

ACM install platform none clusters

| | |
|---|-----------|
| Installing a hybrid cluster | 1 |
| IPI cluster installation | 2 |
| Infra nodes | 7 |
| Moving the router pods to the infra nodes | 11 |
| Moving the registry pods to the infra nodes | 12 |
| Moving the monitoring stack to the infra nodes | 12 |
| Other elements that need to be moved to infra nodes | 14 |
| RHACM Operator Deployment | 14 |
| Mirror Registry | 18 |
| Setting up the environment | 18 |
| Installing the mirror registry software | 20 |
| Mirroring packages | 24 |
| Operator Catalog Content | 29 |
| Managed cluster environment configuration | 35 |
| Create a Host inventory | 38 |
| Create new VMs | 43 |
| Advance network configuration | 53 |
| Download the discovery ISO | 57 |
| Upload the discovery ISO to vsphere | 58 |
| Boot the VMs with the discovery ISO | 59 |
| Create the cluster | 63 |
| TODO | 70 |
| Cluster Image Sets | 70 |
| Install a cluster with customizations | 70 |

[Is it supported to have mixed environment \(VMware+Baremetal\) setup for RHOC P 4.x cluster?](#)

The next link could be outdated because the support is supposed to be available in 4.14, according to the Epic. But the docs still state that platform:none is only supported on SNO.

<https://access.redhat.com/solutions/7008469>

There is a documentation bug opened for that

<https://issues.redhat.com/browse/OCPBUGS-29306>

The final response is that **YES**, it is supported

Installing a hybrid cluster

Create 2 **VMware Cloud Public Cloud Open Environment** in demo.redhat.com.

vm

VMware Cloud Public Cloud Open Environment

provided by RHDP

Order

Category

Open Environments

Provider

RHDP

Product Family

Red Hat Cloud

Rating

★★★★★

(124)

Last update

39 days ago

Last successful provision

25 hours ago

Auto-Stop

24 Hours

Auto-Destroy

48 Hours

Deploy The Assisted Installer Via Ansible Navigator

provided by RH Community

Practice installing Red Hat® OpenShift™ on RHDP's VMware Cloud Open Environment service. The 'ocpd-ai-e-

★★★★★

(5)

Description

VMC (VMWare Cloud) Sandbox

Welcome to the Red Hat Open Environment Project

The Open Environments project gives Red Hatters on demand access to a number of different cloud services. These timebound sandboxes offer access to many of the clouds that our customers use, and will enable associates to test, create, and demo the technologies needed to help make our customers and account teams successful. These Sandboxes will allow users to do things not previously offered.

Always remember, "With Great Power, Comes Great Responsibility"

Note:

By ordering this sandbox, the user takes full responsibility for the running of their environment:

- The costs of services ordered
- The security of any credentials provided
- Approval and permission for usage
- Reporting of any issues or anomalies due to usage
- Adherence to Red Hat Compliance and Ethics Policies
- This is for internal use only and not to be shared with customers or partners

Current Offerings:

- AWS - Blank Sandbox
- IBM - Blank Sandbox
- VMware Cloud (VMC) - Sandbox with a bastion

Clusters Specification - IBM Cloud Sandbox

- Access to VCenter from outside is only from one IP specified ordering
- Only one flat network is available

Provisioning Time?

- ~ 20 Min

Default Lifespan of Environment?

- 2 Days
- Extensions Available - Yes

Support Options -

- There is limited support for the Open Environment Sandboxes.

For more information please see <https://content.redhat.com/us/en/product/rhdp.html#>

Copyright © 2023 Red Hat, Inc.

Privacy statement

One will be used to deploy an IPI OCP 4.14 cluster hub cluster, the other will be used to create the managed cluster.

IPI cluster installation

Deploy an IPI cluster on vsphere.

Make sure to enable DNS records creation for OCP.

Enable DNS records for OCP

☒ Enable DNS records for OCP ? Enable api.apps and *.apps

Auto stop

It takes about 20 minutes for the environment to be provisioned and ready.

When the environment is ready an email is received with important information.

Ssh into the bastion node.

Verify the DNS records for the cluster:

Unset

```
$ dig +short api.glnm2.dynamic.opentlc.com
3.223.59.140
$ dig +short *.apps.glnm2.dynamic.opentlc.com
34.198.235.139
```

Get the installer, oc client and pull secret from [here](#) and copy them to the bastion host.
Uncompress the tar files and put them in the running path:

Unset

```
$ scp ~/Descargas/openshift-* /home/jjerezro/Descargas/pull-secret.txt \
  lab-user@bastion-glnm2.glnm2.dynamic.opentlc.com

$ tar xvf openshift-client-linux.tar.gz
README.md
oc
kubectl

$ tar xvf openshift-install-linux.tar.gz
README.md
openshift-install

$ sudo cp -vi openshift-install oc /usr/local/bin
'openshift-install' -> '/usr/local/bin/openshift-install'
'oc' -> '/usr/local/bin/oc'
```

Create an ssh key pair

Unset

```
$ ssh localhost
The authenticity of host 'localhost (::1)' can't be established.
ECDSA key fingerprint is SHA256:AI0gCA9BvbsRJN9NSt0jqJ6xEd4pjlcqyHPwF9aLr3Q.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'localhost' (ECDSA) to the list of known hosts.
lab-user@localhost's password:

$ ssh-keygen -t ed25519 -N '' -f ~/.ssh/ocp
Generating public/private ed25519 key pair.
Your identification has been saved in /home/lab-user/.ssh/ocp.
Your public key has been saved in /home/lab-user/.ssh/ocp.pub.
The key fingerprint is:
...
```

Download the vCenter's root CA certificates. Go to the base URL of the vCenter, for example <https://portal.vc.opentlc.com/>

Click on the link Download trusted root CA certificates on the right side of the web page

The screenshot shows the top portion of the VMware vSphere Documentation Center. The VMware logo is in the top left. Below it are links for 'Getting Started' and 'Documentation'. A blue button labeled 'LAUNCH VSPHERE CLIENT' is also visible. On the right side, there is a sidebar with sections for 'For Administrators' and 'For Developers'. Under 'For Administrators', there is a link for 'Web-Based Datastore Browser'. Under 'For Developers', there are links for 'vSphere Web Services SDK', 'Learn more about the Web Services SDK', 'Learn more about vSphere Automation SDKs', 'Browse objects managed by vSphere', 'Browse vSphere REST APIs', and a blue button labeled 'Download trusted root CA certificates'.

vmware

Getting Started

LAUNCH VSPHERE CLIENT

Documentation

VMware vSphere Documentation Center

For Administrators

Web-Based Datastore Browser

Use your web browser to find and download files (for example, virtual machine and virtual disk files).

[Browse datastores in the vSphere inventory](#)

For Developers

vSphere Web Services SDK

Learn about our latest SDKs, Toolkits, and APIs for managing VMware ESXi and VMware vCenter. Get sample code, reference documentation, participate in our Forum Discussions, and view our latest Sessions and Webinars.

[Learn more about the Web Services SDK](#)

[Learn more about vSphere Automation SDKs](#)

[Browse objects managed by vSphere](#)

[Browse vSphere REST APIs](#)

[Download trusted root CA certificates](#)

The screenshot shows the main content area of the VMware vSphere Documentation Center. It has a blue header bar at the top. Below it, there is a section for 'For Administrators' with a sub-section 'Web-Based Datastore Browser'. This section describes using a web browser to find and download files and provides a link to 'Browse datastores in the vSphere inventory'. Below this is a section for 'For Developers' with a sub-section 'vSphere Web Services SDK'. This section describes learning about SDKs, Toolkits, and APIs for managing VMware ESXi and VMware vCenter, and provides links to 'Learn more about the Web Services SDK', 'Learn more about vSphere Automation SDKs', 'Browse objects managed by vSphere', 'Browse vSphere REST APIs', and a blue button labeled 'Download trusted root CA certificates'.

For Administrators

Web-Based Datastore Browser

Use your web browser to find and download files (for example, virtual machine and virtual disk files).

[Browse datastores in the vSphere inventory](#)

For Developers

vSphere Web Services SDK

Learn about our latest SDKs, Toolkits, and APIs for managing VMware ESXi and VMware vCenter. Get sample code, reference documentation, participate in our Forum Discussions, and view our latest Sessions and Webinars.

[Learn more about the Web Services SDK](#)

[Learn more about vSphere Automation SDKs](#)

[Browse objects managed by vSphere](#)

[Browse vSphere REST APIs](#)

[Download trusted root CA certificates](#)

The copy the downloaded file to the bastion host
Extract the file in the bastion host

Unset

```
$ unzip download.zip
Archive: download.zip
  inflating: certs/lin/7255df92.0
  inflating: certs/mac/7255df92.0
  inflating: certs/win/7255df92.0.crt
  inflating: certs/lin/75c43eb5.r0
  inflating: certs/mac/75c43eb5.r0
  inflating: certs/win/75c43eb5.r0.crl
...
```

Update your system trust

Unset

```
$ sudo cp -vi certs/lin/* /etc/pki/ca-trust/source/anchors
'certs/lin/02265526.0' -> '/etc/pki/ca-trust/source/anchors/02265526.0'
'certs/lin/2835d715.0' -> '/etc/pki/ca-trust/source/anchors/2835d715.0'
...

$ sudo update-ca-trust extract
```

Create the install-config.yaml file. To get the initial configuration that later needs to be modified use the command.

The information required is in the email from RHDP. For the VIP for API and Ingress use the NAT IP described in the email.

Unset

```
$ openshift-install create install-config
? SSH Public Key /home/lab-user/.ssh/ocp.pub
? Platform vsphere
? vCenter vcenter.sddc-44-197-86-61.vmwarevmc.com
? Username sandbox-glnm2@vc.opentlc.com
? Password [? for help] *****
INFO Connecting to vCenter vcenter.sddc-44-197-86-61.vmwarevmc.com
INFO Defaulting to only available datacenter: SDDC-Datacenter
INFO Defaulting to only available cluster: /SDDC-Datacenter/host/Cluster-1
INFO Defaulting to only available datastore: /SDDC-Datacenter/datastore/WorkloadDatastore
INFO Defaulting to only available network: segment-sandbox-glnm2
? Virtual IP Address for API 192.168.95.201
? Virtual IP Address for Ingress 192.168.95.202
? Base Domain dynamic.opentlc.com
? Cluster Name glnm2
? Pull Secret [? for help] *****...
INFO Install-Config created in: .
```

Edit the `install-config.yaml` file and add the `vcenter` folder where the VMs will be created

```
Unset
...
platform:
  vsphere:
    apiVIPs:
      - 192.168.95.201
    failureDomains:
      - name: generated-failure-domain
    region: generated-region
    server: vcenter.sddc-44-197-86-61.vmwarevmc.com
    topology:
      computeCluster: /SDDC-Datacenter/host/Cluster-1
      datacenter: SDDC-Datacenter
      datastore: /SDDC-Datacenter/datastore/WorkloadDatastore
      networks:
        - segment-sandbox-glnm2
      resourcePool: /SDDC-Datacenter/host/Cluster-1/Resources
      folder: /SDDC-Datacenter/vm/Workloads/sandbox-glnm2
      zone: generated-zone
...
```

If this is a disconnected installation, follow the instructions in section Mirror Registry to install the local registry and mirror the installation images.

Create a directory with the name of the cluster in the bastion host and copy the `install-config.yaml` file there:

```
Unset
$ mkdir glnm2
$ cp install-config.yaml glnm2/
```

Run the installer

```
Unset
$ openshift-install create cluster --dir glnm2/
INFO Consuming Install Config from target directory
INFO Creating infrastructure resources...
INFO Waiting up to 20m0s (until 3:21AM EST) for the Kubernetes API at
https://api.glnm2.dynamic.opentlc.com:6443...
INFO API v1.28.6+6216ea1 up
INFO Waiting up to 1h0m0s (until 4:04AM EST) for bootstrapping to complete...
INFO Destroying the bootstrap resources...
```

```
INFO Waiting up to 40m0s (until 3:58AM EST) for the cluster at
https://api.glnm2.dynamic.opentlc.com:6443 to initialize...
INFO Waiting up to 30m0s (until 4:03AM EST) to ensure each cluster operator has finished
progressing...
INFO All cluster operators have completed progressing
INFO Checking to see if there is a route at openshift-console/console...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/lab-user/glnm2/auth/kubeconfig'
INFO Access the OpenShift web-console here:
https://console-openshift-console.apps.glnm2.dynamic.opentlc.com
INFO Login to the console with user: "kubeadmin", and password: "7azGi-RUI6u-HR8ph-PuRyH"
INFO Time elapsed: 36m40s
```

Infra nodes

<https://access.redhat.com/solutions/5034771>

To have infra nodes created at installation time (day 1 configuration) start by creating the manifests from the install-config.yaml file

```
Unset
$ openshift-install create manifests --dir glnm2/
INFO Consuming Install Config from target directory
INFO Manifests created in: glnm2/manifests and glnm2/openshift
```

One of the manifests defines the worker machineset

```
Unset
$ ls glnm2/openshift/*machineset*
glnm2/openshift/99_openshift-cluster-api_worker-machineset-0.yaml
```

In this example a new machineset for infra nodes is created, resulting in 2 machinesets: one for workers and one for infra. See later for an example in which the worker machineset is replaced by an infra machineset

Copy the worker machineset as the infra machineset

```
Unset
$ cd glnm2/openshift/
```

```
$ cp 99_openshift-cluster-api_worker-machineset-0.yaml \
  99_openshift-cluster-api_infra-machineset-0.yaml
```

Modify the infra machineset according to the [documentation](#).

The amount of disk, CPU, and memory resources can be specified in the machineset. Keep the reference for the worker user data

The final result looks like this:

```
Unset
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  creationTimestamp: null
  labels:
    machine.openshift.io/cluster-api-cluster: glnm2-7jvd8
  name: glnm2-7jvd8-infra-0
  namespace: openshift-machine-api
spec:
  replicas: 2
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: glnm2-7jvd8
      machine.openshift.io/cluster-api-machineset: glnm2-7jvd8-infra-0
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: glnm2-7jvd8
        machine.openshift.io/cluster-api-machine-role: infra
        machine.openshift.io/cluster-api-machine-type: infra
        machine.openshift.io/cluster-api-machineset: glnm2-7jvd8-infra-0
    spec:
      lifecycleHooks: {}
      metadata:
        labels:
          node-role.kubernetes.io/infra: ""
      taints:
        - key: node-role.kubernetes.io/infra
          effect: NoSchedule
      providerSpec:
        value:
          apiVersion: machine.openshift.io/v1beta1
          credentialsSecret:
            name: vsphere-cloud-credentials
          diskGiB: 120
          kind: VSphereMachineProviderSpec
          memoryMiB: 8192
```



```

metadata:
  creationTimestamp: null
network:
  devices:
    - networkName: segment-sandbox-glnm2
numCPUs: 4
numCoresPerSocket: 4
  snapshot: ""
  template: glnm2-7jvd8-rhcos-generated-region-generated-zone
userDataSecret:
  name: worker-user-data
workspace:
  datacenter: SDDC-Datacenter
  datastore: /SDDC-Datacenter/datastore/WorkloadDatastore
  folder: /SDDC-Datacenter/vm/Workloads/sandbox-glnm2
  resourcePool: /SDDC-Datacenter/host/Cluster-1/Resources
  server: vcenter.sddc-44-197-86-61.vmwarevmc.com

```

In case of a cluster with no worker nodes, in which only infra workloads are going to be run, replace the worker machineset by an infra machineset and leave the taint out of the infra nodes.

