodes, including name, last octet of the node IP, and MAC addresses.
  - `idracs_user` and `idracs_password`: iDRAC credentials.
  - `idracs`: Node name and IP of the node's iDRAC.

```yaml
# Example playbooks/group_vars/all/cluster-deployment.yml

common_git_repos:
  - "https://github.com/cjnovak98/ocp4-disconnected-collection.git"
  - "https://github.com/cjnovak98/ocp4-disconnected-config"

common_openshift_dir: /pods/content
common_connected_cluster: false
mirror_content_pull_mirror: false

common_openshift_interface: ens1
common_fips_enabled: true
common_cluster_domain: example.com

common_ip_space: 192.168.122
common_nodes:
  - name: master-0
    ip: '42'
    mac: 52:54:00:8d:13:77
  - name: worker-0
    ip: '43'
    mac: 52:54:00:8d:13:78

# iDRAC
#idrac_user: test
#idrac_password: tester

#media_share_idracs:
#  - name: master-0
#    ip: '{{ ip_space }}.5'
#  - name: master-1
#    ip: '{{ ip_space }}.6'
#  - name: master-2
#    ip: '{{ ip_space }}.7'
#  - name: worker-0
#    ip: '{{ ip_space }}.8'
```

NOTE: if cluster hardware supports idrac, uncomment to idrac vars.

For both connected and disconnected clusters, deploy the cluster by running:

```
ansible-playbook -K deploy-cluster.yml
```

or

```
./deploy-cluster.yml
```

Fix DNS for disconnected cluser without a DNS Server

If you are running a disconnected server (or atleast on a network with no DNS), We recommend that you now point your bastion at master-0 node to be able to resolve cluster resources (ingress/api)

1. Open nmtui Interface:

   Run the following command in your terminal:

   ```
   sudo nmtui
   ```

2. Select "Edit a Connection":

   Use the arrow keys to navigate to Edit a connection and press Enter.

3. Choose Your Interface:

   Highlight the interface that lives in the same IP space and VLAN as your soon to be deployed cluster (e.g. eth0, or eth0.670).

   Press Enter to edit the selected connection.

4. Edit IPv4:

   Use the arrow keys to navigate to IPv4 CONFIGURATION.

5. Add DNS Server:

   Navigate to the DNS servers field.

   Enter the IP of master-0.

6. Save Changes:

   Use the arrow keys to navigate to `OK` and press Enter to save the settings.

7. Activate the Connection:

   Navigate back to the main nmtui menu and select Activate a connection.

   Highlight the interface you just edited, and press Enter to deactivate and reactivate the connection, applying the changes.

8. Verify the DNS Settings:

   Run the following command to ensure the DNS server is correctly applied:

   ```
   nmcli dev show <interface_name> | grep IP4.DNS
   ```

   Replace `<interface_name>` with the name of your network interface.

9. Test with oc:

   You can test resolution with oc by running the following:

   ```
   oc whoami
   ```

   It should return:

   ```
   sysadmin
   ```

Add an additional node

Check Variables Validate the variables definded from the `Run the Deployment Playbook` portion above

> **NOTE:** Double check the `add_worker_disk_type: sda` playbooks/group_vars/all/cluster-deployment.yml matches the actual disk type of the target nodes; i.e. vda (virtual drives), sda (sata drives), or nvme drives.

Run the playbook

```
ansible-playbook -K create-worker-for-single-node.yml
```

or

```
./create-worker-for-single-node.yml
```

> **NOTE:** DNS needs to be functional as definded in the previous section, or it will fail when pulling the worker ignition file from the "cluster".

OCP GitOps Day 1

This helm chart is in charge of some basic setup after the initial deployment of OpenShift.

The idea behind this is that without OpenShift GitOps, working with an easily observable and maintainable infrastructure as code setup becomes hard. This template essentially just sets up that GitOps installation to where is is useful for the next steps (day2) where the bulk of the cluster configuration is performed.

This helm chart currently is responsible for the initial deployment and configuration of the OpenShift GitOps operator, including the automated setup of the ArgoCD application for the 'Day 2' operations.

Values

In order to properly run this chart, you will need to either modify the existing values.yaml file or create your own values file specifying the values relevant to your setup.

This can be built by copying the existing values.yaml, and adjusting accordingly.

Mainly, you will need the IP of the system you ran the Ansible from, which is where resources are hosted for the disconnected install, as well as for continued GitOps work. You will also need the ssh private key of the user that was used to run the Ansible, as that is in turn used to access the Git repository for the aforementioned GitOps. Example values file:

```
gitOps:
  #the namespace where the argocd instance will reside, defaults to openshift-gitops since that is the
  namespace: openshift-gitops
  #This source variable represents the catalog source where the operator lives. For disconnected instal
  source: cs-redhat-operator-index

gitRepoTemplate:
  # The URL of the repository setup in the Ansible step. replace the IP with that of your syscon
  url: git@<ip of syscon>:repos/ocp4-disconnected-config
  # The SSH key of the user that ran the Ansible step, used to connect to the git repo. Be sure to incl
  sshPrivateKey: |
    -----BEGIN OPENSSH PRIVATE KEY-----
    -----END OPENSSH PRIVATE KEY-----
```

Usage

To apply this template to your cluster, simply run the helm install command:

```
helm install gitops-day1 . --values=./your/values.yaml
```

This applies the yaml to the cluster, and will perform the day 1 operations. Note that it might take a few minutes for everything to come up and be ready, so be patient. You can tell the installation has finished and was successful by navigating to ArgoCD (top right apps menu in OCP web ui -> ArgoCD), clicking 'Log in via OpenShift', and seeing the ArgoCD dashboard. You should see the cluster-day2 app already loaded and attempting to sync. If not you can create the day 2 application manually. You should also see your git repository that's hosted on the syscon by navigating to Settings (on the left) -> Repositories. You should see your gitOps git repo in the list, with a green checkmark indicating success.

You can also use the following command to output the finished product without actually applying it to OpenShift, in order to debug and check:

```
helm template test . --debug | less
```

OCP 4 Disconnected Config Day2 Helm Chart

This chart applies the cluster settings and resources necessary for day to day operations in the target environment. It is intended to be deployed via OpenShift GitOps as part of a infrastructure as code process.

In this case, comments have been provided in both the default `values.yaml` and each of the `.yaml` files in the `templates/` directory to help explain each component.

Deployment

Configuring the Template

To use, you will simply add an application in ArgoCD (Openshift GitOps), and supply this repo/directory, as well as the values file customized for your environment.

Since the intent is to have ArgoCD automatically sync your configurations, it is important to first ensure you have prepared the appropriate values for your environment. When creating the application manually in argoCD you have the option to override the values but it is recommended to maintain the values in a file that can be managed and updated as needed. This method also allows for a more automated method of creating the ArgoCD application. To do this, it is recommended that you copy the default values file found at: `./helm/openshift-gitops-day2/values.yml` to a separate file, like `./helm/openshift-gitops-day2/day2-values-prod.yml`.

