



# **Installation overview**

## **Astra Control Center**

NetApp

February 12, 2024

This PDF was generated from <https://docs.netapp.com/us-en/astra-control-center-2211/get-started/cert-manager-prereqs.html> on February 12, 2024. Always check docs.netapp.com for the latest.

# Table of Contents

- Installation overview ..... 1
  - Install Astra Control Center using the standard process ..... 1
  - Install Astra Control Center using OpenShift OperatorHub ..... 31
  - Install Astra Control Center with a Cloud Volumes ONTAP storage backend ..... 39

# Installation overview

Choose and complete one of the following Astra Control Center installation procedures:

- [Install Astra Control Center using the standard process](#)
- (If you use Red Hat OpenShift) [Install Astra Control Center using OpenShift OperatorHub](#)
- [Install Astra Control Center with a Cloud Volumes ONTAP storage backend](#)

Depending on your environment, there might be additional configuration needed after you install Astra Control Center:

- [Configure Astra Control Center after installation](#)

## Install Astra Control Center using the standard process

To install Astra Control Center, download the installation bundle from the NetApp Support Site and perform the following steps. You can use this procedure to install Astra Control Center in internet-connected or air-gapped environments.

### Other installation procedures

- **Install with RedHat Openshift OperatorHub:** Use this [alternative procedure](#) to install Astra Control Center on Openshift using OperatorHub.
- **Install in the public cloud with Cloud Volumes ONTAP backend:** Use [these procedures](#) to install Astra Control Center in Amazon Web Services (AWS), Google Cloud Platform (GCP), or Microsoft Azure with a Cloud Volumes ONTAP storage backend.

For a demonstration of the Astra Control Center installation process, see [this video](#).

### What you'll need

- [Before you begin installation, prepare your environment for Astra Control Center deployment.](#)
- If you have configured or want to configure pod security policies in your environment, familiarize yourself with pod security policies and how they affect Astra Control Center installation. See [Understand pod security policy restrictions](#).
- Ensure all API services are in a healthy state and available:

```
kubectl get apiservices
```

- Ensure the Astra FQDN you plan to use is routable to this cluster. This means that you either have a DNS entry in your internal DNS server or you are using a core URL route that is already registered.
- If a cert manager already exists in the cluster, you need to perform some [prerequisite steps](#) so that Astra Control Center does not attempt to install its own cert manager. By default, Astra Control Center installs its own cert manager during installation.

### About this task

The Astra Control Center installation process helps you to do the following:

- Install the Astra components into the `netapp-acc` (or custom-named) namespace.
- Create a default Astra Control Owner admin account.
- Establish an administrative user email address and default initial setup password. This user is assigned the Owner role that is needed for first time login to the UI.
- Determine that all Astra Control Center pods are running.
- Install the Astra Control Center UI.



Do not delete the Astra Control Center operator (for example, `kubectl delete -f astra_control_center_operator_deploy.yaml`) at any time during Astra Control Center installation or operation to avoid deleting pods.

## Steps

To install Astra Control Center, do the following steps:

- [Download and extract Astra Control Center](#)
- [Install the NetApp Astra kubectl plugin](#)
- [Add the images to your local registry](#)
- [Set up namespace and secret for registries with auth requirements](#)
- [Install the Astra Control Center operator](#)
- [Configure Astra Control Center](#)
- [Complete Astra Control Center and operator installation](#)
- [Verify system status](#)
- [Set up ingress for load balancing](#)
- [Log in to the Astra Control Center UI](#)

## Download and extract Astra Control Center

1. Go to the [Astra Control Center Evaluation downloads page](#) on the NetApp Support Site.
2. Download the bundle containing Astra Control Center (`astra-control-center-[version].tar.gz`).
3. (Recommended but optional) Download the certificates and signatures bundle for Astra Control Center (`astra-control-center-certs-[version].tar.gz`) to verify the signature of the bundle:

```
tar -vxzf astra-control-center-certs-[version].tar.gz
```

```
openssl dgst -sha256 -verify certs/AstraControlCenter-public.pub
-signature certs/astra-control-center-[version].tar.gz.sig astra-
control-center-[version].tar.gz
```

The output will show `Verified OK` after successful verification.

4. Extract the images from the Astra Control Center bundle:

```
tar -vxzf astra-control-center-[version].tar.gz
```

## Install the NetApp Astra kubectl plugin

The NetApp Astra kubectl command line plugin saves time when performing common tasks associated with deploying and upgrading Astra Control Center.

### What you'll need

NetApp provides plugin binaries for different CPU architectures and operating systems. You need to know which CPU and operating system you have before you perform this task.