After creating the manifests edit the manifest defining the machineset for workers like this. In the example below the **taint** is not present. Keep the reference for the worker user data:

```

Unset
$ mv 99_openshift-cluster-api_worker-machineset-0.yaml \
  99_openshift-cluster-api_infra-machineset-0.yaml

$ vim 99_openshift-cluster-api_infra-machineset-0.yaml
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  creationTimestamp: null
  labels:
    machine.openshift.io/cluster-api-cluster: glnm2-ln65x
  name: glnm2-ln65x-infra-0
  namespace: openshift-machine-api
spec:
  replicas: 3
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: glnm2-ln65x
      machine.openshift.io/cluster-api-machineset: glnm2-ln65x-infra-0
  template:
    metadata:

```

```

labels:
  machine.openshift.io/cluster-api-cluster: glnm2-ln65x
  machine.openshift.io/cluster-api-machine-role: infra
  machine.openshift.io/cluster-api-machine-type: infra
  machine.openshift.io/cluster-api-machineset: glnm2-ln65x-infra-0
spec:
  lifecycleHooks: {}
  metadata:
    labels:
      node-role.kubernetes.io/infra: ""
  providerSpec:
    value:
      apiVersion: machine.openshift.io/v1beta1
      credentialsSecret:
        name: vsphere-cloud-credentials
      diskGiB: 120
      kind: VSphereMachineProviderSpec
      memoryMiB: 16384
      metadata:
        creationTimestamp: null
      network:
        devices:
          - networkName: segment-sandbox-glnm2
      numCPUs: 4
      numCoresPerSocket: 4
      snapshot: ""
      template: glnm2-ln65x-rhcos-generated-region-generated-zone
      userDataSecret:
        name: worker-user-data
      workspace:
        datacenter: SDDC-Datacenter
        datastore: /SDDC-Datacenter/datastore/WorkloadDatastore
        folder: /SDDC-Datacenter/vm/Workloads/sandbox-glnm2
        resourcePool: /SDDC-Datacenter/host/Cluster-1/Resources
        server: vcenter.sddc-44-197-86-61.vmwarevmc.com

```

Run the installer again.

Unset

```

$ openshift-install create cluster --dir glnm2/
INFO Consuming OpenShift Install (Manifests) from target directory
INFO Consuming Master Machines from target directory
INFO Consuming Common Manifests from target directory

```

```

INFO Consuming Worker Machines from target directory
INFO Consuming Openshift Manifests from target directory
INFO Creating infrastructure resources...
INFO Waiting up to 20m0s (until 4:49AM EST) for the Kubernetes API at
https://api.glnm2.dynamic.opentlc.com:6443...
INFO API v1.28.6+6216ea1 up
INFO Waiting up to 1h0m0s (until 5:32AM EST) for bootstrapping to complete...
INFO Destroying the bootstrap resources...
INFO Waiting up to 40m0s (until 5:29AM EST) for the cluster at
https://api.glnm2.dynamic.opentlc.com:6443 to initialize...
INFO Waiting up to 30m0s (until 5:32AM EST) to ensure each cluster operator has finished
progressing...
INFO All cluster operators have completed progressing
INFO Checking to see if there is a route at openshift-console/console...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/lab-user/glnm2/auth/kubeconfig'
INFO Access the OpenShift web-console here:
https://console-openshift-console.apps.glnm2.dynamic.opentlc.com
INFO Login to the console with user: "kubeadmin", and password: "RLAiW-c83BZ-9SiAU-89pWa"
INFO Time elapsed: 38m14s

```

The resulting cluster has 3 worker and 2 infra nodes, but the infra services are running on the worker nodes because they don't have the matching tolerations to run on the infra nodes. This must be taken into account if the cluster does not have any worker nodes.

Moving the router pods to the infra nodes

Edit the default ingress controller and add a nodePlacement section that matches the infra label, and add the toleration for the infra taint

```

Unset
$ oc edit ingresscontroller default -n openshift-ingress-operator
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: default
  namespace: openshift-ingress-operator
spec:
  ...
  nodePlacement:
    nodeSelector:
      matchLabels:
        node-role.kubernetes.io/infra: ""
    tolerations:
      - effect: NoSchedule

```

```
key: node-role.kubernetes.io/infra
replicas: 2
tuningOptions:
  reloadInterval: 0s
```

After this, the router pods are redeployed to the infra nodes.

Moving the registry pods to the infra nodes

In the case of vsphere IPI installation, the registry is in a removed state because no object storage is readily available:

```
Unset
$ oc get config cluster -o yaml
apiVersion: imageregistry.operator.openshift.io/v1
kind: Config
metadata:
  creationTimestamp: "2024-03-02T09:49:10Z"
  finalizers:
  - imageregistry.operator.openshift.io/finalizer
  generation: 1
  name: cluster
  resourceVersion: "825739"
  uid: 53953d91-929f-40a6-93bb-f6ff25b0cb3f
spec:
  logLevel: Normal
  managementState: Removed
...
```

This has the effect of not having any registry pods running in the cluster

```
Unset
$ oc get pod -n openshift-image-registry -l docker-registry=default
No resources found in openshift-image-registry namespace.
```

So I would have to [assing storage to the registry](#) and the [add the nodePlacement section](#)

Moving the monitoring stack to the infra nodes

You need to create a configmap with the nodePlacement section and tolerations for all the monitoring components:

Unset

```
$ cat cluster-monitoring-config
apiVersion: v1
kind: ConfigMap
metadata:
  name: cluster-monitoring-config
  namespace: openshift-monitoring
data:
  config.yaml: |+
    alertmanagerMain:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
        - key: node-role.kubernetes.io/infra
          effect: NoSchedule
    prometheusK8s:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
        - key: node-role.kubernetes.io/infra
          effect: NoSchedule
    prometheusOperator:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
        - key: node-role.kubernetes.io/infra
          effect: NoSchedule
    k8sPrometheusAdapter:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
        - key: node-role.kubernetes.io/infra
          effect: NoSchedule
    kubeStateMetrics:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
        - key: node-role.kubernetes.io/infra
          effect: NoSchedule
    telemeterClient:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
        - key: node-role.kubernetes.io/infra
          effect: NoSchedule
    openshiftStateMetrics:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      tolerations:
```

```
- key: node-role.kubernetes.io/infra
  effect: NoSchedule
thanosQuerier:
  nodeSelector:
    node-role.kubernetes.io/infra: ""
  tolerations:
  - key: node-role.kubernetes.io/infra
    effect: NoSchedule
```

Apply the config map definition

```
Unset
$ oc create -f cluster-monitoring-config

$ watch -n4 "oc get pods -n openshift-monitoring -o wide"
```

Monitoring-plugin pods are still running on worker nodes

Other elements that need to be moved to infra nodes

Cronjobs for image-prunner and openshift-operator-lifecycle-manager may need to be modified so that the pods can run on infra nodes.

The network-check-source pod

```
Unset
$ oc get po -n openshift-network-diagnostics network-check-source-c9468c84c-4q9cn -o
yaml|less

$ oc get deployment -n openshift-network-diagnostics network-check-source -o yaml|less

$ oc get network cluster -o yaml|less
```

RHACM Operator Deployment

https://access.redhat.com/documentation/en-us/red_hat_advanced_cluster_management_for_kubernetes/2.9/html/install/installing#installing-from-the-cli

RHACM is installed on infra nodes and therefore the OCP web console cannot be used, so the installation instructions are based on the following chapters in the RHACM documentation:

- Installing from the OpenShift Container Platform CLI
- Installing the Red Hat Advanced Cluster Management hub cluster on infrastructure nodes

Create the recommended namespace to hold ACM components:

```
Unset
$ oc create namespace open-cluster-management
namespace/open-cluster-management created

$ oc project open-cluster-management
Now using project "open-cluster-management"
```

Create the operator group based on the following yaml file

```
Unset
$ cat og.yaml
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: rhacm-og
  namespace: open-cluster-management
spec:
  targetNamespaces:
    - open-cluster-management

$ oc create -f og.yaml
operatorgroup.operators.coreos.com/rhacm-og created
```

Create the operator subscription based on the following yaml file

There is no **tolerations** section because the infra nodes don't have a taint section, because the cluster has only infra nodes:

```
Unset
$ cat subs.yaml
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: acm-operator-subscription
spec:
  sourceNamespace: openshift-marketplace
  source: redhat-operators
  channel: release-2.9
  installPlanApproval: Automatic
  name: advanced-cluster-management
```

```

config:
  nodeSelector:
    node-role.kubernetes.io/infra: ""

$ oc apply -f subs.yaml
subscription.operators.coreos.com/acm-operator-subscription created

$ oc get pod
NAME                                READY   STATUS    RESTARTS   AGE
multiclustertest-operator-7bbd6b66bc-699hz  1/1     Running   0           32s

$ oc get csv
NAME                                ...   VERSION   ...   PHASE
advanced-cluster-management.v2.9.2  ...   2.9.2     ...   Succeeded

```

Create the multiclustertest custom resource based on the following yaml file:

```

Unset
$ cat multiclustertest.yaml
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
  name: multiclustertest
  namespace: open-cluster-management
spec:
  nodeSelector:
    node-role.kubernetes.io/infra: ""

```

If RHACM is being installed on a disconnected environment an special annotation needs to be added to the multiclustertest definition with the name of the catalogsource for the redat operators

```

Unset
$ oc get catalogsource -n openshift-marketplace
NAME                                DISPLAY TYPE   PUBLISHER   AGE
cs-redhat-operator-index            grpc          4h16m

$ cat multiclustertest.yaml
apiVersion: operator.open-cluster-management.io/v1
kind: MultiClusterHub
metadata:
  name: multiclustertest
  namespace: open-cluster-management
annotations:

```



```
installer.open-cluster-management.io/mce-subscription-spec: '{"source":  
"cs-redhat-operator-index"}'  
spec: {}
```

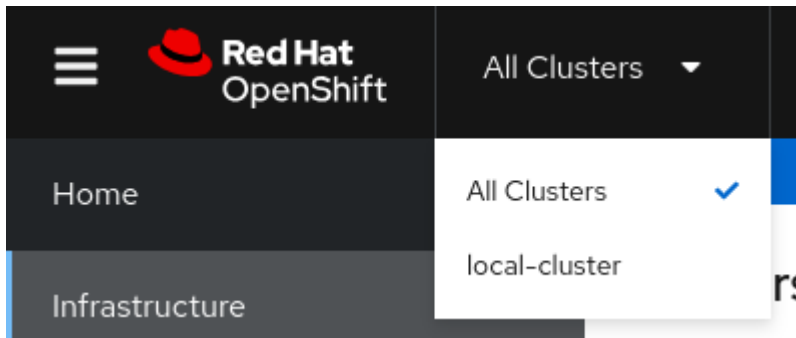
Unset

```
$ oc apply -f multiclusterhub.yaml  
multiclusterhub.operator.open-cluster-management.io/multiclusterhub created
```

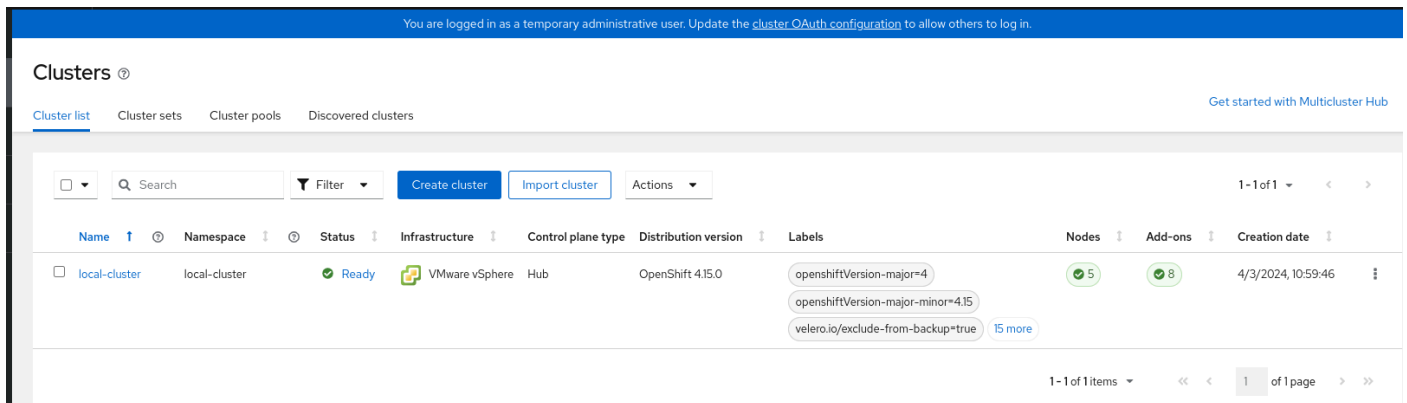
```
$ oc get pods -n open-cluster-management
```

| NAME | READY | STATUS | ... |
|--|-------|---------|-----|
| cluster-permission-8595fb4db-wrgjw | 1/1 | Running | |
| console-chart-console-v2-865d774dff-76jhg | 1/1 | Running | |
| console-chart-console-v2-865d774dff-9hnfh | 1/1 | Running | |
| grc-policy-addon-controller-7db8c447bd-5xs8p | 1/1 | Running | |
| grc-policy-addon-controller-7db8c447bd-fjtzs | 1/1 | Running | |
| grc-policy-propagator-6975744dbd-h8tfg | 2/2 | Running | |
| grc-policy-propagator-6975744dbd-tlnvs | 2/2 | Running | |
| insights-client-b9c7995b7-r2nnh | 1/1 | Running | |
| insights-metrics-5c947cd6d6-qnrwg | 2/2 | Running | |
| klusterlet-addon-controller-v2-589566fd7f-fl7dv | 1/1 | Running | |
| klusterlet-addon-controller-v2-589566fd7f-lfrf5 | 1/1 | Running | |
| multicluster-integrations-74d4b8d46f-42xxh | 3/3 | Running | |
| multicluster-observability-operator-659cf79d5f-zxqx4 | 1/1 | Running | |
| multicluster-operators-application-cfb6bb778-jbbnz | 3/3 | Running | |
| multicluster-operators-channel-c5c98b697-m8f46 | 1/1 | Running | |
| multicluster-operators-hub-subscription-6c565cb64b-j6gjn | 1/1 | Running | |
| multicluster-operators-standalone-subscription-69d95c6f5-npqnv | 1/1 | Running | |
| multicluster-operators-subscription-report-69d7675479-499g8 | 1/1 | Running | |
| multiclusterhub-operator-7bbd6b66bc-699hz | 1/1 | Running | |
| search-api-7bc68d4785-nbxwm | 1/1 | Running | |
| search-collector-6b499ffdbf-whxxp | 1/1 | Running | |
| search-indexer-dcf75d665-pnx2m | 1/1 | Running | |
| search-postgres-5d96ccd86b-qbnpn | 1/1 | Running | |
| search-v2-operator-controller-manager-5b6fdd4b6f-tnmgt | 2/2 | Running | |
| submariner-addon-98b6fff65-kk92v | 1/1 | Running | |
| volsync-addon-controller-8cfbd5d5f-h6dqn | 1/1 | Running | |

A new dropdown menu should appear in the Openshift web site.



The local cluster should go into status Ready after a couple minutes:



Mirror Registry

<https://github.com/quay/mirror-registry>

https://docs.openshift.com/container-platform/4.15/installing/disconnected_install/installing-mirroring-creating-registry.html#mirror-registry-localhost_installing-mirroring-creating-registry

[How to use the oc-mirror plug-in to mirror operators](#)

Setting up the environment

The mirror registry service needs to be resolvable by DNS, so we are going to deploy it remotely from one **VMware Cloud Public Cloud Open Environment** onto another so the bastion on the second environment can be used as the mirror registry for the first one, and the bastion public DNS name can be used to access the mirror registry.

Create a new **VMware Cloud Public Cloud Open Environment**

On the second VMware Cloud Public Cloud Open Environment

The bastion host does not have enough resources to run the mirror registry.

Shutdown the bastion host and add:

- One additional vCPU, 2 in total
- Increase the memory to 8GB
- Add a disk of 50GB.