This template expects the following cluster specific configurations for the following in your helm values:

1. Cluster-wide certificiates
2. Networking configurations for each node in your cluster
3. LDAP sync configuration
4. Chrony configuration
5. Storage configuration

The example values file structure can be seen here

Configuring ArgoCD (Openshift Gitops)

To do this you can run the following command:

```
oc apply -f ./cluster-day2-app.yml
```

To create the ArgoCD application manually you can use similar values to the below:

- Application Name: `cluster-day2`
- Project Name: `default`
- Sync Policy: `Automatic`
- Source:

    - Repository URL: Select the `ocp4-disconnected-config` repo running on the bastion host
    - Revision: `HEAD` (or override to your desired branch)
    - Path: `./helm/openshift-gitops-day2`
- Destination:
    - Cluster URL: `https://kubernetes.default.svc` # Use this value to target the same cluster that Argo is running on.
    - Namespace: leave blank or set to default if required
- Values:
    - Override the values as needed. If Argo was able to successfully connect to the source repo configuration, then you should see a list of optional values to override. You can also upload your own `values.yaml` file.

To get to Argo, simply select it from the top right dropdown on the OpenShift web ui.

You can also use the following command to see what will be applied before actually applying:

```
helm template test . --debug | less
```

The default values.yaml is designed to be give you the basic layout, as well as provide a template for your own custom values file. Simply make a copy of this default, and start filling in your real-world data. You can even run the above template command, adding `-f /path/to/yourValues.yaml` before the `|` to tell Helm to use your new file, in order to check your work.

Components overview

nmstate operator

Relevant templates:

- nmstate-namespace.yaml
- nmstate-operatorgroup.yaml
- nmstate-operator.yaml
- nmstate-instance.yaml
- nmstate-nncp-bond.yaml
- nmstate-nncp-nodes.yaml

The nmstate operator is what brings in the NodeNetworkConfigurationPolicy CRD's , allowing us to further tweak the network setup of the cluster.

Chrony

Relevant templates:

- chrony-configuration.yaml

LDAP

Relevant templates:

- ldap-accounts,secret,and oauth provider

Storage

Relevant templates:

- trident-sc.yaml
- NOTE Will trigger a reboot of the cluster: trident-machineconfig-master.yaml
- NOTE Will trigger a reboot of the cluster: trident-machineconfig-worker.yaml

Most Kubernetes distributions come with the packages and utilities to mount NFS backends installed by default, including Red Hat OpenShift.

However, for NFSv3, there is no mechanism to negotiate concurrency between the client and the server. Hence the maximum number of client-side sunrpc slot table entries must be manually synced with supported value on the server to ensure the best performance for the NFS connection without the server having to decrease the window size of the connection.

For ONTAP, the supported maximum number of sunrpc slot table entries is 128 i.e. ONTAP can serve 128 concurrent NFS requests at a time. However, by default, Red Hat CoreOS/Red Hat Enterprise Linux has maximum of 65,536 sunrpc slot table entries per connection. We need to set this value to 128 and this can be done using Machine Config Operator (MCO) in OpenShift.

For more information see https://docs.netapp.corn/us-en/netapp-solutions/containers/rh- os-n overview trident.html#nfs

Other

Proxy Relevant templates:

- proxy-custom-ca-cm.yaml

Ingress Relevant templates:

- ingress-config.yaml

VM Migration from VMware vCenter to OpenShift Virtualization

This guide outlines the steps to migrate virtual machines (VMs) from VMware vCenter to OpenShift Virtualization. It includes setting up networking, managing VM configurations, executing the migration, and troubleshooting common issues.

Step 1: Prepare ESXi/vSphere environment

1.1 EXPAND: migrate all target vms to single esxi host

1.2 EXPAND: remove esxi host from vcenter

1.3 EXPAND: disable fast-boot and gracefully shut down windows vms

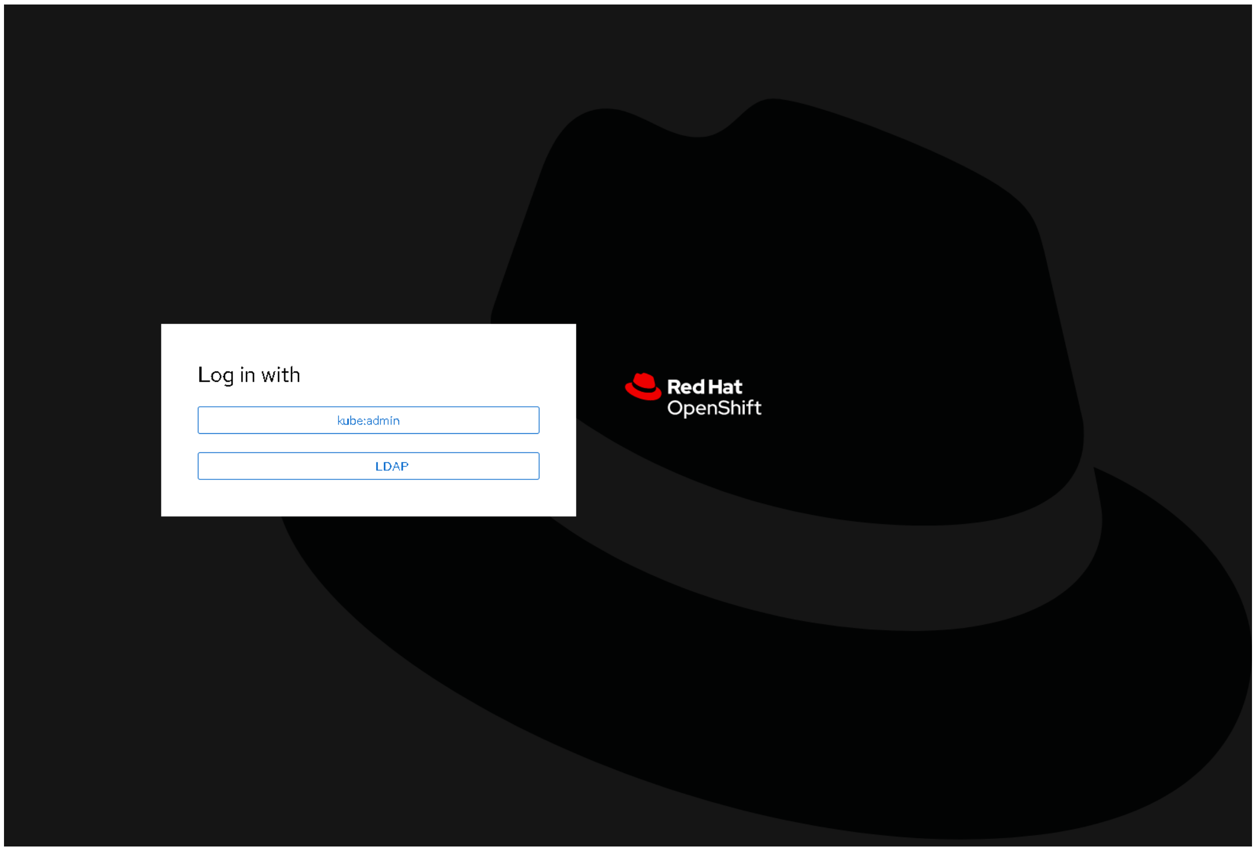
1.4 EXPAND: gracefully shut down remaining linux vms if necessary

Step 2: Prepare OpenShift Virtualization environment

First we login to the OpenShift console in the web browser.

2.1 Go to `{{OPENSHIFT_URL}}`

2.2 Click `{{LOGIN_BUTTON}}` to login



In the next few steps we are making sure the automation successfully installed some critical components

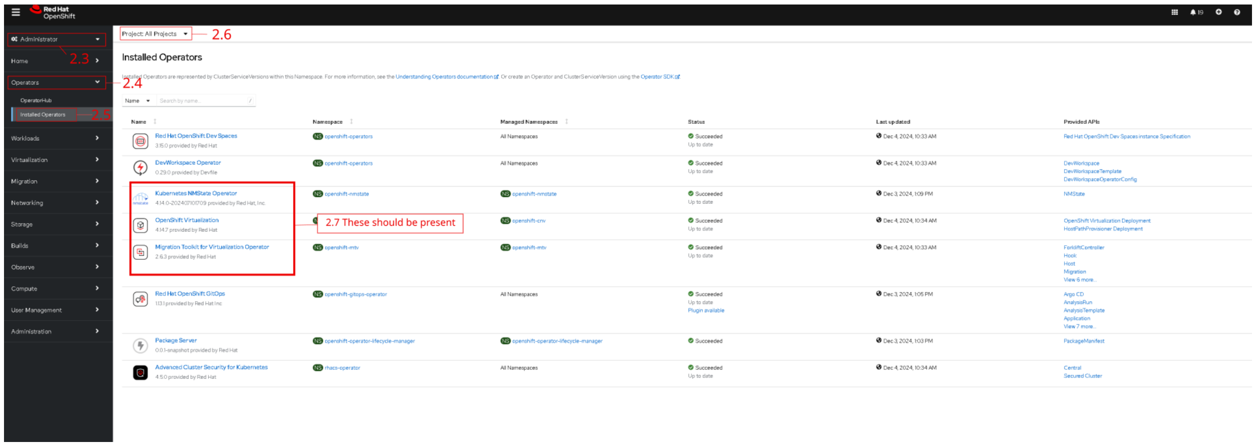
2.3 Switch to Administrator perspective (if not already)

2.4 Click `Operators` in menu on left navigation

2.5 Click `Installed Operators` in sub-menu

2.6 Ensure project is set to `All Projects` at the top

2.7 Verify you see the following - Openshift Virtualization - Migration Toolkit for Virtualization - Kubernetes NMState



Installed Operators

Installed Operators are represented by ClusterServiceVersions within this Namespace. For more information, see the Understanding Operators documentation. Or create an Operator and ClusterServiceVersion using the Operator SDK.

Name	Namespace	Managed Namespaces	Status	Last updated	Provided APIs
Red Hat OpenShift Dev Spaces 3.15.0 provided by Red Hat	NS openshift-operators	All Namespaces	Succeeded Up to date	Dec 4, 2024, 10:33 AM	Red Hat OpenShift Dev Spaces instance Specification
DevWorkspace Operator 0.29.0 provided by Devfile	NS openshift-operators	All Namespaces	Succeeded Up to date	Dec 4, 2024, 10:33 AM	DevWorkspace DevWorkspaceTemplate DevWorkspaceOperatorConfig
Kubernetes NMState Operator 4.14.0-202407101709 provided by Red Hat, Inc.	NS openshift-nmstate	NS openshift-nmstate	Succeeded Up to date	Dec 3, 2024, 1:09 PM	NMState
OpenShift Virtualization 4.14.7 provided by Red Hat	[illegible]	NS openshift-cnv	Succeeded Up to date	Dec 4, 2024, 10:34 AM	OpenShift Virtualization Deployment HostPathProvisioner Deployment
Migration Toolkit for Virtualization Operator 2.6.3 provided by Red Hat	NS openshift-mtv	NS openshift-mtv	Succeeded Up to date	Dec 4, 2024, 10:33 AM	ForkliftController Hook Host Migration View 6 more...
Red Hat OpenShift GitOps 1.13.1 provided by Red Hat Inc	NS openshift-gitops-operator	All Namespaces	Succeeded Up to date Plugin available	Dec 3, 2024, 1:05 PM	Argo CD AnalysisRun AnalysisTemplate Application View 7 more...
Package Server 0.0.1-snapshot provided by Red Hat	NS openshift-operator-lifecycle-manager	NS openshift-operator-lifecycle-manager	Succeeded	Dec 3, 2024, 1:03 PM	PackageManifest
Advanced Cluster Security for Kubernetes 4.5.0 provided by Red Hat	NS rhacs-operator	All Namespaces	Succeeded Up to date	Dec 4, 2024, 10:34 AM	Central Secured Cluster

2.8 Click `Networking` in the left navigation menu

2.9 Click `NodeNetworkConfigurationPolicy`

2.10 Verify you see the following objects: - bond0 - etc

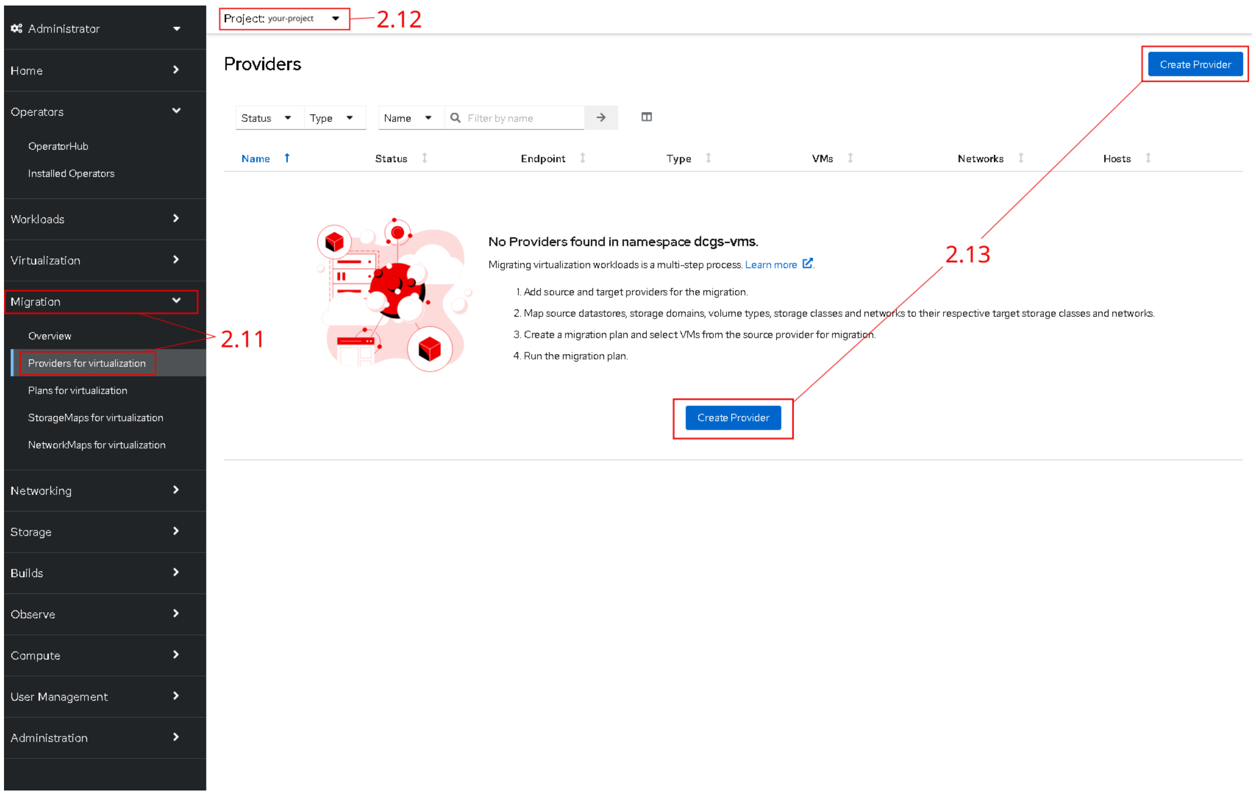
Now we will configure the `Migration Toolkit` operator so that it can see the source host where the vms will be migrated from