Boot up the bastion

Show the available disks, a new **sdb** disk of 50G should appear

```
Unset
$ lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda   8:0    0 30G 0 disk
├─sda1 8:1    0    1M 0 part
├─sda2 8:2    0 100M 0 part /boot/efi
└─sda3 8:3    0 29.9G 0 part /
sdb   8:16   0 50G 0 disk
```

(Use LVM instead of the device directly, this way you can extend the partition if you run out of space)

```
Unset
$ sudo yum install lvm2
$ lsblk
$ sudo pvcreate /dev/sdb
$ sudo pvcreate /dev/sdd
$ sudo vgcreate viena /dev/sdb /dev/sdd
$ sudo vgs
$ sudo lvcreate -n lvmirror -l +100%FREE viena
$ sudo lvs
$ sudo mkfs.xfs /dev/viena/lvmirror
```

Format the new disk

```
Unset
$ sudo mkfs.xfs /dev/sdb
```

Create the mount point

```
Unset
$ sudo mkdir /var/mirror-registry
```

Get the UUID for the /dev/sdb disk

```
Unset
$ sudo blkid
...
```

```
/dev/sdb: UUID="d327e61d-8372-435..." BLOCK_SIZE="512" TYPE="xfs"
```

Add an entry like the following to the **/etc/fstab** file

```
Unset
UUID=d327e61d-8372-435... /var/mirror-registry xfs defaults 0 0
```

Mount the partition

```
Unset
$ sudo mount -a
$ df -Ph
```

Installing the mirror registry software

On the first VMware Cloud Public Cloud Open Environment

The mirror registry installer needs to be executed on the first environment, but it needs to run an ansible playbook against the second environment with root privileges so that the registry service can listen on privileged port 443.

Copy the existing ssh key for the root user in the second environment to the first environment

```
Unset
<second env>$ sudo cat /root/.ssh/bastion_995pv
-----BEGIN OPENSSSH PRIVATE KEY-----
b3B1bnNzaC1rZXktdjEAAAABG5vbmUAAAABbm9uZQAAAAAAAAABAAABlwAAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEA1QlqAIineSlrxIyEgOQ1A78uD36e1p32sTtpqCV/00ZUWng1LBhz
...
9yijev/D90ldJLAAAAEnJvb3RAYmFzdGlvbi050TVwdg==
-----END OPENSSSH PRIVATE KEY-----

<first env>$ vim ~/.ssh/mirror-root
<first env>$ chmod 0400 ~/.ssh/mirror-root
<first env>$ ssh -i ~/.ssh/mirror-root root@bastion-995pv.995pv.dynamic.opentlc.com
Last failed login: Fri Mar  8 04:31:40 EST 2024 from 79.138.85.216 on ssh:notty
There were 17 failed login attempts since the last successful login.
Last login: Thu Dec  7 05:08:51 2023
<first env># whoami
root
```

If a non privileged port like the default 843 is used, the ansible playbook can run as a normal user in the second environment and an ssh key for the lab user can be created and copied over

Unset

```
$ ssh-keygen -N '' -f ~/.ssh/mirror
$ ssh-copy-id -i ~/.ssh/mirror.pub lab-user@bastion-995pv.995pv.dynamic.opentlc.com

$ ssh -i ~/.ssh/mirror lab-user@bastion-995pv.995pv.dynamic.opentlc.com
```

Download the **mirror-registry.tar.gz** package for the latest version of the mirror registry found on the [OpenShift console Downloads](#) page.

Uncompress the tar file

Unset

```
$ mkdir mirror
$ mv mirror-registry.tar.gz mirror
$ cd mirror/
$ tar xvf mirror-registry.tar.gz
image-archive.tar
execution-environment.tar
mirror-registry
```

Run the installer on the first env to install the mirror registry remotely on the second env.

The resolvable hostname for the mirror registry service is the public hostname for the bastion host on the second environment and the port it uses is the standard for https 443

```
--quayHostname bastion-995pv.995pv.dynamic.opentlc.com:443
```

The partition where the images will be storage is the one associated with the disk added earlier

```
--quayStorage /var/mirror-registry
```

Because this is a remote installation from the first environment into the second environment, the connection parameters for the second environment are needed

```
--targetHostname bastion-995pv.995pv.dynamic.opentlc.com
--targetUsername lab-user
-k ~/.ssh/mirror-root
```

Unset

```
$ ./mirror-registry install --quayHostname bastion-995pv.995pv.dynamic.opentlc.com:443
--quayStorage /var/mirror-registry --targetHostname bastion-995pv.995pv.dynamic.opentlc.com
--targetUsername lab-user -k ~/.ssh/mirror-root
```

The final message contains the URL to access the registry and the user credentials.

Unset

```
....  
INFO[2024-03-07 02:22:58] Quay is available at https://bastion-sjmkz:8443 with credentials  
(init, j4HPx6Gu18052I70z1knDQw9do3thqKF)
```

Save the credentials for later use.

The installer creates systemd services for the different components:

Unset

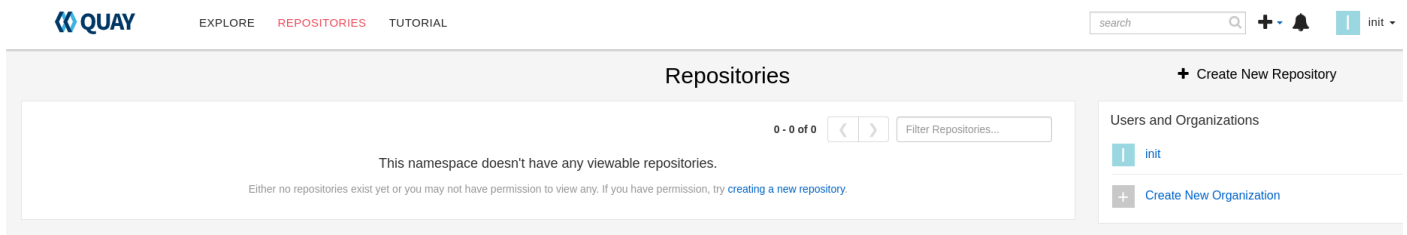
```
$ systemctl status quay-app.service  
$ systemctl status quay-postgres.service  
$ systemctl status quay-redis.service  
$ systemctl status quay-pod.service
```

Test the access to the registry. The option `--tls-verify=false` is used so that podman does not reject the unknown root CA in the mirror registry:

Unset

```
$ podman login --tls-verify=false -u init -p j4HPx... bastion-sjmkz.sjmkz.dynamic.opentlc.com  
Login Succeeded!
```

Access the web interface using the bastion external name (i.e. `bastion-sjmkz.sjmkz.dynamic.opentlc.com`)



Download the **mirror-plugin** and the oc CLI packages from the [OpenShift console Downloads](#) page. Install the oc CLI and the oc-mirror plugin

Unset

```
$ tar xvf oc-mirror.tar.gz  
oc-mirror  
$ chmod +x oc-mirror  
  
$ tar xvf openshift-client-linux.tar.gz  
README.md  
oc
```

```
kubect1
$ sudo cp oc oc-mirror /usr/local/bin/
```

Download your registry.redhat.io [pull secret from Red Hat OpenShift Cluster Manager](#).

Make a copy of your pull secret in JSON format:

```
Unset
$ cat pull-secret.txt | jq . > pull-secret_json.txt
```

Generate the base64-encoded username and password or token for your mirror registry:

```
Unset
$ echo -n 'init:j4HPx6Gu18052I70z1knDQw9do3thqKF' | base64 -w0
aW5pdDpqNEhQeDZHdTE4TzUySTcwemxrbkRRdz1kbzN0aHFLRg==
```

Edit the JSON file and add a section that describes your registry to it:
Make sure to add the port where the mirror registry is listening on

```
Unset
{
  "auths": {
    "cloud.openshift.com": {
      "auth": "b3B1bnNoaWZ0LXJlbGVhc2...
      "email": "jjerezro@redhat.com"
    },
    "quay.io": {
      "auth": "b3B1bnNoaWZ0LXJlbGVhc...
      "email": "jjerezro@redhat.com"
    },
    "registry.connect.redhat.com": {
      "auth": "NTIzMjU0MDB8dWhjLTFIaW...
      "email": "jjerezro@redhat.com"
    },
    "registry.redhat.io": {
      "auth": "NTIzMjU0MDB8dWhjLTFIaWRhWDBvbHZ0Wn1...
      "email": "jjerezro@redhat.com"
    },
    "bastion-sjmkz.sjmkz.dynamic.opentlc.com:443": {
      "auth": "aW5pdDpwa1pFUDhzNj12Q251SDcyVk9mMzFkeEpRTTBsWDRvNQ==",
```

```
      "email": "jjerezro@redhat.com"
    }
  }
}
```

Install the merged pull secret so that it can be used by the oc CLI.
Verify the pull secret and create the image set configuration file template.
The “oc mirror” command can take a couple minutes to complete

```
Unset
$ jq . pull-secret_json.txt

$ cp pull-secret_json.txt ~/.docker/config.json

$ oc mirror init --registry bastion-sjzmk.sjzmk.dynamic.opentlc.com/mirror/oc-mirror-metadata \
> imageset-config.yaml
```

Mirroring packages

Edit the imageset-config.yaml file and include the required content to be mirrored to the local registry.
Examples and further details can be found [here](#)

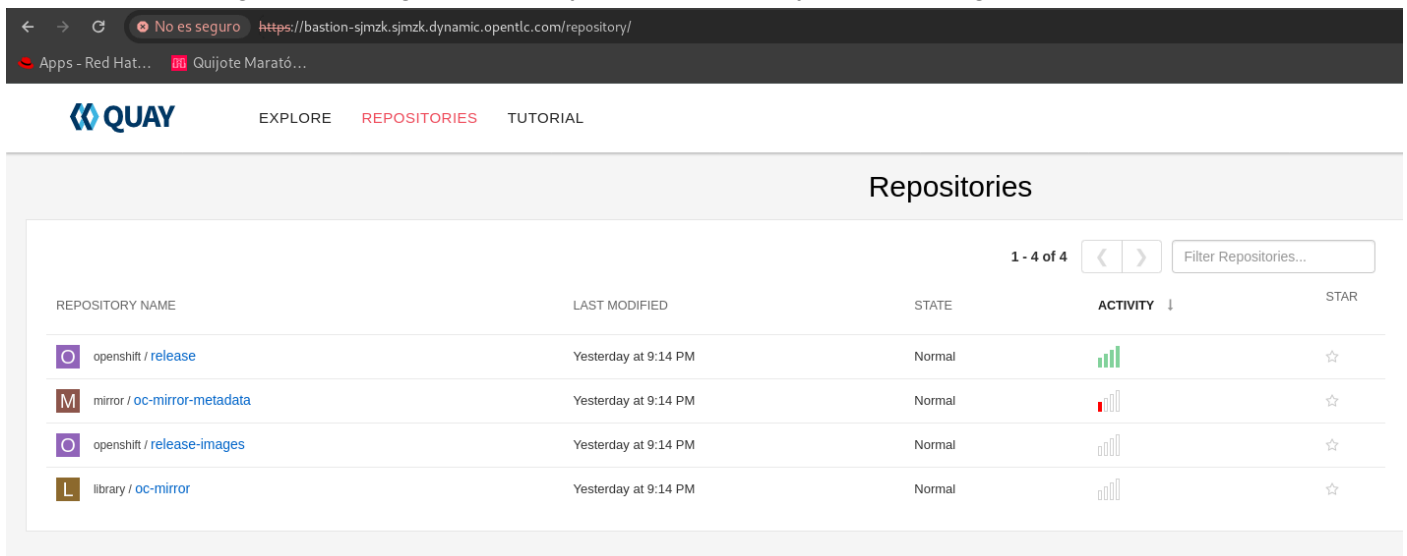
To mirror a particular version of Openshift use a configuration like. This configuration file only mirrors one particular OCP version and does not include any operators:

```
Unset
kind: ImageSetConfiguration
apiVersion: mirror.openshift.io/v1alpha2
storageConfig:
  registry:
    imageURL: bastion-995pv.995pv.dynamic.opentlc.com:443/mirror/oc-mirror-metadata
    skipTLS: true
mirror:
  platform:
    architectures:
      - amd64
    channels:
      - name: stable-4.15
        minVersion: 4.15.0
        maxVersion: 4.15.0
        type: ocp
```


If the ImageSetConfiguration contains an operators section you need to apply the catalog source obtained after the content is mirrored into the registry, see [here](#)
Execute the images mirroring:

```
Unset
$ oc mirror --config=./imageset-config.yaml \
  docker://bastion-995pv.995pv.dynamic.opentlc.com:443 --dest-skip-tls
Checking push permissions for bastion-sjzmk.localdomain:443
Found: oc-mirror-workspace/src/publish
Found: oc-mirror-workspace/src/v2
Found: oc-mirror-workspace/src/charts
Found: oc-mirror-workspace/src/release-signatures
No metadata detected, creating new workspace
...
info: Mirroring completed in 7m43.76s (42.41MB/s)
Writing image mapping to oc-mirror-workspace/results-1709842466/mapping.txt
Writing ICSP manifests to oc-mirror-workspace/results-1709842466
```

When the mirroring is finished go to the quay web site to verify that the images are there



The screenshot shows the Quay.io web interface. The browser address bar displays `https://bastion-sjzmk.sjzmk.dynamic.opentlc.com/repository/`. The Quay logo is visible in the top left, with navigation links for EXPLORE, REPOSITORIES, and TUTORIAL. The main heading is "Repositories". Below this is a table with the following columns: REPOSITORY NAME, LAST MODIFIED, STATE, ACTIVITY, and STAR. There are 4 repositories listed, all with a "Normal" state and "Yesterday at 9:14 PM" last modified time. The repositories are: openshift / release, mirror / oc-mirror-metadata, openshift / release-images, and library / oc-mirror. Each repository has an activity bar chart and a star icon.

| REPOSITORY NAME | LAST MODIFIED | STATE | ACTIVITY | STAR |
|---|----------------------|--------|----------|------|
| openshift / release | Yesterday at 9:14 PM | Normal | | |
| mirror / oc-mirror-metadata | Yesterday at 9:14 PM | Normal | | |
| openshift / release-images | Yesterday at 9:14 PM | Normal | | |
| library / oc-mirror | Yesterday at 9:14 PM | Normal | | |

On the second VMware Cloud Public Cloud Open Environment

Download the RHCOS image to install OpenShift on a restricted network vSphere environment. The installer needs to load the installation ova image into vSphere, so we need to download the image “Red Hat Enterprise Linux CoreOS - vSphere” from the [Product Downloads page](#). This ova image is more than 1GB in size, keep that in mind.

Copy the image to the bastion host in the second environment:

```
Unset
$ scp /home/jjerezro/Descargas/rhcos-vmware.x86_64.ova \
```

```
lab-user@bastion-995pv.995pv.dynamic.opentlc.com:
```

On the bastion in the second environment verify the RHCOS image and move to a directory

```
Unset
$ mkdir images
$ mv rhcos-vmware.x86_64.ova images

$ sha256sum images/rhcos-vmware.x86_64.ova
9b3d5a598928ec52b0d32092d0a9a41f0ec8a238eb9fff8563266b9351919e20 ...
```

The image must be published in a web server, in this case an apache container image is used for such purpose since the server only has to be available during installation of the cluster.