Note: you will need the username and login for the ESXi host

2.11 Click `Migration` –> `Providers for virtualization` in the left navigation menu

2.12 Ensure Project is set to `{{PROJECT_NAME}}` at the top

2.13 Click `Create Provider`

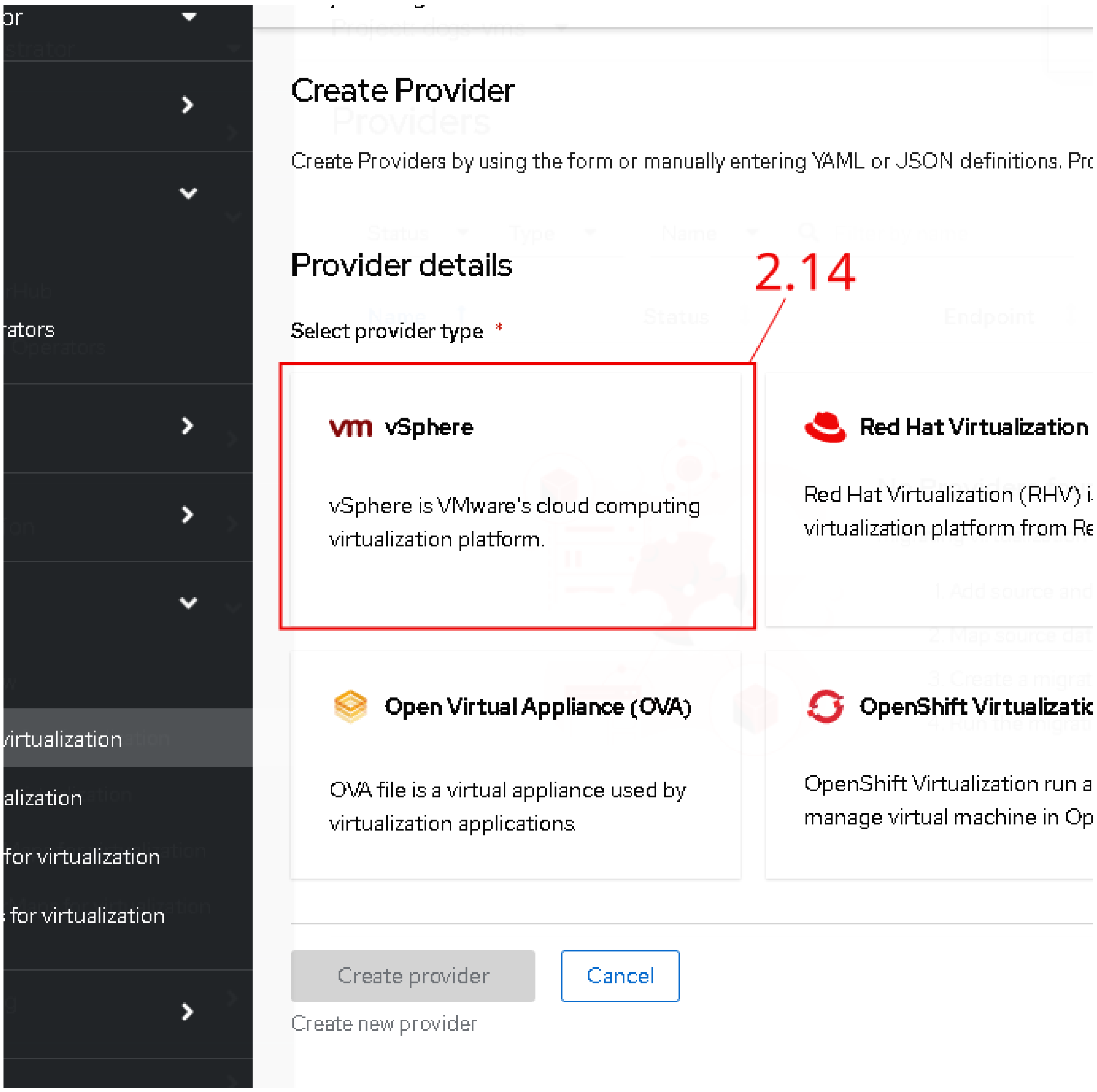


No Providers found in namespace dcgs-vms.

Migrating virtualization workloads is a multi-step process. Learn more

1. Add source and target providers for the migration.
2. Map source datastores, storage domains, volume types, storage classes and networks to their respective target storage classes and networks.
3. Create a migration plan and select VMs from the source provider for migration.
4. Run the migration plan.

2.14 Select `VM vSphere` option under `Provider details`


Create Provider

Create Providers by using the form or manually entering YAML or JSON definitions. Pro

Provider details

Select provider type *

vSphere

vSphere is VMware's cloud computing virtualization platform.

Red Hat Virtualization

Red Hat Virtualization (RHV) i virtualization platform from Re

Open Virtual Appliance (OVA)

OVA file is a virtual appliance used by virtualization applications.

OpenShift Virtualizatio

OpenShift Virtualization run a manage virtual machine in Op

Create new provider

2.15 Type `esxi-host` for the `Provider resource name` field (all lowercase and no spaces)

2.16 Change the `Endpoint type` from vCenter to `ESXi`

2.17 Type in the `URL` (i.e. https://1.1.1.1/sdk)

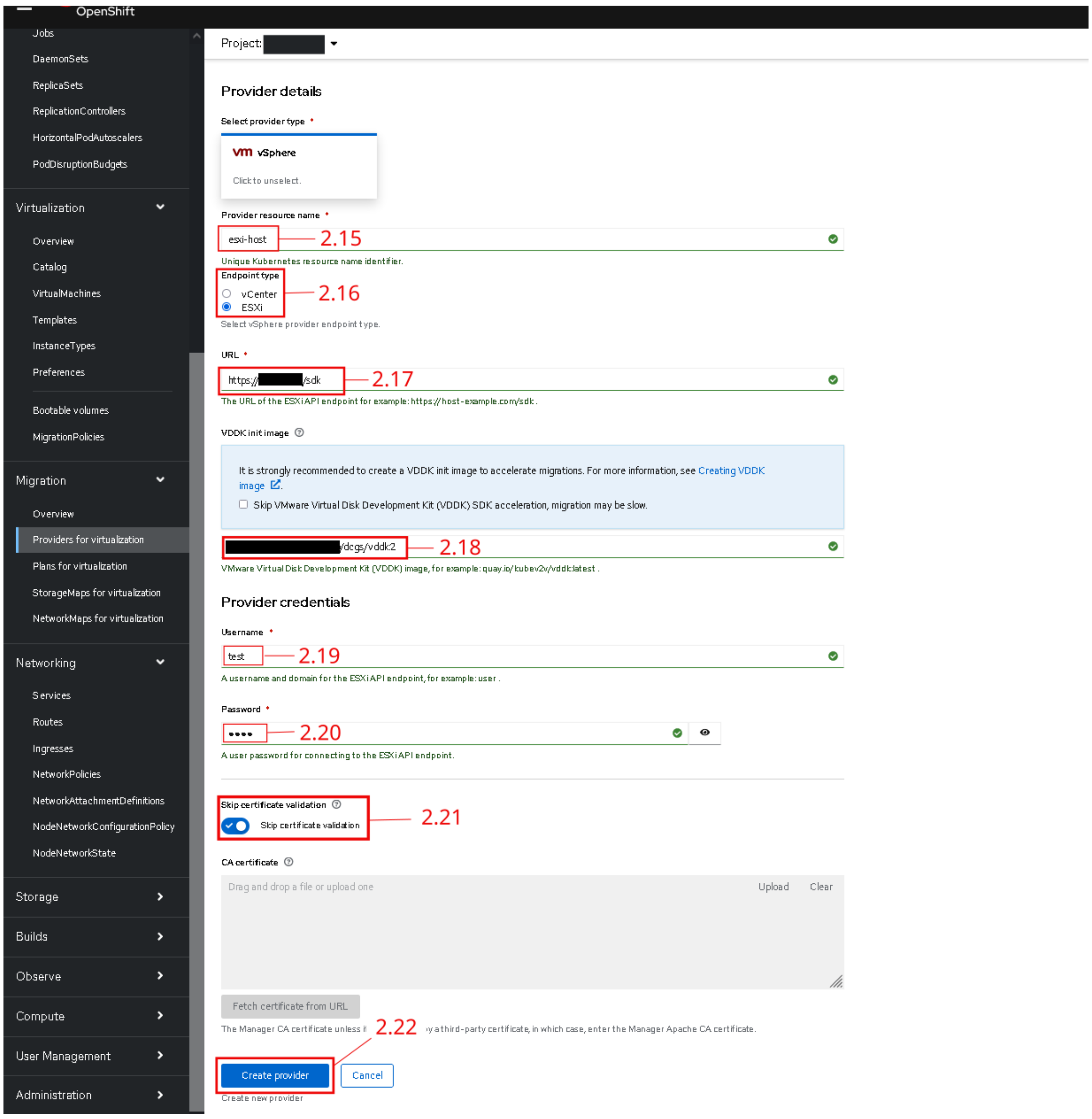
2.18 Type in `{{SYSCON_IP_ADDR}}/dcgs/vddk:2` for the `VDDK init image` path field

2.19 Type in the ESXi username for `Username`

2.20 Type in the ESXi password for the `Password`

2.21 Click the `Skip certificate validation` radio button

2.22 Click the blue `Create provider` button



Project: [redacted]

Provider details

Select provider type *

vm vSphere

Click to unselect.

Provider resource name *

esxi-host — 2.15

Unique Kubernetes resource name identifier.

Endpoint type — 2.16

vCenter [✓ ESXi]

Select vSphere provider endpoint type.

URL *

https://[redacted]/sdk — 2.17

The URL of the ESXi API endpoint for example: https://host-example.com/sdk .

VDDK init image

It is strongly recommended to create a VDDK init image to accelerate migrations. For more information, see Creating VDDK image.

☐ Skip VMware Virtual Disk Development Kit (VDDK) SDK acceleration, migration may be slow.

[redacted]/dcgs/vddk:2 — 2.18

VMware Virtual Disk Development Kit (VDDK) image, for example: quay.io/kubev2v/vddk:latest .

Provider credentials

Username *

test — 2.19

A username and domain for the ESXi API endpoint, for example: user .

Password *

•••• — 2.20

A user password for connecting to the ESXi API endpoint.

Skip certificate validation — 2.21

[✓ Skip certificate validation]

CA certificate

Drag and drop a file or upload one Upload Clear

Fetch certificate from URL

The Manager CA certificate unless i 2.22 y a third-party certificate, in which case, enter the Manager Apache CA certificate.

Create provider Cancel

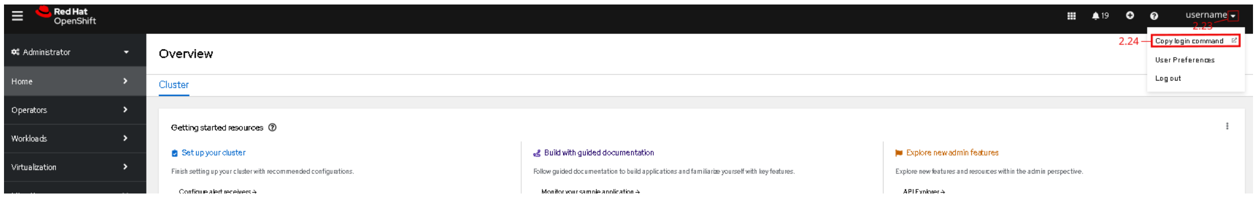
Create new provider

Now we will configure the `Migration Toolkit` operator so that it can see the OpenShift virtualization environment where the vms will be migrated to

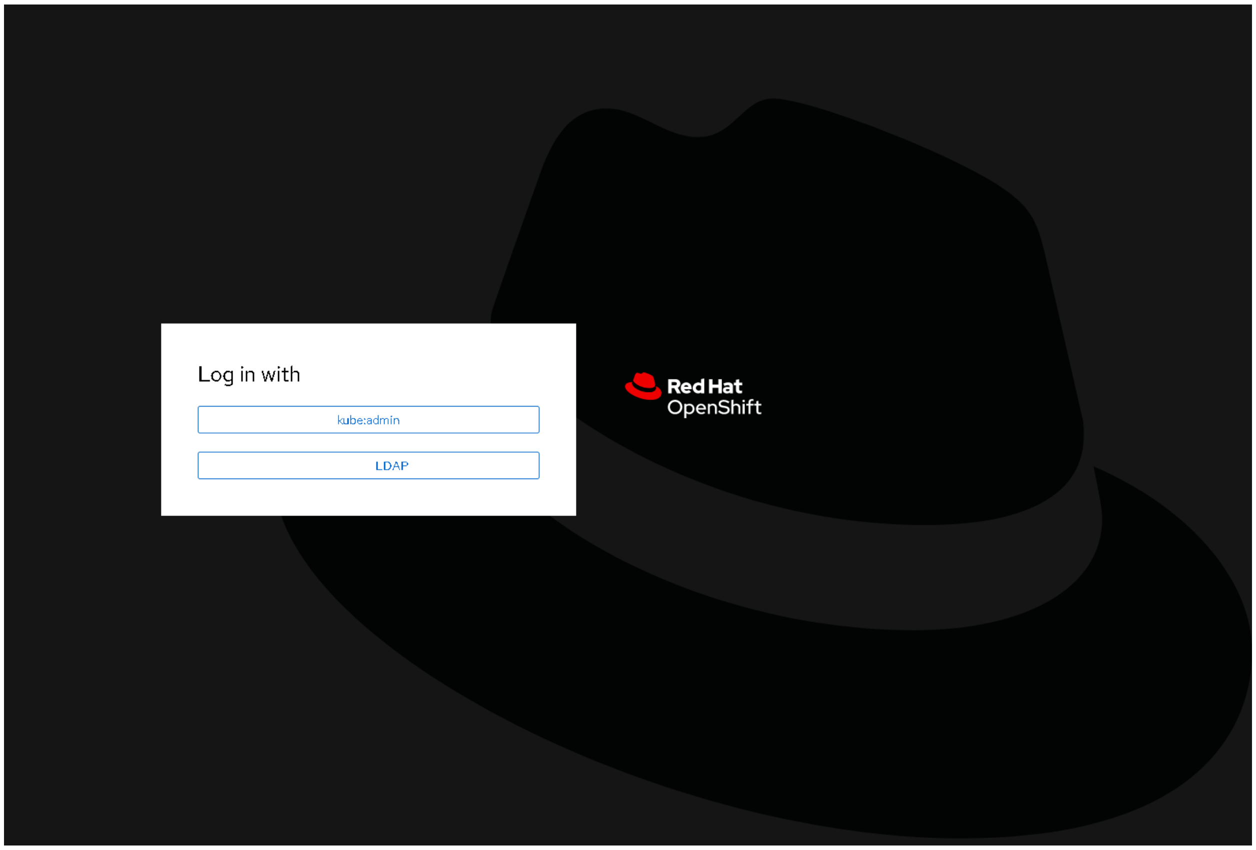
In the first few steps we will get the OpenShift API token for your user so that we can add that credential. Note: some steps will be a repeat from above.