The container has to be run as root because the pod binds to port 80, because other ports are not accessible from other environments.

```
Unset
$ sudo podman run -p 80:8080 --name httpd -d \
  --volume ~/images:/var/www/html:Z registry.access.redhat.com/ubi8/httpd-24
```

Test the access to the web site from the first environment:

```
Unset
$ wget http://bastion-995pv.995pv.dynamic.opentlc.com/rhcos-vmware.x86_64.ova
--2024-03-08 09:57:29--
http://bastion-995pv.995pv.dynamic.opentlc.com/rhcos-vmware.x86_64.ova
Resolving bastion-995pv.995pv.dynamic.opentlc.com
(bastion-995pv.995pv.dynamic.opentlc.com)... 52.23.56.147
Connecting to bastion-995pv.995pv.dynamic.opentlc.com
(bastion-995pv.995pv.dynamic.opentlc.com)|52.23.56.147|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1261096960 (1.2G) [application/x-tar]
Saving to: 'rhcos-vmware.x86_64.ova'

rhcos-vmware.x86_64.ova      23%[=====>                ] 280.19M   156MB/s
```

The **install-config.yaml** file needs to include additional configurations for the disconnected installation:

- The **clusterOSImage** with the URL to get the ova image, and the sha256 sum obtained from the downloads page or computed using the sha256sum command

- The pull secret for the mirror registry
Use the same definition that you used in the json pull-secret for the mirror registry, including the port number. If they differ the nodes will not be able to pull the images from the registry.
- The CA certificate for the mirror registry
This is obtained from the file **quay-install/quay-rootCA/rootCA.pem** in the home of the user who installed the mirror registry in the target host. In this case because the installer was run with sudo, the file is in the root's home directory of the 2nd environment.
It is important that the certificate itself is properly indented
- The image content resources obtained from the file
oc-mirror-workspace/results-1709896183/imageContentSourcePolicy.yaml
This file is found in the host where the installer was run (bastion in 1st environment)

The resulting file looks like this:

```
Unset
additionalTrustBundlePolicy: Proxyonly
apiVersion: v1
baseDomain: dynamic.opentlc.com
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 3
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform: {}
  replicas: 3
metadata:
  creationTimestamp: null
  name: sjmzk
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes
  serviceNetwork:
  - 172.30.0.0/16
platform:
  vsphere:
    apiVIPs:
    - 192.168.95.201
    failureDomains:
    - name: generated-failure-domain
```

```

region: generated-region
server: vcenter.sddc-44-197-86-61.vmwarevmc.com
topology:
  computeCluster: /SDDC-Datacenter/host/Cluster-1
  datacenter: SDDC-Datacenter
  datastore: /SDDC-Datacenter/datastore/WorkloadDatastore
  networks:
    - segment-sandbox-sjmkz
  resourcePool: /SDDC-Datacenter/host/Cluster-1/Resources
  folder: /SDDC-Datacenter/vm/Workloads/sandbox-sjmkz
  zone: generated-zone
ingressVIPs:
- 192.168.95.202
vcenters:
- datacenters:
  - SDDC-Datacenter
  password: t4uy8zoGPJQ7
  port: 443
  server: vcenter.sddc-44-197-86-61.vmwarevmc.com
  user: sandbox-sjmkz@vc.opentlc.com
clusterOSImage: http://bastion-995pv.995pv.dynamic.opentlc.com/rhcos-vmware.x86_64.ova?sha256=9b3d...
imageContentSources:
- mirrors:
  - bastion-995pv.995pv.dynamic.opentlc.com:443/openshift/release-images
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - bastion-995pv.995pv.dynamic.opentlc.com:443/openshift/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
pullSecret: '{"auths":{"bastion-995pv.995pv.dynamic.opentlc.com:443":{"auth":"aW5pd..."}}}'
sshKey: |
  ssh-rsa AAAAB3NzaC1y...
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  MIID0jCCArqgAwIBAgIULG+VTl7hJhwc+1H9hHg0cAQAWQEwDQYJKoZIhvcNAQEL
  ...
  X2d/YiJ23sGBtDxKuQpdTSS1iG9wag==
  -----END CERTIFICATE-----

```

Add the vCenter root CA certificates as described [above](#)
 Run the installer

```

Unset
$ mkdir sjmkz
$ cp install-config.yaml sjmkz/
$ openshift-install create cluster --dir sjmkz/

```

Operator Catalog Content

The Operator Lifecycle Manager (OLM) applies the original catalogSources to the cluster, but the pods associated with them don't exist in the mirror registry and therefore cannot be started.

Listing the pods in the openshift-marketplace project shows that some pods are not starting.

Unset

```
$ oc get pods -n openshift-marketplace
```

| NAME | READY | STATUS | RESTARTS | AGE |
|---------------------------------------|------------|------------------------------|-------------|------------|
| certified-operators-9rtn5 | 0/1 | Init:ImagePullBackOff | 0 | 21h |
| certified-operators-wshfs | 0/1 | Init:ImagePullBackOff | 0 | 21h |
| community-operators-5h5hc | 0/1 | Init:ImagePullBackOff | 0 | 21h |
| community-operators-wpb48 | 0/1 | Init:ImagePullBackOff | 0 | 21h |
| cs-redhat-operator-index-4jdk6 | 1/1 | Running | 0 | 4h7m |
| marketplace-operator-5fc5bc69d8-gd7zr | 1/1 | Running | 5 (21h ago) | 21h |
| redhat-marketplace-d2tm8 | 0/1 | Init:ImagePullBackOff | 0 | 21h |
| redhat-marketplace-srs5v | 0/1 | Init:ImagePullBackOff | 0 | 21h |
| redhat-operators-m679s | 0/1 | Init:ImagePullBackOff | 0 | 21h |
| redhat-operators-pzlkf | 0/1 | Init:ImagePullBackOff | 0 | 21h |

For some reason this situation blocks the installation of any operator that has been loaded to the mirror registry.

The logs of the catalog operator show constant connection errors

Unset

```
I0309 17:41:44.746653      1 event.go:298] Event(v1.ObjectReference{Kind:"Namespace",
Namespace:"", Name:"open-cluster-management", UID:"21e8b0a3-78ff-4374-90bf-820ad49ce97b",
APIVersion:"v1", ResourceVersion:"491216", FieldPath:""}): type: 'Warning' reason:
'ResolutionFailed' [failed to populate resolver cache from source
redhat-operators/openshift-marketplace: failed to list bundles: rpc error: code = Unavailable
desc = connection error: desc = "transport: Error while dialing dial tcp 172.30.192.30:50051:
connect: connection refused", failed to populate resolver cache from source
community-operators/openshift-marketplace: failed to list bundles: rpc error: code =
Unavailable desc = connection error: desc = "transport: Error while dialing dial tcp
172.30.238.156:50051: connect: connection refused", failed to populate resolver cache from
source certified-operators/openshift-marketplace: failed to list bundles: rpc error: code =
Unavailable desc = connection error: desc = "transport: Error while dialing dial tcp
172.30.65.26:50051: connect: connection refused", failed to populate resolver cache from source
redhat-marketplace/openshift-marketplace: failed to list bundles: rpc error: code = Unavailable
desc = connection error: desc = "transport: Error while dialing dial tcp 172.30.142.8:50051:
connect: connection refused"]]
```

To fix this situation, Disable the default OperatorHub sources.

Unset

```
$ oc patch OperatorHub cluster --type json \
  -p ' [{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}] '
```

```
$ oc get operatorhub cluster -o yaml
apiVersion: config.openshift.io/v1
kind: OperatorHub
metadata:
  annotations:
    capability.openshift.io/name: marketplace
    include.release.openshift.io/ibm-cloud-managed: "true"
    include.release.openshift.io/self-managed-high-availability: "true"
    include.release.openshift.io/single-node-developer: "true"
    release.openshift.io/create-only: "true"
  name: cluster
spec:
  disableAllDefaultSources: true
status:
  sources:
    - disabled: true
      name: redhat-operators
      status: Success
    - disabled: true
      name: certified-operators
      status: Success
    - disabled: true
      name: community-operators
      status: Success
    - disabled: true
      name: redhat-marketplace
      status: Success
```

To include operator images in the mirror registry we need to collect some additional information. All these commands take up to a couple of minutes to execute. Faster alternatives may exist using **opm** or **grpcurl**

Get the catalogs

Unset

```
$ oc-mirror list operators --catalogs --version=4.15 > catalogs_4.15
```

```
$ cat catalogs_4.15
Available OpenShift OperatorHub catalogs:
OpenShift 4.15:
registry.redhat.io/redhat/redhat-operator-index:v4.15
```

```
registry.redhat.io/redhat/certified-operator-index:v4.15
registry.redhat.io/redhat/community-operator-index:v4.15
registry.redhat.io/redhat/redhat-marketplace-index:v4.15
```

Find the available packages in each catalog

```
Unset
$ oc-mirror list operators \
  --catalog=registry.redhat.io/redhat/redhat-operator-index:v4.15 \
  > redhat_package_4.15

$ oc-mirror list operators \
  --catalog=registry.redhat.io/redhat/certified-operator-index:v4.15 \
  > certified_package_4.15

$ oc-mirror list operators \
  --catalog=registry.redhat.io/redhat/community-operator-index:v4.15 \
  > community_package_4.15

$ oc-mirror list operators \
  --catalog=registry.redhat.io/redhat/redhat-marketplace-index:v4.15 \
  > marketplace_package_4.15
```

Find the available channels for the selected packages. This command is very slow, if you are requesting information about several operators, it can take a long time to complete.

```
Unset
$ oc-mirror list operators \
  --catalog=registry.redhat.io/redhat/redhat-operator-index:v4.15 \
  --package=advanced-cluster-management
```

| NAME | DISPLAY NAME | DEFAULT CHANNEL |
|-----------------------------|--|-----------------|
| advanced-cluster-management | Advanced Cluster Management for Kubernetes | release-2.9 |

| PACKAGE | CHANNEL | HEAD |
|-----------------------------|-------------|------------------------------------|
| advanced-cluster-management | release-2.9 | advanced-cluster-management.v2.9.2 |

Find the package versions within the selected channel

```
Unset
$ oc-mirror list operators \
  --catalog=registry.redhat.io/redhat/redhat-operator-index:v4.15 \
  --package=advanced-cluster-management --channel=release-2.9
```

VERSIONS

2.9.0

2.9.1

2.9.2

With this information add a new section for the operators to the **ImageSetConfiguration** file

```
Unset
kind: ImageSetConfiguration
apiVersion: mirror.openshift.io/v1alpha2
storageConfig:
  registry:
    imageURL: bastion-995pv.995pv.dynamic.opentlc.com:443/mirror/oc-mirror-metadata
    skipTLS: true
mirror:
  platform:
    architectures:
      - "amd64"
      - "multi"
    channels:
      - name: stable-4.15
        minVersion: 4.15.0
        maxVersion: 4.15.0
        type: ocp
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.15
      packages:
        - name: advanced-cluster-management
          channels:
            - name: release-2.9
              minVersion: '2.9.1'
              maxVersion: '2.9.2'
        - name: multicloud-engine
          channels: stable-2.4
        - name:
          minVersion: '2.4.2'
          maxVersion: '2.4.3'
```

Mirror the content, in this example the packages are downloaded to the local filesystem (/var/mirror-registry/operator_catalog) instead of uploading them directly to the mirror registry. The directory is created if it does not exist.

This command takes quite some time to complete, to the point where it looks like it is hung.

Unset

```
$ oc mirror --config=./imageset-config.yaml \
  file:///var/mirror-registry/operator_catalog --dest-skip-tls
Creating directory: /var/mirror-registry/operator_catalog/oc-mirror-workspace/src/publish
Creating directory: /var/mirror-registry/operator_catalog/oc-mirror-workspace/src/v2
Creating directory: /var/mirror-registry/operator_catalog/oc-mirror-workspace/src/charts
Creating directory: /var/mirror-registry/operator_catalog/oc-mirror-workspace/src/release-signatures

wrote mirroring manifests to
/var/mirror-registry/operator_catalog/oc-mirror-workspace/operators.1709985535/manifests-r
edhat-operator-index
...
info: Mirroring completed in 1m47.59s (133MB/s)
Creating archive /var/mirror-registry/operator_catalog/mirror_seq2_000000.tar
```

One reason it takes so long is because it compares the contents of the mirror registry with the packages to be downloaded so that it only downloads the packages that are not already in the mirror registry. The command also removes any packages from the mirror registry that are not referenced in the imagesetconfiguration.

Running the same command a second time renders the following result:

Unset

```
$ oc mirror --config=./2imageset-config.yaml file:///var/mirror-registry/operator_catalog
--dest-skip-tls
Creating directory: /var/mirror-registry/operator_catalog/oc-mirror-workspace/src/publish
Creating directory: /var/mirror-registry/operator_catalog/oc-mirror-workspace/src/v2
Creating directory: /var/mirror-registry/operator_catalog/oc-mirror-workspace/src/charts
Creating directory:
/var/mirror-registry/operator_catalog/oc-mirror-workspace/src/release-signatures
wrote mirroring manifests to
/var/mirror-registry/operator_catalog/oc-mirror-workspace/operators.1709994990/manifests-r
edhat-operator-index

To upload local images to a registry, run:

    oc adm catalog mirror file://redhat/redhat-operator-index:v4.15 REGISTRY/REPOSITORY
No new images detected, process stopping
```

The file needs to be uploaded to the mirror registry from the first bastion where the public DNS name can be used to access the mirror registry, so the tar file will not be transferred to the second bastion and will not be loaded from there as would be the case in another less stringent situation.

Load the file from the first bastion into the second bastion

Unset

```
$ oc mirror --from=/var/mirror-registry/operator_catalog/mirror_seq2_000000.tar \
  docker://bastion-995pv.995pv.dynamic.opentlc.com:443 --dest-skip-tls
Checking push permissions for bastion-995pv.995pv.dynamic.opentlc.com:443

Publishing image set from archive
"/var/mirror-registry/operator_catalog/mirror_seq2_000000.tar" to registry
"bastion-995pv.995pv.dynamic.opentlc.com:443"
...
Rendering catalog image
"bastion-995pv.995pv.dynamic.opentlc.com:443/redhat/redhat-operator-index:v4.15" with
file-based catalog
Writing image mapping to oc-mirror-workspace/results-1709989778/mapping.txt
Writing CatalogSource manifests to oc-mirror-workspace/results-1709989778
Writing ICSP manifests to oc-mirror-workspace/results-1709989778
```

Link the new content uploaded to the mirror registry with the OCP cluster by applying the `imageContentSourcePolicy` and `catalogSource` files generated by the `oc mirror` command in the first bastion host

Unset

```
$ ls -l oc-mirror-workspace/results-1709989778/
catalogSource-cs-redhat-operator-index.yaml
charts
imageContentSourcePolicy.yaml
mapping.txt
release-signatures

$ oc apply -f oc-mirror-workspace/results-1709989778/imageContentSourcePolicy.yaml
imagecontentsourcepolicy.operator.openshift.io/operator-0 created

$ oc apply -f
oc-mirror-workspace/results-1709989778/catalogSource-cs-redhat-operator-index.yaml
catalogsource.operators.coreos.com/cs-redhat-operator-index created
```

Verify that the cluster sees the new operator. It takes a few minutes for the catalog operator to process the information and make the packagemanifests and the operators available.