2.23 Click the down arrow next to your username at the top right

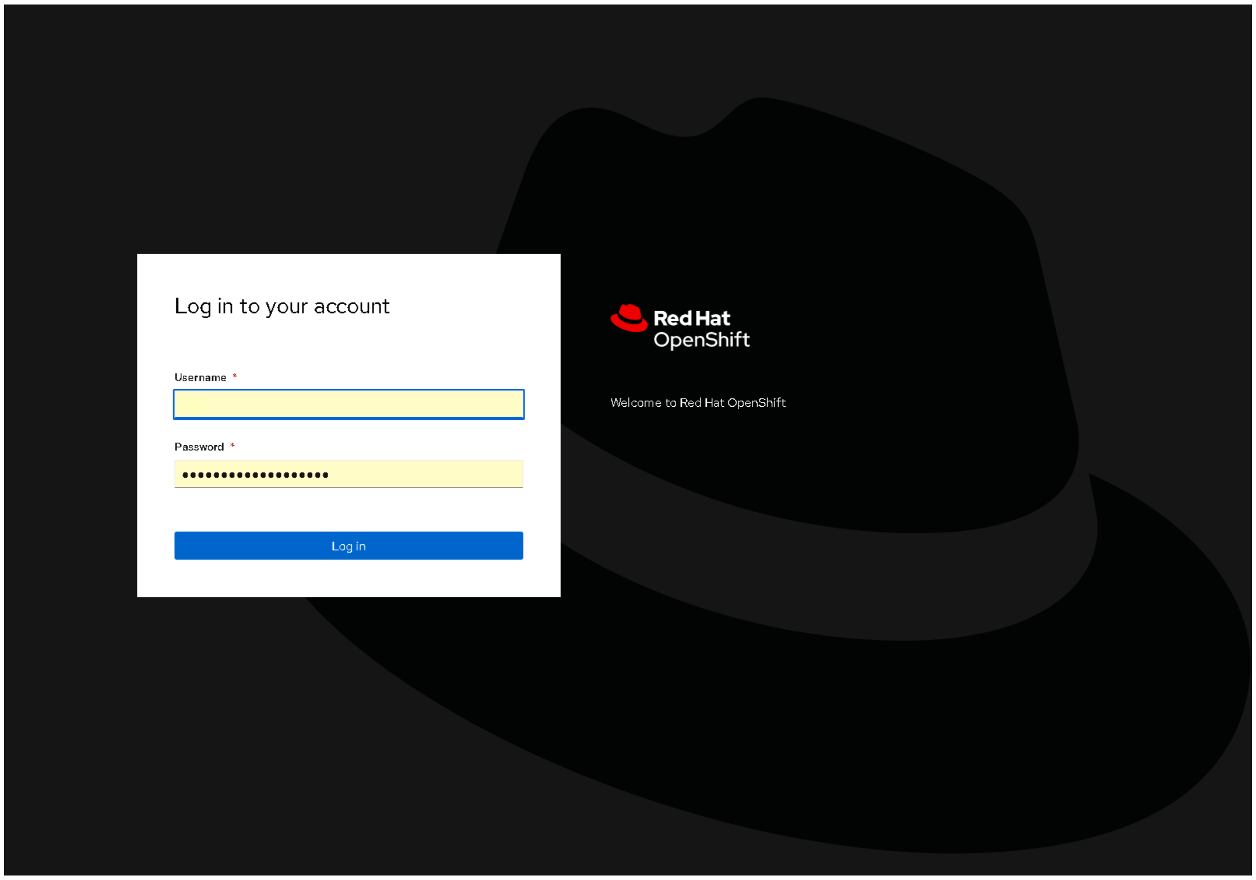
2.24 Click `Copy Login Command`


Red Hat OpenShift
19
username
2.23
2.24
Copy login command
User Preferences
Log out
Administrator
Home
Operators
Workloads
Virtualization
Overview
Cluster
Getting started resources
Set up your cluster
Finish setting up your cluster with recommended configurations.
Configure alert receivers →
Build with guided documentation
Follow guided documentation to build applications and familiarize yourself with key features.
Monitor your sample application →
Explore new admin features
Explore new features and resources within the admin perspective.
API Explorer →

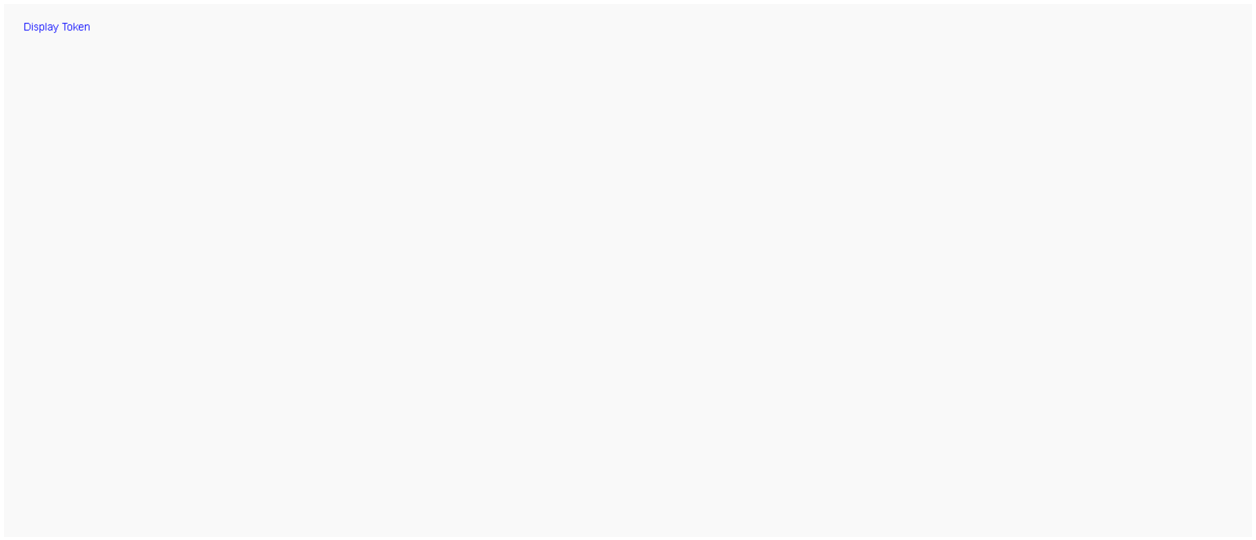
2.25 Click `{{LOGIN_BUTTON}}` to verify login