Unset

```
$ oc get catalogsource -n openshift-marketplace cs-redhat-operator-index
NAME                                DISPLAY    TYPE    PUBLISHER    AGE
cs-redhat-operator-index           grpc                               7m4s

$ oc logs -f catalog-operator-674487f7b5-w4bmb -n openshift-operator-lifecycle-manager
```

```
$ oc get packagemanifests -n openshift-marketplace
NAME                                CATALOG    AGE
advanced-cluster-management        7m29s
```

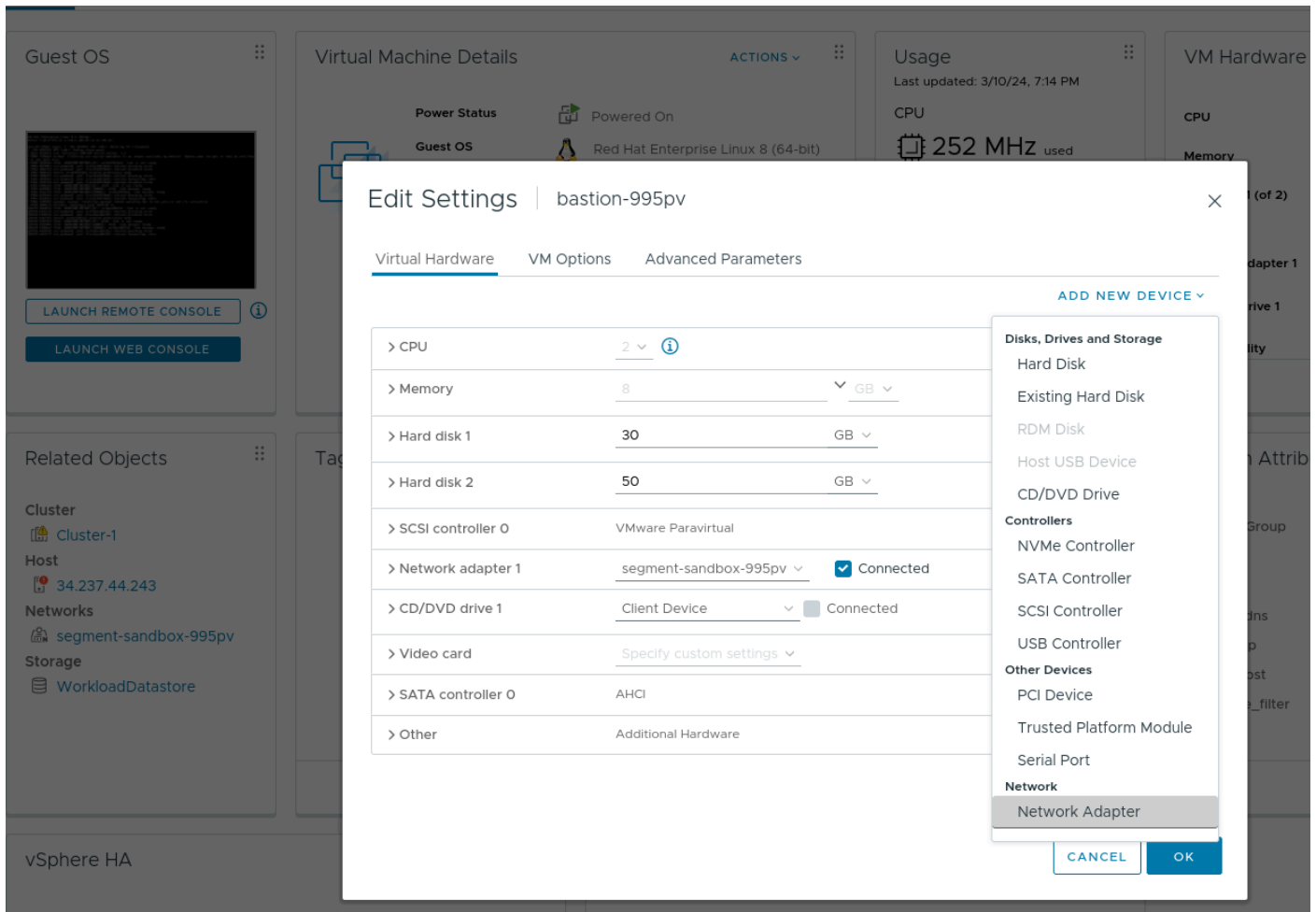
Managed cluster environment configuration

This section applies mostly to the additional **VMware Cloud Public Cloud Open Environments** where the managed OCP clusters are being installed.

You cannot use the same **VMware Cloud Public Cloud Open Environments** for the mirror registry and for the managed cluster because the mirror registry is already using port 443, and probably there is an http server to publish the `clusterOSImage` using port 80. In this case a 3rd **VMware Cloud Public Cloud Open Environments**

The goal here is to assign the IP addresses for api and *.apps DNS records to the bastion host, and install haproxy in the bastion to redirect requests to the OCP 4 managed cluster.

On the secondary **VMware Cloud Public Cloud Open Environment** add a new network interface to the bastion host.



Configure the network interface to have the IP addresses assigned to the api and *.apps DNS records

API DNS api.rscp4.dynamic.opentlc.com points to NAT IP to 192.168.188.201

Wildcard DNS *.apps.rscp4.dynamic.opentlc.com points to NAT IP to 192.168.188.202

The new NIC is eth1

Unset

```
$ nmcli con show
```

```
NAME                                UUID                                TYPE    DEVICE
System eth0                        5fb06bd0-0bb0-7ffb-45f1-d6edd65f3e03 ethernet eth0
cni-podman0                        9529674c-0f4d-4738-8736-1e589df0afc4 bridge  cni-podman0
Wired connection 1                 1c6cb2f2-78a7-3a1a-a7ef-4d18be5d80cd ethernet eth1
```

```
$ sudo nmcli con down "Wired connection 1"
```

```
$ sudo nmcli con mod "Wired connection 1" ipv4.method manual ipv4.addr "192.168.188.201, 192.168.188.202"
```

```
$ sudo nmcli con up "Wired connection 1"
```

```
$ ip -4 a
```

```
...
```

```
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default
qlen 1000
    inet 192.168.188.201/32 scope global noprefixroute eth1
    valid_lft forever preferred_lft forever
    inet 192.168.188.202/32 scope global noprefixroute eth1
```

Clone the following git repository:

```
Unset
$ git clone https://github.com/tale-toul/OCPonpremviaACM.git
$ $ cd OCPonpremviaACM/
```

Install ansible:

```
Unset
$ sudo python3 -m pip install ansible
```

Update the vars file. Add some properties to ansible/vars.yml. We're interested in this section:

```
Unset
##### YOUR PROPERTIES HERE #####
api_ip: 192.168.188.201
apps_ip: 192.168.188.202
guid: 'rscp4'
ocp_version: 4.14.13
#####
```

The ansible playbook is simplified so that only the tasks related to installing and enabling haproxy, the bind DNS server, and the task that downloads the oc client are executed.

Run the playbook

```
Unset
$ ansible-playbook -vvv ansible/main.yml
```

Verify that haproxy is listening on the expected ports

Unset

```
$ sudo ss -tlnp
```

| State | Local Address:Port | Peer Address:Port |
|--------|-----------------------|-------------------|
| ... | | |
| LISTEN | 192.168.188.202:443 | 0.0.0.0:* |
| LISTEN | 192.168.188.201:22623 | 0.0.0.0:* |
| LISTEN | 192.168.188.201:6443 | 0.0.0.0:* |
| LISTEN | 192.168.188.202:80 | 0.0.0.0:* |

Create a Host inventory

https://access.redhat.com/documentation/en-us/red_hat_advanced_cluster_management_for_kubernetes/2.9/html/clusters/cluster_mce_overview#create-host-inventory-console-steps

Before creating the first host inventory, the host inventory settings must be set.

From the All cluster web console -> Host inventory -> Configure host inventory settings

This creates the following pods in the project **multiclustere-engine**

The screenshot shows the Red Hat OpenShift web console interface. On the left is a navigation menu with options like Home, Infrastructure, Clusters, Automation, Host inventory, Applications, Governance, and Credentials. The main area displays the 'Host inventory' section, indicating '0 hosts in 0 infrastructure environments'. A modal dialog titled 'Configure host inventory settings' is open. It contains a message about enabling Central Infrastructure Management and a link to documentation. Below this are 'Storage sizes' fields: 'Database storage' (10 Gi), 'System storage' (100 Gi), and 'Image storage' (50 Gi). There is a checkbox for 'Configure load balancer on Amazon Web Services for me.' and a warning message: 'Storage sizes cannot be changed once you configure.' At the bottom of the dialog are 'Configure' and 'Cancel' buttons.

Unset

```
$ oc get pods -n multiclustere-engine
```

| | | | | |
|--|-----|---------|---|-------|
| agentinstalladmission-6bdf8c65d4-8pszw | 1/1 | Running | 0 | 8m15s |
|--|-----|---------|---|-------|

```

agentinstalladmission-6bdf8c65d4-ltjnp      1/1      Running 0      8m15s
assisted-image-service-0                    1/1      Running 0      8m15s
assisted-service-5c84fc8bf9-rj5dj           2/2      Running 0      8m15s

```

And the following PVCs

```

Unset
$ oc get pvc -n multicloud-engine
NAME                                STATUS  VOLUME                                CAPACITY  ACCESS MODES
STORAGECLASS  AGE
assisted-service      Bound  pvc-c3802bb2-bb29-4d34-b90d-6e318cf6a660  50Gi      RWO
thin-csi            9m53s
image-service-data-assisted-image-service-0 Bound
pvc-dd039e55-d750-4d62-936a-b7447e26750e 30Gi      RWO      thin-csi      9m52s
postgres      Bound  pvc-5de1ba28-41e1-4c7d-bad1-9d55cd2a5f9a 10Gi      RWO
thin-csi            9m53s

```

I am not creating the baremetal host and provisioning resources as requested in the [documentation](#) because I am not using baremetal hosts.

For a disconnected environment create the following config map, in a connected environment skip the creation of this configmap

The ca-bundle.crt content is the same as was used in the **additionalTrustBundle** section of the install-config.yaml and contains the CA root for the mirror registry.

The registry configuration should contain entries for any repositories that are being accessed on the mirror registry. The official documentation is not clear on this part, it looks like only [registry.redhat.io/multicloud-engine](#) is needed, but that is not enough.

The entries in registries.conf will end up in the imagecontentsourcepolicy resource in the installed cluster. The format for the registries.conf is different from that of the install-config.yaml and the imagecontentsourcepolicy, more info can be found [here](#)

```

Unset
$ cat mirror-config

apiVersion: v1
kind: ConfigMap
metadata:
  name: mirror-config
  namespace: multicloud-engine
  labels:
    app: assisted-service
data:
  ca-bundle.crt: |

```

```

-----BEGIN CERTIFICATE-----
MIID/jCCAuagAwIBAgIUcTPgO9RSv9q5Ys76wxsvZjS3PdwDQYJKoZIhvcNAQEL
...
2VAODDFugUjzzmTV+UTfM5mu
-----END CERTIFICATE-----
registries.conf: |
unqualified-search-registries = ["registry.access.redhat.com", "docker.io"]
[[registry]]
    prefix = ""
    location = "registry.redhat.io/multicloud-engine"
    mirror-by-digest-only = true
[[registry.mirror]]
    location = "bastion-995pv.995pv.dynamic.opentlc.com:443/multicloud-engine"
[[registry]]
    prefix = ""
    location = "quay.io/openshift-release-dev/ocp-release"
    mirror-by-digest-only = true
[[registry.mirror]]
    location = "bastion-995pv.995pv.dynamic.opentlc.com:443/openshift-release-images"
[[registry]]
    prefix = ""
    location = "quay.io/openshift-release-dev/ocp-v4.0-art-dev"
    mirror-by-digest-only = true
[[registry.mirror]]
    location = "bastion-995pv.995pv.dynamic.opentlc.com:443/openshift-release"
[[registry]]
    prefix = ""
    location = "registry.redhat.io/rhacm2"
    mirror-by-digest-only = true
[[registry.mirror]]
    location = "bastion-995pv.995pv.dynamic.opentlc.com:443/rhacm2"

$ oc apply -f mirror-config

```

If you modify the mirror-config configmap at some point you need to restart the assisted service pod

For a disconnected environment create the `AgentServiceConfig` custom resource

The `mirrorRegistryRef` points to the config map created above

The `osImages` section defines a collection of Operating System images (ie. RHCOS images) that the assisted-service should use as the base when generating discovery ISOs.

- The `url` field must point to a web server holding the live iso and live-rootfs images. The `rootFSUrl` field is deprecated
The images can be obtained from [here](#).
The same pod based http web server that was used to hold the “ova” image can be used for this purpose.
Make sure that the file has a SELinux label fo `container_file_t`

Unset

```
$ sudo semanage fcontext -a -t container_file_t /home/lab-user/images/

$ sudo restorecon -Rv /home/lab-user/images/
...
Relabeled /home/lab-user/images/rhcos-4.15.0-x86_64-live-rootfs.x86_64.img from
unconfined_u:object_r:user_home_t:s0 to unconfined_u:object_r:container_file_t:s0
```

- The version field is obtained from the live iso image itself, use the following command

Unset

```
$ file rhcos-4.15.0-x86_64-live.x86_64.iso
rhcos-4.15.0-x86_64-live.x86_64.iso: ISO 9660 CD-ROM filesystem data (DOS/MBR boot sector)
'rhcos-415.92.202402130021-0' (bootable)
```

Unset

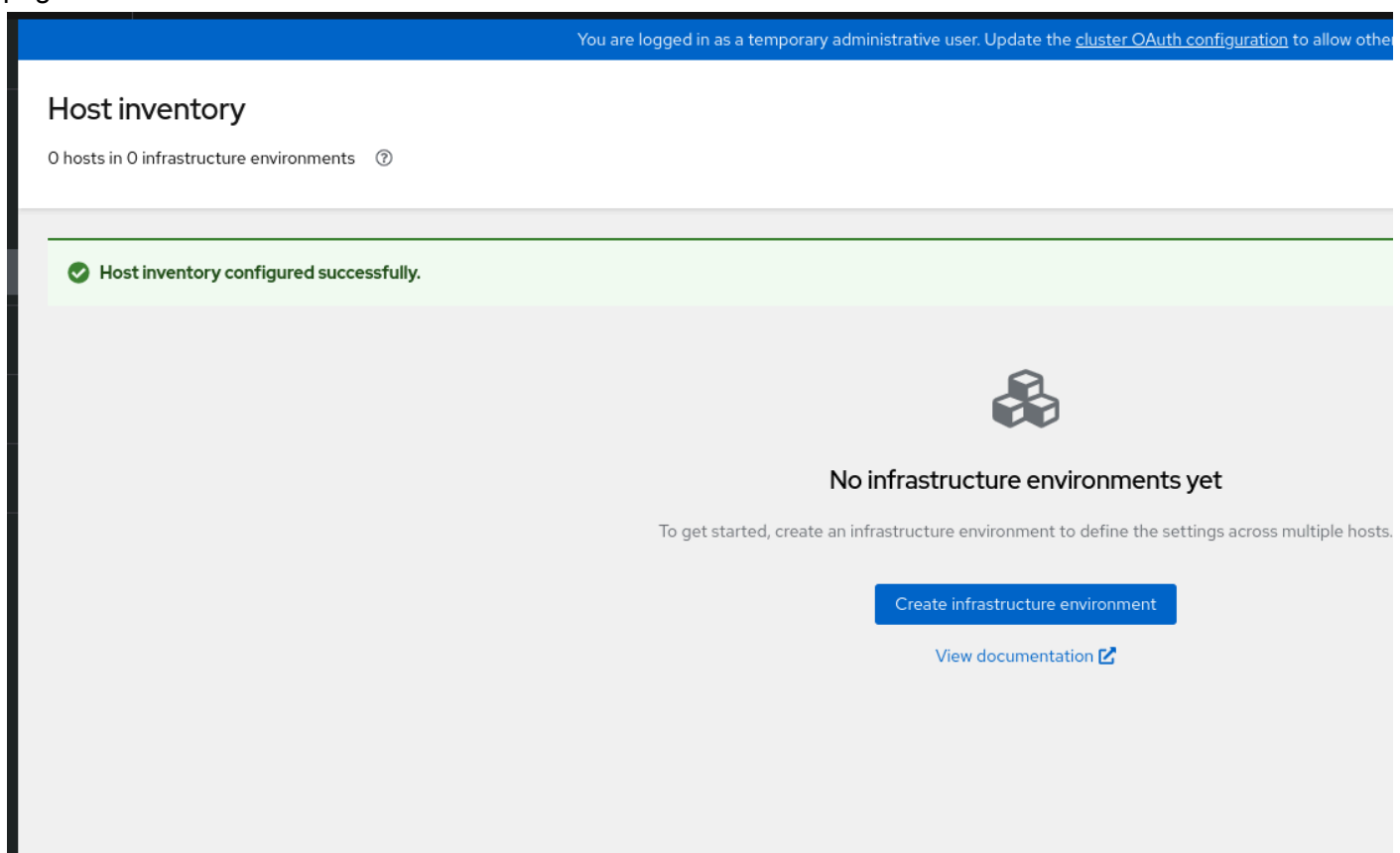
```
$ cat agent-service.yaml

apiVersion: agent-install.openshift.io/v1beta1
kind: AgentServiceConfig
metadata:
  name: agent
spec:
  databaseStorage:
    accessModes:
      - ReadWriteOnce
    resources:
      requests:
        storage: 10Gi
  filesystemStorage:
    accessModes:
      - ReadWriteOnce
    resources:
      requests:
        storage: 100Gi
  imageStorage:
    accessModes:
      - ReadWriteOnce
    resources:
      requests:
        storage: 50Gi
  mirrorRegistryRef:
    name: mirror-config
```

```
osImages:  
- openshiftVersion: "4.15"  
  version: "415.92.202402130021-0"  
  url: "http://bastion-995pv.995pv.dynamic.opentlc.com/rhcos-4.15.0-x86_64-live.x86_64.iso"  
  cpuArchitecture: "x86_64"
```

```
$ oc apply -f agent-service.yaml
```

Create the infrastructure environment by clicking on the blue button on the middle of the Host inventory page.



In a disconnected cluster the pull secret is for the mirror registry, but it must contain at least an entry for registry.redhat.io or the web interface complains about this.

Create infrastructure environment

☒ YAML: Off

Infrastructure environments are used by clusters. Create an infrastructure environment to add resources to your cluster.

Name *

clue 

Network type

☐ DHCP only ☒ Static IP, bridges and bonds 

CPU architecture

☒ x86_64 ☐ arm64

Location *

VMWareCloudPublicOpenEnv-rscp4 

Used to describe hosts' physical location. Helps for quicker host selection during cluster creation.

Labels

Enter key=value and then press 'enter' or 'space' or use a ',' to input the label.

cluster=clue 

Infrastructure provider credentials

Select a credential 

Pull secret *

```
DESCRIPTOR--: email: jterez@redhat.com; registry.redhat.io -  
{*auth*:NTlzMjU0MDBB8dWhjLTFIaWRhWDBvbHZOWnlkNmtidVNNc3psb0JXTJpleUpoYkdjaU9pSINVelV4TWIKOS5leUp6ZFdJaU9pSTJaamszTjJGbVpUY3lINM  
kkwTURjME9HSmlNakJoWmpWa05Ua3haRFkzWkNKOS5pZGx6YTNUc2swaWc1UiIxQTNRtkVqUDFnemQ5MzBYOGIzOVVrMTZzWEdxVWVqNDVkamVDQ1ZJa
```

Create new VMs

Create new virtual machines in vsphere.

Most of the information is left as default, exceptions are:

- The guest OS is changed to Linux - RHEL 8
- Hardware resources are updated to 4 CPU; 16GB RAM; 120GB disk
- In advance parameters tab, the diskEnableUUID=TRUE is added

- portal.vc.opentlc.com
 - SDDC-Datacenter
 - Templates
 - Workloads
 - sandbox-r2sfl

ba

m

m

m

Actions - sandbox-r2sfl

New Virtual Machine...

Deploy OVF Template...

New Folder

Rename...

Move To...

Add Permission...

Tags & Custom Attributes >

Alarms >

Remove from Inventory

New Virtual Machine

1 Select a creation type

2 Select a name and folder

3 Select a compute resource

4 Select storage

5 Select compatibility

6 Select a guest OS

7 Customize hardware

8 Ready to complete

Select a creation type

How would you like to create a virtual machine?

Create a new virtual machine

Deploy from template

Clone an existing virtual machine

Clone virtual machine to template

Convert template to virtual machine

Clone template to template

This option guides you through creating a new virtual machine. You will be able to customize processors, memory, network connections, and storage. You will need to install a guest operating system after creation.

CANCEL

NEXT

New Virtual Machine

1 Select a creation type

2 Select a name and folder

3 Select a compute resource

4 Select storage

5 Select compatibility

6 Select a guest OS

7 Customize hardware

8 Ready to complete

Select a name and folder

Specify a unique name and target location

Virtual machine name:

worker1

Select a location for the virtual machine.

portal.vc.opentlc.com

SDDC-Datacenter

Templates

Workloads

sandbox-r2sfl

CANCEL

BACK

NEXT

New Virtual Machine

1 Select a creation type

2 Select a name and folder

3 Select a compute resource

4 Select storage

5 Select compatibility

6 Select a guest OS

7 Customize hardware

8 Ready to complete

Select a compute resource

Select the destination compute resource for this operation

▼ SDDC-Datacenter

> Cluster-1

Compatibility

✔ Compatibility checks succeeded.

CANCEL

BACK

NEXT

New Virtual Machine

1 Select a creation type

2 Select a name and folder

3 Select a compute resource

4 Select storage

5 Select compatibility

6 Select a guest OS

7 Customize hardware

8 Ready to complete

Select storage

Select the storage for the configuration and disk files

☐ Encrypt this virtual machine

VM Storage Policy

Disable Storage DRS for this virtual machine

Datastore Default

| | Name | Storage Compatibility | Capacity | Provisioned | Free | Type | Place |
|--|--------------------|-----------------------|-----------|-------------|----------|------|-------|
| | WorkloadDatasto... | -- | 124.42 TB | 156.26 TB | 71.26 TB | vSAN | Loc |

Items per page101 item

Compatibility

vSAN storage consumption would be - disk space and 0 B reserved Flash space.

CANCEL

BACK

NEXT

New Virtual Machine

1 Select a creation type

2 Select a name and folder

3 Select a compute resource

4 Select storage

5 Select compatibility

6 Select a guest OS

7 Customize hardware

8 Ready to complete

Select compatibility

Select compatibility for this virtual machine depending on the hosts in your environment

The host or cluster supports more than one VMware virtual machine version. Select a compatibility for the virtual machine.

Compatible with:

ESXi 6.7 and later

This virtual machine uses hardware version 14, which is compatible with ESXi 6.7 and later. Some virtual machine hardware features are unavailable with this option.

CANCEL

BACK

NEXT

New Virtual Machine

1 Select a creation type

2 Select a name and folder

3 Select a compute resource

4 Select storage

5 Select compatibility

6 Select a guest OS

7 Customize hardware

8 Ready to complete

Select a guest OS

Choose the guest OS that will be installed on the virtual machine

Identifying the guest operating system here allows the wizard to provide the appropriate defaults for the operating system installation.

Guest OS Family:

Linux

Guest OS Version:

Red Hat Enterprise Linux 8 (64-bit)

Compatibility: ESXi 6.7 and later (VM version 14)

CANCEL

BACK

NEXT

New Virtual Machine

1 Select a creation type

2 Select a name and folder

3 Select a compute resource

4 Select storage

5 Select compatibility

6 Select a guest OS

7 Customize hardware

8 Ready to complete

Customize hardware

Configure the virtual machine hardware

Virtual Hardware

VM Options

Advanced Parameters

ADD NEW DEVICE

| | | | |
|-----------------------|-------------------------|---|--|
| > CPU * | 4 | | |
| > Memory * | 16 | GB | |
| > New Hard disk * | 120 | GB | |
| > New SCSI controller | VMware Paravirtual | | |
| > New Network | segment-sandbox-r2sfl | <input checked="" type="checkbox"/> Connected | |
| > New CD/DVD Drive | Client Device | <input type="checkbox"/> Connect At Power On | |
| > Video card | Specify custom settings | | |
| > New SATA Controller | New SATA Controller | | |
| > Other | Additional Hardware | | |

Compatibility: ESXi 6.7 and later (VM version 14)

CANCEL

BACK

NEXT

New Virtual Machine

1 Select a creation type

2 Select a name and folder

3 Select a compute resource

4 Select storage

5 Select compatibility

6 Select a guest OS

7 Customize hardware

8 Ready to complete

Customize hardware

Configure the virtual machine hardware

Virtual Hardware

VM Options

Advanced Parameters

Advanced Configuration Parameters

Modify or add configuration parameters as needed for experimental features or as instructed by technical support. Empty values will be removed (supported on ESXi 6.0 and later).

Attribute

Value

ADD

| Attribute | Value |
|-----------------|-------|
| disk.EnableUUID | TRUE |

CANCEL

BACK

NEXT

New Virtual Machine

1 Select a creation type

2 Select a name and folder

3 Select a compute resource

4 Select storage

5 Select compatibility

6 Select a guest OS

7 Customize hardware

8 Ready to complete

Ready to complete

Click Finish to start creation.

| | |
|-------------------------------|--|
| Virtual machine name | worker1 |
| Folder | sandbox-r2sfl |
| Cluster | Cluster-1 |
| Datastore | WorkloadDatastore |
| Compatibility | ESXi 6.7 and later (VM version 14) |
| Guest OS name | Red Hat Enterprise Linux 8 (64-bit) |
| Virtualization Based Security | Disabled |
| CPUs | 4 |
| Memory | 16 GB |
| NICs | 1 |
| NIC 1 network | segment-sandbox-r2sfl (vmc-hostswitch) |
| NIC 1 type | VMXNET 3 |
| SCSI controller 1 | VMware Paravirtual |
| ▼ New hard disk 1 | |
| Capacity | 120 GB |
| Datastore | WorkloadDatastore |
| Virtual device node | SCSI(0:0) |
| Mode | Dependent |

CANCEL

BACK

FINISH

Advance network configuration

If the hosts need to get advanced network configuration, like is the case in the demo.redhat.com environment, where DHCP is available but the IP is unpredictable, so static IPs must be assigned to the nodes so they can be used in the DNS and LB configurations, then add a nmstateconfig object for each of the nodes that are going to be added to the cluster.

https://access.redhat.com/documentation/en-us/red_hat_advanced_cluster_management_for_kubernetes/2.9/html/clusters/cluster_mce_overview#cim-network-steps

https://access.redhat.com/documentation/es-es/openshift_container_platform/4.12/html/installing/installing-an-on-premise-cluster-with-the-agent-based-installer#sample-ztp-custom-resources_installing-with-agent-based-installer

The nmstateconfig definitions are similar to the following, one for each node in the cluster:

- The namespace is the same as the name of the host inventory as defined in ACM
- The label is obtained from the infraenv object in ACM

Unset

```
$ oc get infraenv -n desligado desligado -o yaml
...
spec:
  agentLabels:
    agentclusterinstalls.extensions.hive.openshift.io/location:
VMwareRedHatOpenShiftInstallation-1lr58
    disco: nected
    cpuArchitecture: x86_64
    ipxeScriptType: DiscoveryImageAlways
    nmStateConfigLabelSelector:
      matchLabels:
        infraenvs.agent-install.openshift.io: desligado
  ...
```

- The MACs are obtained from the vsphere VMs.
- The IPs are assigned as:
 - IP for the node is chosen at random within the network or the one defined in DNS
 - IP for the DNS server is obtained from /etc/resolv.conf in the bastion IP, should be the bastion's IP
 - IP for the default route is obtained from running the command "ip route" in the bastion

Unset

```
apiVersion: agent-install.openshift.io/v1beta1
kind: NMStateConfig
metadata:
  name: master-1
  namespace: hybridcluster
  labels:
    infraenvs.agent-install.openshift.io: hybridcluster
spec:
  config:
    interfaces:
      - name: eth0
        type: ethernet
        state: up
        mac-address: 00:50:56:a2:5d:4d
        ipv4:
          enabled: true
          address:
            - ip: 192.168.64.101
              prefix-length: 24
          dhcp: false
    dns-resolver:
      config:
```

```

    server:
      - 192.168.64.10
  routes:
    config:
      - destination: 0.0.0.0/0
        next-hop-address: 192.168.64.1
        next-hop-interface: eth0
        table-id: 254
  interfaces:
    - name: "eth0"
      macAddress: 00:50:56:a2:5d:4d
---
apiVersion: agent-install.openshift.io/v1beta1
kind: NMStateConfig
metadata:
  name: master-2
  namespace: hybridcluster
  labels:
    infraenvs.agent-install.openshift.io: hybridcluster
spec:
  config:
    interfaces:
      - name: eth0
        type: ethernet
        state: up
        mac-address: 00:50:56:a2:e9:43
        ipv4:
          enabled: true
          address:
            - ip: 192.168.64.102
              prefix-length: 24
        dhcp: false
        dns-resolver:
          config:
            server:
              - 192.168.64.10
            routes:
              config:
                - destination: 0.0.0.0/0
                  next-hop-address: 192.168.64.1
                  next-hop-interface: eth0
                  table-id: 254
    interfaces:
      - name: "eth0"
        macAddress: 00:50:56:a2:e9:43
---
apiVersion: agent-install.openshift.io/v1beta1
kind: NMStateConfig

```

```

metadata:
  name: master-3
  namespace: hybridcluster
  labels:
    infraenvs.agent-install.openshift.io: hybridcluster
spec:
  config:
    interfaces:
      - name: eth0
        type: ethernet
        state: up
        mac-address: 00:50:56:a2:4a:1d
        ipv4:
          enabled: true
          address:
            - ip: 192.168.64.103
              prefix-length: 24
        dhcp: false
        dns-resolver:
          config:
            server:
              - 192.168.64.10
        routes:
          config:
            - destination: 0.0.0.0/0
              next-hop-address: 192.168.64.1
              next-hop-interface: eth0
              table-id: 254
    interfaces:
      - name: "eth0"
        macAddress: 00:50:56:a2:4a:1d

```

Apply the objects to the cluster:

```

Unset
$ oc apply -f nmstate-master-1.yaml
$ oc apply -f nmstate-master-2.yaml
$ oc apply -f nmstate-master-3.yaml

$ oc get nmstateconfig -n hybridcluster
NAME          AGE
master-1      6m25s
master-2      5m40s
master-3      5m35s

```


After applying the nmstateconfig objects, a new discovery ISO Image is created

Download the discovery ISO

From Host inventory -> Add hosts -> With Discovery ISO

You are logged in as a temporary administrative user. Update the [cluster OAuth configuration](#) to allow others to log in.

Host inventory > carlito

carlito

Details Hosts

Information & Troubleshooting

Minimum hardware requirements Hosts not showing up?

0 selected Find by hostname Status Actions

| Hostname | Discovery type | Status | Cluster | Discovered on | CPU Cores |
|----------|----------------|--------|---------|---------------|-----------|
|----------|----------------|--------|---------|---------------|-----------|

With Discovery ISO
Discover hosts by booting a discovery image

With IPXE
Use when you have an IPXE server that has already been set up

Baseboard Management Controller (BMC)

With BMC form
Discover a single host via Baseboard Management Controller

By uploading a YAML
Discover multiple hosts by providing YAML with Bare Metal Host definitions

Download Discovery ISO

Add host



✓ **Discovery ISO is ready to be downloaded.**

Adding hosts instructions

1. Download the Discovery ISO (onto a USB drive, attach it to a virtual media, etc.) and use it to boot your hosts.
2. Keep the Discovery ISO media connected to the device throughout the installation process and set each host to boot **only one time** from this device.
3. Booted hosts should appear in the host inventory table. This might take a few minutes.

i To use static network configuration, follow the steps listed in the documentation.