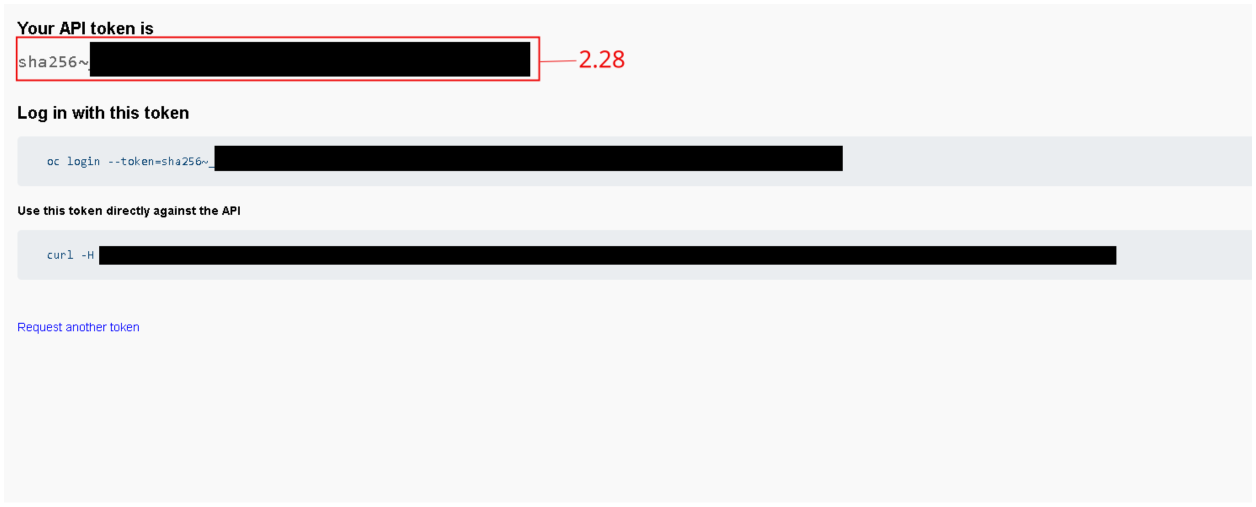
2.26 Type in your username and password and click `Login`



2.27 Click `Display Token`



2.28 Copy the line where your `sha256~....` token is displayed



Your API token is

sha256~[redacted]

Log in with this token

```
oc login --token=sha256~[redacted]
```

Use this token directly against the API

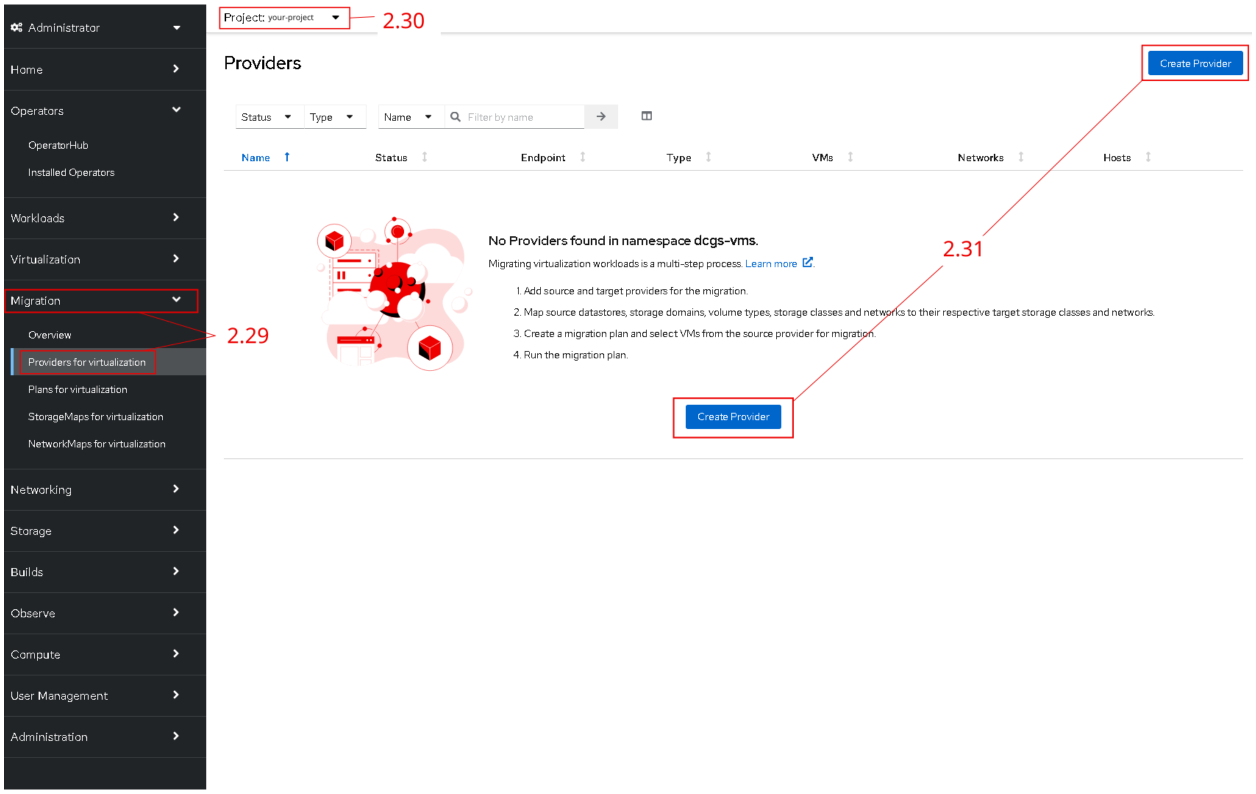
```
curl -H [redacted]
```

Request another token

2.29 Click `Migration` –> `Providers for virtualization` in the left navigation menu. NOTE: This is a second provider being added, so some steps will be a repeat from earlier.

2.30 Ensure Project is set to `{{PROJECT_NAME}}` at the top

2.31 Click `Create Provider`



No Providers found in namespace dcgs-vms.

Migrating virtualization workloads is a multi-step process. Learn more

1. Add source and target providers for the migration.
2. Map source datastores, storage domains, volume types, storage classes and networks to their respective target storage classes and networks.
3. Create a migration plan and select VMs from the source provider for migration.
4. Run the migration plan.