[View documentation](#)

Discovery ISO URL

`https://assisted-image-service-multicluster-engine.apps.n...`



Command to download the ISO:

`wget -O discovery.iso 'https://assisted-image-service-mult...`



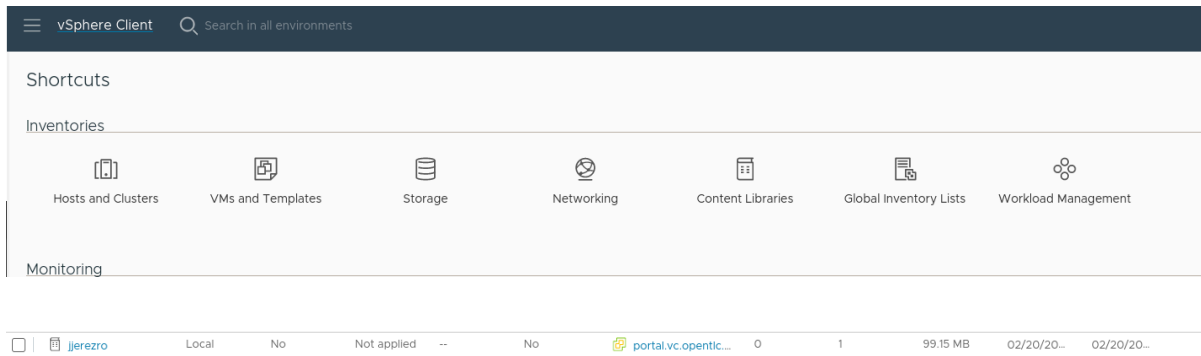
i Never share your downloaded ISO with anyone else.

[Download Discovery ISO](#)

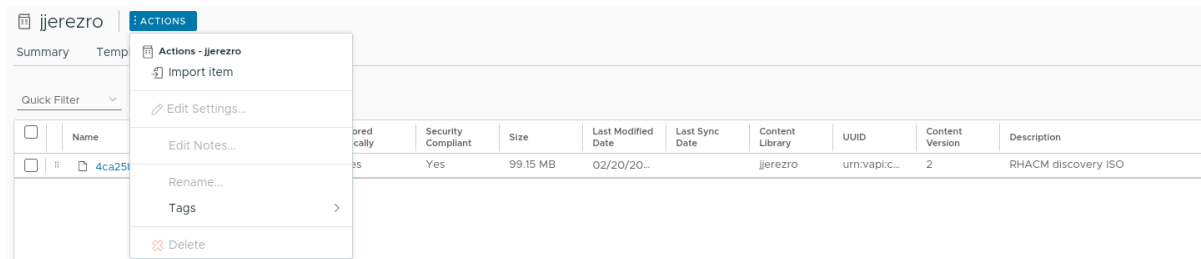
[Close](#)

Upload the discovery ISO to vsphere

Create a content library by going to Content Libraries -> Create

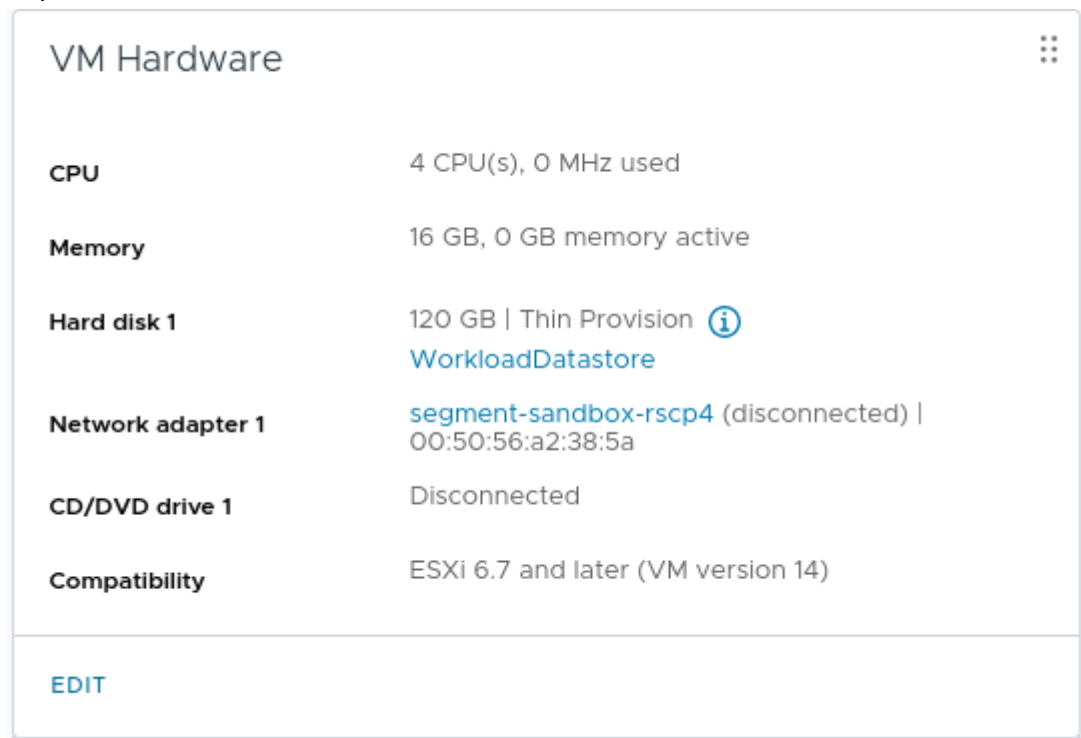


Once created, go to the content library -> Actions -> Import Item



Boot the VMs with the discovery ISO

Edit the VM hardware and attach the discovery ISO from the content library to the CD.
 Enable the option connect at Power On so that the VM boots from the ISO.
 Repeat for all VMs



Edit Settings | master1

Virtual Hardware

VM Options

Advanced Parameters

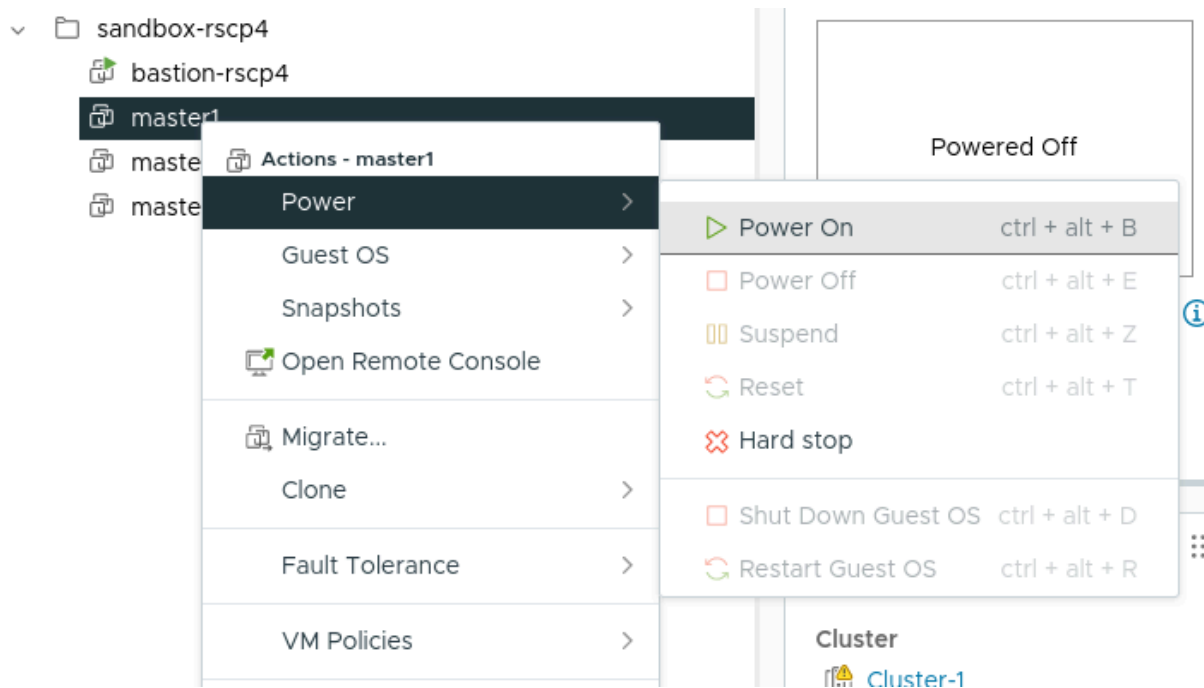
ADD NEW DEVICE

| | | | |
|---------------------|---|-----------|-----------|
| > CPU | 4 | | |
| > Memory | 16 | GB | |
| > Hard disk 1 | 120 | GB | |
| > SCSI controller 0 | VMware Paravirtual | | |
| > Network adapter 1 | segment-sandbox-rscp4 | Connected | |
| > CD/DVD drive 1 * | Content Library ISO File | | |
| Status | <input checked="" type="checkbox"/> Connect At Power On | | |
| CD/DVD Media | [contentLib] /jjerezro/00bc4at | | BROWSE... |
| Device Mode | Emulate CD-ROM | | |
| Virtual Device Node | SATA controller 0 SATA(0:0) CD/DVD drive 1 | | |
| > Video card | Specify custom settings | | |
| > SATA controller 0 | AHCI | | |
| > Other | Additional Hardware | | |

CANCEL

OK

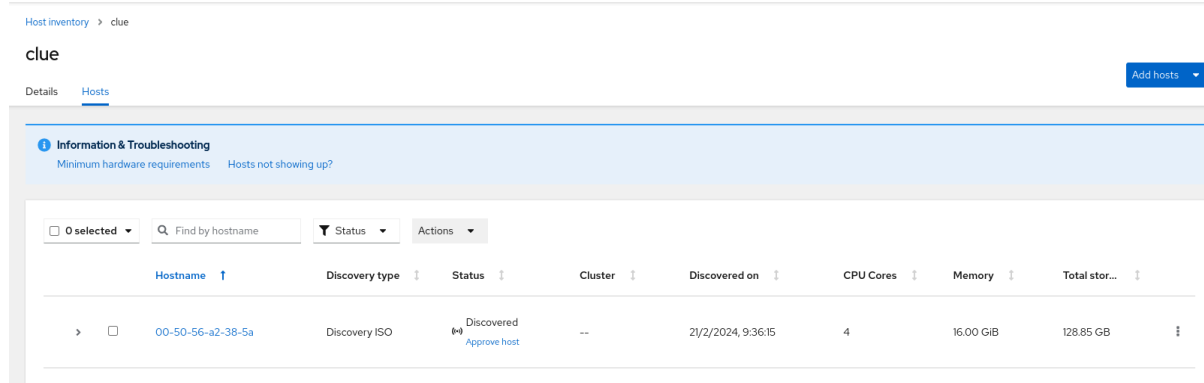
Boot the VMs, one by one



After a couple minutes the VM should be fully booted

```
Red Hat Enterprise Linux CoreOS 414.92.202305090606-0 (Plow) 4.14
SSH host key: SHA256:rmdmxBsDFph/Fw0xxPpFPDAmt7U2VU3hUIfyjie1FKQ (ED25519)
SSH host key: SHA256:33CcEYjbbEhh0VbDng7yNM5CDom7p005R7RWij0nY7s (ECDSA)
SSH host key: SHA256:EJwuHX3HxcPowrqh+JoU1kz/xv2B2h2XUgMS08wMcIg (RSA)
ens192: 192.168.188.100
Ignition: ran on 2024/02/21 08:35:43 UTC (this boot)
Ignition: user-provided config was applied
localhost login: _
```

And the new host should appear in the Hosts list of the ACM inventory:



If the name cannot be resolved by DNS in the ACM host, it appears as the mac address, this can be changed here.

Change hostname

This name will replace the original discovered hostname

Discovered hostname

00-50-56-a2-38-5a

New hostname *

master1

✓

Change

master1

master1-din

✓ 1-63 characters

✓ Must be unique

✓ Use lowercase alphanumeric characters, dot (.) or hyphen (-)

✓ Must start and end with an lowercase alphanumeric character

✓ Do not use forbidden words, for example: "localhost".

Approve the discovered host

Approve host to join infrastructure environment

Make sure that you expect and recognize the host before approving.

Hostname: master1

Approve host

Discovery ISO

Discovered

Approve host

The final result looks like this:

0 selected

Find by hostname

Status

Actions

| | Hostname | Discovery type | Status | Cluster | Discovered on | CPU Cores | Memory | Total storage |
|----------------------------|----------|----------------|-------------|---------|--------------------|-----------|-----------|---------------|
| > <input type="checkbox"/> | master1 | Discovery ISO | + Available | -- | 21/2/2024, 9:36:15 | 4 | 16.00 GiB | 128.85 GB |
| > <input type="checkbox"/> | master2 | Discovery ISO | + Available | -- | 21/2/2024, 9:40:54 | 4 | 16.00 GiB | 128.85 GB |
| > <input type="checkbox"/> | master3 | Discovery ISO | + Available | -- | 21/2/2024, 9:41:08 | 4 | 16.00 GiB | 128.85 GB |

Create the cluster

https://access.redhat.com/documentation/en-us/red_hat_advanced_cluster_management_for_kubernetes/2.9/html/clusters/cluster_mce_overview#creating-a-cluster-on-premises

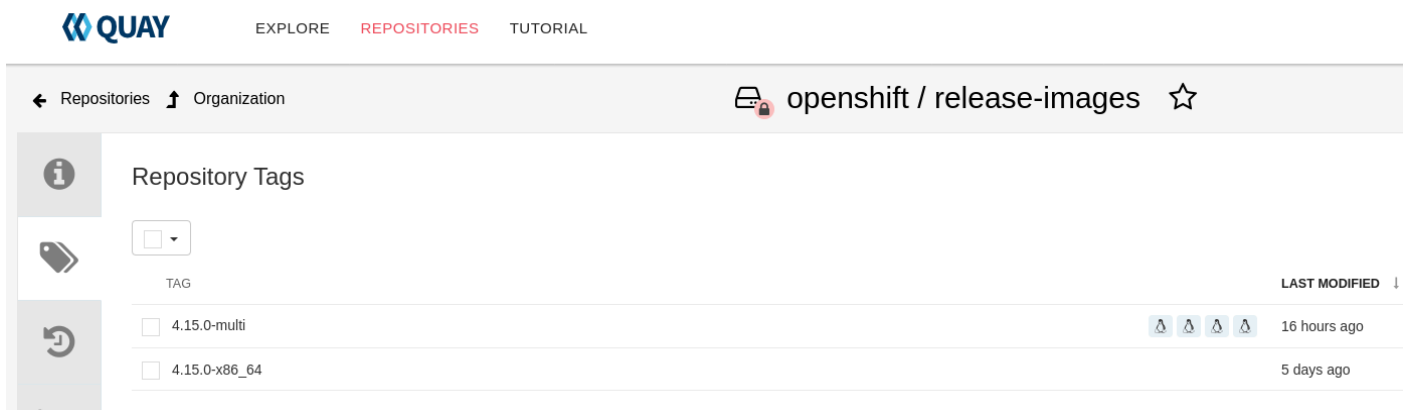
If the cluster is created in a disconnected environment you need to create one (or more) cluster image sets that reference the images by digest and not by tag.

The list of cluster image sets is maintained in the git repository [acm-hive-openshift-releases](#). Select the branch that matches the multicluster-engine operator version, backplane-2.4 in this case.

The cluster image set looks like this

```
Unset
apiVersion: hive.openshift.io/v1
kind: ClusterImageSet
metadata:
  labels:
    channel: stable
    visible: 'false'
  name: img4.15.0-x86-64-appsub
spec:
  releaseImage: quay.io/openshift-release-dev/ocp-release:4.15.0-x86_64
```

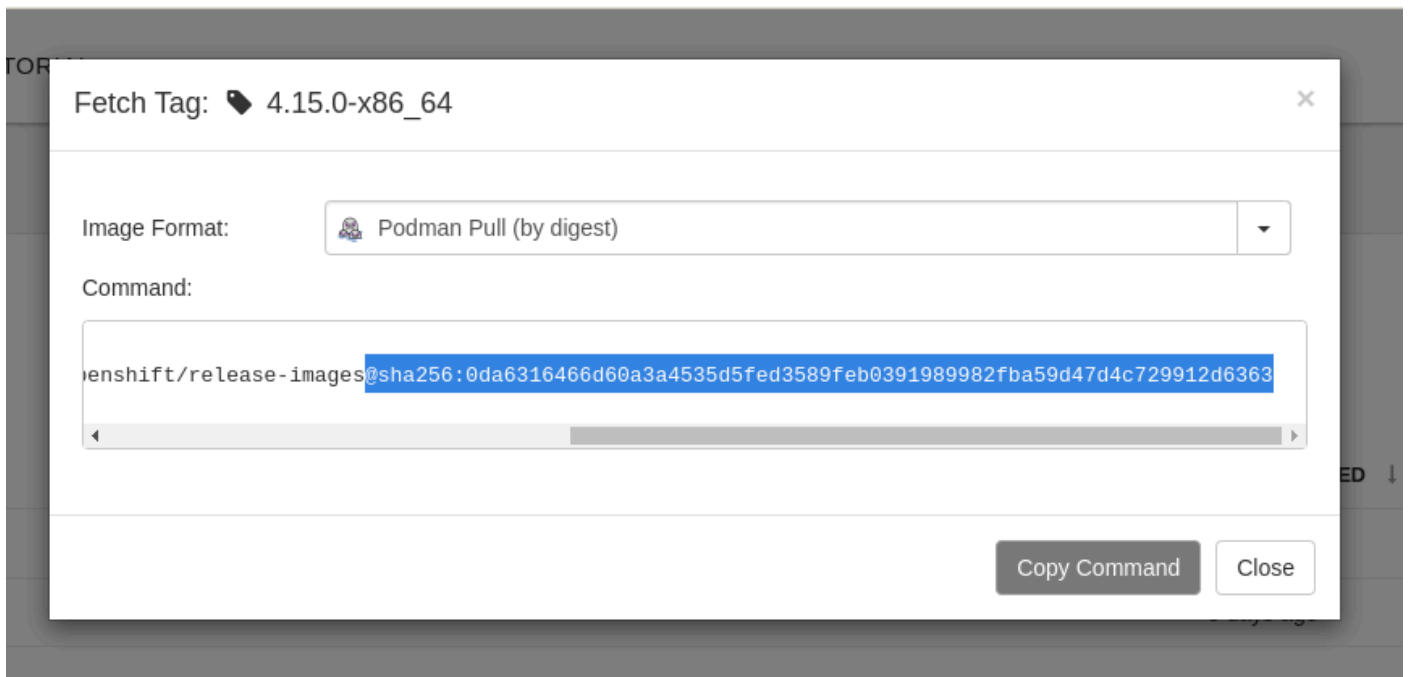
Make sure that the releaseImage reference exists in the mirror registry



The screenshot shows the Quay.io web interface. At the top, there's a navigation bar with the Quay logo and links for EXPLORE, REPOSITORIES, and TUTORIAL. Below this, a breadcrumb trail shows 'Repositories' and 'Organization'. The main content area is titled 'Repository Tags' and shows a list of tags for the 'openshift / release-images' repository. The tags are '4.15.0-multi' and '4.15.0-x86_64'. The '4.15.0-multi' tag is selected and shows a 'LAST MODIFIED' time of '16 hours ago'. The '4.15.0-x86_64' tag shows a 'LAST MODIFIED' time of '5 days ago'.

| TAG | LAST MODIFIED |
|--|---------------|
| <input checked="" type="checkbox"/> 4.15.0-multi | 16 hours ago |
| <input type="checkbox"/> 4.15.0-x86_64 | 5 days ago |

Modify the definition to include the sha256 hash shown in quay.



It is likely that the cluster image set already exists, change the name and make it visible so it shows up in the web console

```
Unset
apiVersion: hive.openshift.io/v1
kind: ClusterImageSet
metadata:
  labels:
    channel: stable
    visible: 'true'
  name: img4.15.0-x86-64-appsub-stable
spec:
  releaseImage:
    quay.io/openshift-release-dev/ocp-release@sha256:0da6316466d60a3a4535d5fed3589feb0391989982fba59d47d4c729912d6363
```

Apply the new cluster image set

```
Unset
$ oc apply -f clusterimageset-4.15.0.yaml
clusterimageset.hive.openshift.io/img4.15.0-x86-64-appsub-stable created
```

Later you have to make sure that the clusterimageset reference is in the AgentClusterInstall yaml definition. The new cluster image set should be available in the dropdown list in the web interface but it takes a while to appear.

dynamic.opentlc.com

All DNS records must be subdomains of this base and include the cluster name. This cannot be changed after cluster installation. The full cluster address will be:
llr58.dynamic.opentlc.com

OpenShift version *

OpenShift img4.15.0-x86-64-appsub-stable

OpenShift img4.15.0-x86-64-appsub-stable

OpenShift 4.15.0

OpenShift 4.14.15

OpenShift 4.13.36

OpenShift 4.12.51

☐ Use arm64 CPU architecture ?
Make sure all the hosts are using arm64 CPU architecture.

Additional labels

Enter key=value, then press return, space, or comma

If you modify any of the cluster image sets you may have to restart the assisted service pod to remove the cached information so that the new definition is used.

Go to Infrastructure -> Clusters -> Create cluster

Clusters ⓘ

[Cluster list](#) [Cluster sets](#) [Cluster pools](#) [Discovered clusters](#) [Get started with Multicluster](#)

☐

Search

Filter

Create cluster

Import cluster

Actions

1-1 of 1

| Name | Namespace | Status | Infrastructure | Control plane type | Distribution version | Labels | Nodes | Add-ons | Creation date |
|--|---------------|--------|----------------|--------------------|--|--|-------|---------|---------------------|
| <input type="checkbox"/> local-cluster | local-cluster | Ready | VMware vSphere | Hub | OpenShift 4.14.11 Upgrade available | openshiftVersion-major=4 openshiftVersion-major-minor=4.14 velero.io/exclude-from-backup=true 15 more | 6 | 8 | 19/2/2024, 19:11:04 |

Select Host inventory



Host inventory

A Red Hat OpenShift cluster that is running on available hosts from your on-premise inventory; bare metal or virtualized.

In this case, choose Standalone as opposed to Hosted control plane cluster.

Standalone

Run an OpenShift cluster where the control plane and data plane are coupled. The control plane is hosted by a dedicated group of physical or virtual nodes and the network stack is shared.

- ✓ Increased resiliency with closely interconnected control plane and worker nodes.
- ✓ Provide customized control plane cluster configuration.
 - Standard
 - Single node OpenShift
 - Three-node cluster

Use existing hosts

Use existing hosts

Create a cluster from hosts that have been discovered and made available in your host inventory.

Enter the cluster details.

The cluster name must be the GUID assigned by demo.redhat.com

GUID

rscp4

The base domain in the case of demo.redhat.com is **dynamic.opentlc.com**

Add a pull secret

Cluster details

Infrastructure provider credential ⓘ

Select a credential ▼

Cluster name *

rscp4 ✓

Cluster set ⓘ

default ⓘ ▼

[Manage cluster sets](#)

Base domain *

dynamic.opentlc.com

All DNS records must be subdomains of this base and include the cluster name. This cannot be changed after cluster installation. The full cluster address will be: rscp4.dynamic.opentlc.com

OpenShift version *

OpenShift 4.14.13 ▼

☐ Install single node OpenShift (SNO)
SNO enables you to install OpenShift using only one host.

☐ Use arm64 CPU architecture ⓘ
Make sure all the hosts are using arm64 CPU architecture.

Next

Back

Cancel

Assign the hosts for the cluster. In this case a 3 node compact cluster is being created, and the hosts are being assigned manually

Cluster hosts

At least 3 hosts are required that are capable of functioning as control plane nodes.

[Minimum hardware requirements](#)

☐ Auto-select hosts

Host locations [?](#)

VMWareCloudPublic...

Type or select location(s)

✕ ▼

Select one or more locations to view hosts

Labels matching hosts

cluster=clue ✕

app=frontend

✕ ▼

Provide as many labels as you can to narrow the list to relevant hosts.

Displaying only hosts with x86_64 architecture in the table.

| | Hostname ↑ | Infrastruct... ↓ | Status ↓ | Role ↓ | CPU Cores ↓ | Memory ↓ | Total stora... ↓ |
|---|---|------------------|--------------------------|---------------|-------------|-----------|------------------|
| > | <input checked="" type="checkbox"/> master1 | clue | + Available | Auto-assign ▼ | 4 | 16.00 GiB | 128.85 GB |

Next

Back

Cancel

After clicking Next on the previous page, the hosts are checked and bound

| | Hostname ↑ | Infrastruct... ↓ | Status ↓ | Role ↓ | CPU C... ↓ | Memo... ↓ | Total s... ↓ |
|---|--|------------------|------------------------|----------------------|------------|-----------|--------------|
| > | <input checked="" type="checkbox"/> master-0.rscp4.dynamic.opentlc.com | clue | 🔗 Binding | Auto-assign ▼ | 4 | 16.00 GiB | 128.85 GB |
| > | <input checked="" type="checkbox"/> master-1.rscp4.dynamic.opentlc.com | clue | ✅ Ready | Control plane node ▼ | 4 | 16.00 GiB | 128.85 GB |
| > | <input checked="" type="checkbox"/> master-2.rscp4.dynamic.opentlc.com | clue | 🔗 Binding | Auto-assign ▼ | 4 | 16.00 GiB | 128.85 GB |

Next

Back

Cancel

🔄 Binding hosts...

Add the networking configuration.

In this case User-Managed networking is used because we provide the LB and DNS. This is defined in the DNS server and is populated by the ansible playbook that was executed earlier.

Add a public ssh key to propagate to the nodes.

It will take a short while until the hosts status goes from insufficient to Ready.

Host inventory

| | Hostname ↑ | Role ↓ | Status ↓ | Active NIC ↓ |
|---|------------------------------------|----------------------|----------------------|--------------|
| > | master-0.rscp4.dynamic.opentlc.com | Auto-assign ▼ | ✅ Ready | - |
| > | master-1.rscp4.dynamic.opentlc.com | Control plane node ▼ | ✅ Ready | - |
| > | master-2.rscp4.dynamic.opentlc.com | Control plane node ▼ | ✅ Ready | - |

Next

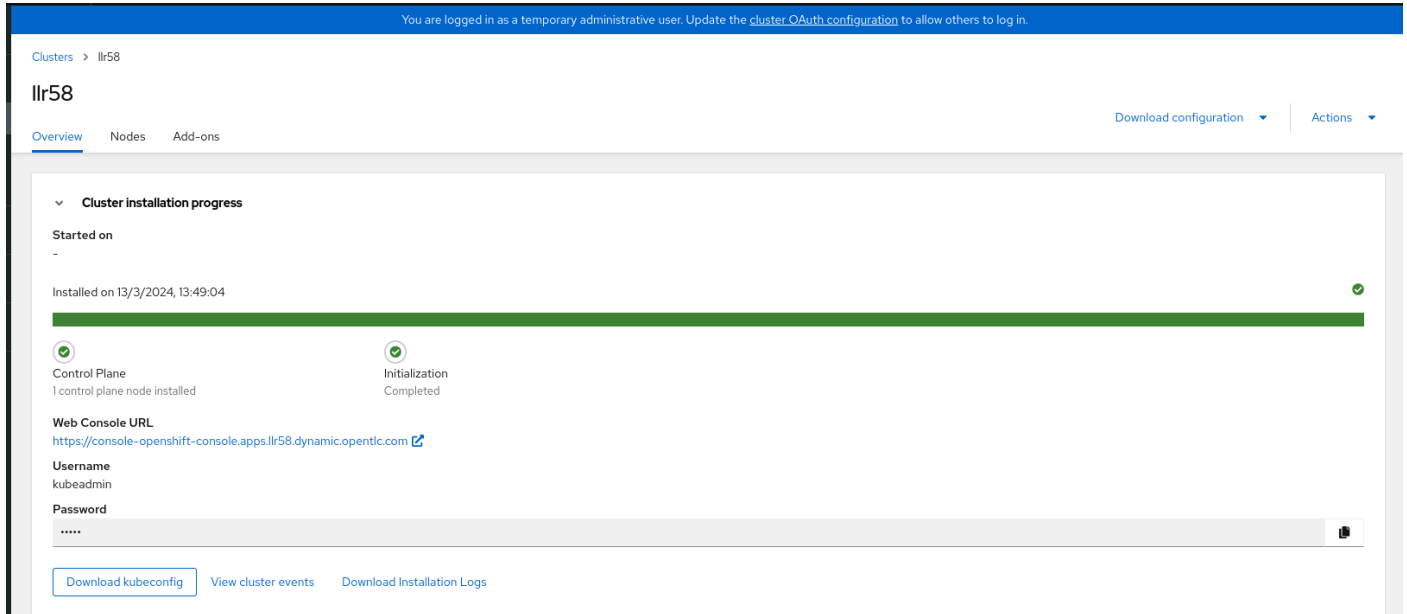
Back

Cancel

The next page shows the summary before proceeding to the actual cluster installation.

Follow the logs from the assisted service pod and the logs from the nodes where OCP is being installed to see the evolution of the installation.

After some time, no less than 40 minutes the cluster is installed



In the case of the demo.redhat.com platform, the environment is not actually disconnected. To verify if any images were pulled from the Internet check the crictl logs from the nodes:

Unset

```
[core@master-0 ~]$ journalctl -u crio --no-pager | grep "Trying to access" | less -S
...
171Z" level=info msg="Trying to access
\"bastion-995pv.995pv.dynamic.opentlc.com:443/multiclustere-engine/cluster-proxy-addon-rhel
8@sha256:add356580a3aa058d6e26e72023fd605726fbe1f65d5169c38b>

097Z" level=info msg="Trying to access
\"bastion-995pv.995pv.dynamic.opentlc.com:443/multiclustere-engine/cluster-proxy-addon-rhel
8@sha256:add356580a3aa058d6e26e72023fd605726fbe1f65d5169c38b>

659Z" level=info msg="Trying to access
\"registry.redhat.io/rhacm2/search-collector-rhel8@sha256:181655607a19a17083f24dc34e018804
eed046ef3a1f0f68673cc24851caf5f0\"

075Z" level=info msg="Trying to access
\"bastion-995pv.995pv.dynamic.opentlc.com:443/multiclustere-engine/cluster-proxy-addon-rhel
8@sha256:add356580a3aa058d6e26e72023fd605726fbe1f65d5169c38b>
```

```
964Z" level=info msg="Trying to access
\"registry.redhat.io/rhacm2/search-collector-rhel8@sha256:181655607a19a17083f24dc34e018804
eed046ef3a1f0f68673cc24851caf5f0\""

000Z" level=info msg="Trying to access
\"bastion-995pv.995pv.dynamic.opentlc.com:443/multiclustere-engine/cluster-proxy-addon-rhel
8@sha256:add356580a3aa058d6e26e72023fd605726fbe1f65d5169c38b>

717Z" level=info msg="Trying to access
\"bastion-995pv.995pv.dynamic.opentlc.com:443/multiclustere-engine/multicloud-manager-rhel8
@sha256:7b6b5b839fd1df2d4faaecf93279000786e3ccdc3f332aea5566>

076Z" level=info msg="Trying to access
\"registry.redhat.io/rhacm2/kube-rbac-proxy-rhel8@sha256:eb13f6fa0768104e940701c50914d0d1b
049177f75ab2dd98abf123468e08958\""

877Z" level=info msg="Trying to access
\"bastion-995pv.995pv.dynamic.opentlc.com:443/multiclustere-engine/multicloud-manager-rhel8
@sha256:7b6b5b839fd1df2d4faaecf93
```

Most of the images come from the mirror registry (`bastion-995pv`) but some are pulled directly from `registry.redhat.io`

This means that the mirror-config configmap should have included entries for these repositories.