2.32 Select `Openshift Virtualization` for the provider type

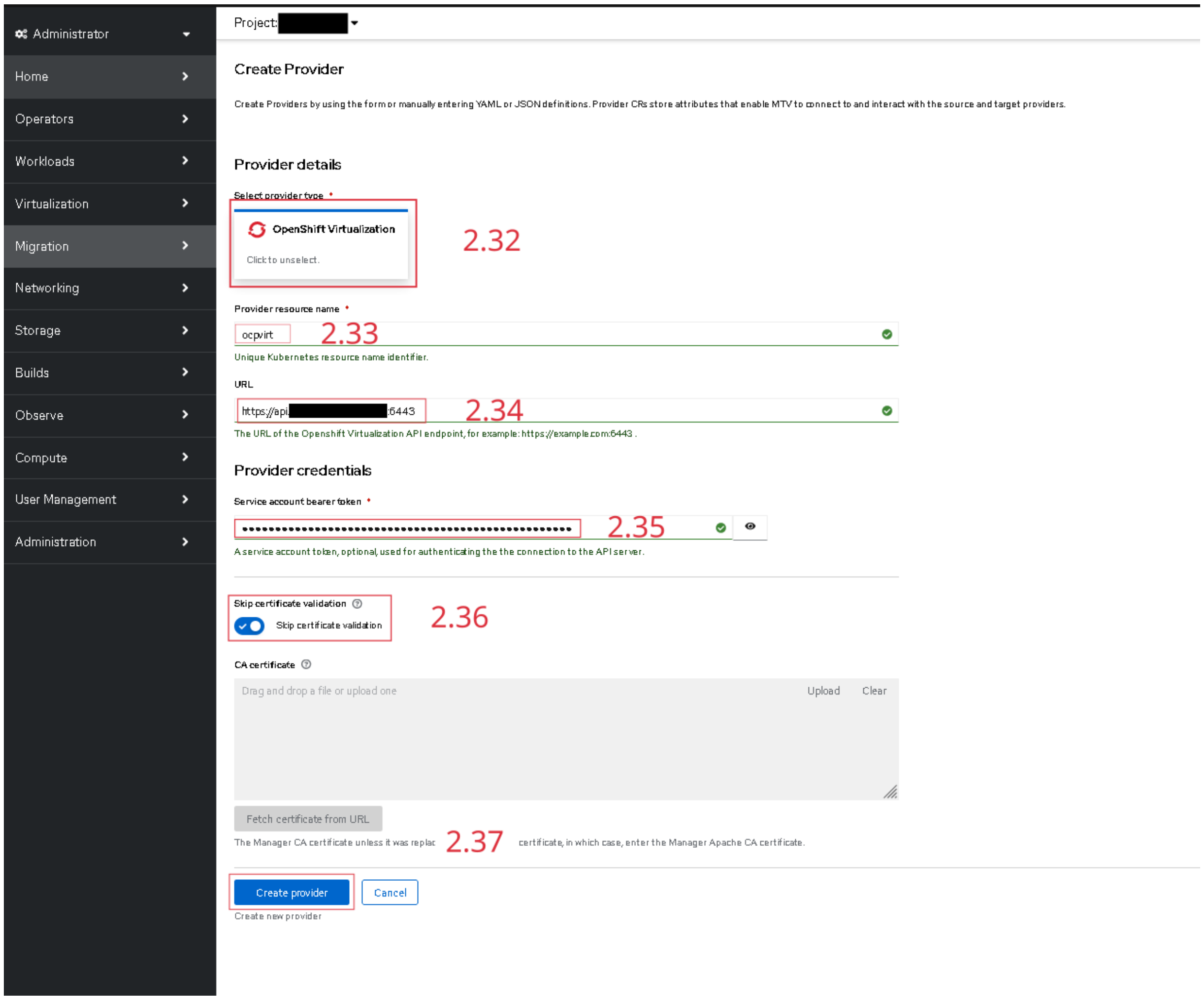
2.33 Type `ocpvirt` for the `Provider resource name`

2.34 Type `{{OPENSHIFT_CLUSTER_API_URL}}` for the URL

2.35 Paste in your sha256~ token value that you copied earlier into the `Service account bearer token` field

2.36 Click the `Skip certificate validation` radio button

2.37 Click `Create Provider`



Project: [redacted]

Create Provider

Create Providers by using the form or manually entering YAML or JSON definitions. Provider CRs store attributes that enable MTV to connect to and interact with the source and target providers.

Provider details

Select provider type *

OpenShift Virtualization

Click to unselect.

Provider resource name *

ocpvirt

Unique Kubernetes resource name identifier.

URL

https://api.[redacted]:6443

The URL of the Openshift Virtualization API endpoint, for example: https://example.com:6443 .

Provider credentials

Service account bearer token *

•••

A service account token, optional, used for authenticating the the connection to the API server.

Skip certificate validation

Skip certificate validation

CA certificate

Drag and drop a file or upload one

Upload Clear

Fetch certificate from URL

The Manager CA certificate unless it was replac certificate, in which case, enter the Manager Apache CA certificate.

Create provider Cancel

Create new provider

(1) Click `Operators` –> `Installed Operators` in the left navigation menu
(2) Ensure project is set to `All Projects` at the top
(3) Select the `Migration Toolkit for Virtualization Operator` option
(4) Switch to the `ForkliftController` tab
(5) Select the `FC forklift-controller` resource
(6) Switch to the `YAML` tab at the top and wait for it to load (text will colorize once resource is loaded)
(7) Type `controller_max_vm_inflight: 5` immediately following the `spec:` line. (ensure proper spacing allignment)
(8) Click `Save`
(9) Click `Migration` –> `Plans for virtualization` in the left navigation menu
(10) Ensure project is set to `dcgs-vms` at the top
(11) Click `Create Plan`
(12) Select `esxi-host` source provider
(13) Once the list of available VMs is loaded, change the number of displayed results to 100 for easier selection process
(14) Select vms intended to be migrated over
(15) Click `Next`
(16) Type `dcgs-vm-migration-initial` for the `Plan name` field. (all lowercase and no spaces)

(17) Verify the number of `Selected VMs` is what you expect
(18) Select `ocpvirt` for the `Target provider`
(19) Ensure `dcgs-vms` is selected for `Target namespace`
(20) Complete the network mappings by matching the appropriate vlan ids
(21) Verify storage map is set to `basic-csi` on the right.
(22) Click `Create migration plan`
(23) Wait for Migration Plan to be validated (you can expect to see a Warning flag and this validation may take several minutes before the next button is available)
(24) Once validation is complete, click `Start Migration` button.
(25) Once the migration has started the page should display some progress bars
(26) Switch to `Virtual Machines` tab to monitor progress
(27) Once all VMs have been migrated, go back to syscon console
(28) Change directory to the config repo if not already there (i.e. /opt/syscon/ocp4-disconnected-config)
(29) Run command `./scripts/update-drivers.sh`
(30) Go back to the Openshift Console in the browser window
(31) Select `Virtualization` –> `VirtualMachines` in the left navigation menu
(32) Ensure the project is set to `dcgs-vms` at the top
(33) You should see a list of your migrated virtual machines on this screen and be able to click on them to manage them.

5 Troubleshooting and more information

6 Alternate migration method (manual ova copy instructions)