### Steps

1. List the available NetApp Astra kubectl plugin binaries, and note the name of the file you need for your operating system and CPU architecture:



The kubectl plugin library is part of the tar bundle and is extracted into the folder `kubectl-astra`.

```
ls kubectl-astra/
```

2. Move the correct binary into the current path and rename it to `kubectl-astra`:

```
cp kubectl-astra/<binary-name> /usr/local/bin/kubectl-astra
```

## Add the images to your local registry

1. Complete the appropriate step sequence for your container engine:

## Docker

- a. Change to the root directory of the tarball. You should see this file and directory:

```
acc.manifest.bundle.yaml  
acc/
```

- b. Push the package images in the Astra Control Center image directory to your local registry. Make the following substitutions before running the `push-images` command:

- Replace `<BUNDLE_FILE>` with the name of the Astra Control bundle file (`acc.manifest.bundle.yaml`).
- Replace `<MY_FULL_REGISTRY_PATH>` with the URL of the Docker repository; for example, `"https://<docker-registry>"`.
- Replace `<MY_REGISTRY_USER>` with the user name.
- Replace `<MY_REGISTRY_TOKEN>` with an authorized token for the registry.

```
kubectrl astra packages push-images -m <BUNDLE_FILE> -r  
<MY_FULL_REGISTRY_PATH> -u <MY_REGISTRY_USER> -p  
<MY_REGISTRY_TOKEN>
```

## Podman

- a. Change to the root directory of the tarball. You should see this file and directory:

```
acc.manifest.bundle.yaml  
acc/
```

- b. Log in to your registry:

```
podman login <YOUR_REGISTRY>
```

- c. Prepare and run one of the following scripts that is customized for the version of Podman you use. Substitute `<MY_FULL_REGISTRY_PATH>` with the URL of your repository that includes any sub-directories.

```
<strong>Podman 4</strong>
```

```

export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=22.11.0-82
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image: //'')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/${
PACKAGEVERSION}/${astraImageNoPath}
done

```

**Podman 3**

```

export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=22.11.0-82
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image: //'')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/${
PACKAGEVERSION}/${astraImageNoPath}
done

```



The image path the script creates should resemble the following, depending on your registry configuration: <https://netappdownloads.jfrog.io/docker-astra-control-prod/netapp/astra/acc/22.11.0-82/image:version>

## Set up namespace and secret for registries with auth requirements

1. Export the KUBECONFIG for the Astra Control Center host cluster:

```
export KUBECONFIG=[file path]
```



Before you complete the installation, be sure your KUBECONFIG is pointing to the cluster where you want to install Astra Control Center. The KUBECONFIG can contain only one context.

2. If you use a registry that requires authentication, you need to do the following:

a. Create the `netapp-acc-operator` namespace:

```
kubectl create ns netapp-acc-operator
```

Response:

```
namespace/netapp-acc-operator created
```

b. Create a secret for the `netapp-acc-operator` namespace. Add Docker information and run the following command:



The placeholder `your_registry_path` should match the location of the images that you uploaded earlier (for example, `[Registry_URL]/netapp/astra/astracc/22.11.0-82`).

```
kubectl create secret docker-registry astra-registry-cred -n netapp-acc-operator --docker-server=[your_registry_path] --docker-username=[username] --docker-password=[token]
```

Sample response:

```
secret/astra-registry-cred created
```



If you delete the namespace after the secret is generated, recreate the namespace and then regenerate the secret for the namespace.

c. Create the `netapp-acc` (or custom-named) namespace.

```
kubectl create ns [netapp-acc or custom namespace]
```

Sample response:

```
namespace/netapp-acc created
```

d. Create a secret for the `netapp-acc` (or custom-named) namespace. Add Docker information and run the following command:



```
kubectl create secret docker-registry astra-registry-cred -n [netapp-acc or custom namespace] --docker-server=[your_registry_path] --docker-username=[username] --docker-password=[token]
```

## Response

```
secret/astra-registry-cred created
```

## Install the Astra Control Center operator

1. Change the directory:

```
cd manifests
```

2. Edit the Astra Control Center operator deployment YAML (`astra_control_center_operator_deploy.yaml`) to refer to your local registry and secret.

```
vim astra_control_center_operator_deploy.yaml
```



An annotated sample YAML follows these steps.

- a. If you use a registry that requires authentication, replace the default line of `imagePullSecrets: []` with the following:

```
imagePullSecrets:
- name: astra-registry-cred
```

- b. Change `[your_registry_path]` for the `kube-rbac-proxy` image to the registry path where you pushed the images in a [previous step](#).
- c. Change `[your_registry_path]` for the `acc-operator-controller-manager` image to the registry path where you pushed the images in a [previous step](#).

```
<strong>astra_control_center_operator_deploy.yaml</strong>
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    control-plane: controller-manager
```

```

name: acc-operator-controller-manager
namespace: netapp-acc-operator
spec:
  replicas: 1
  selector:
    matchLabels:
      control-plane: controller-manager
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        control-plane: controller-manager
    spec:
      containers:
        - args:
            - --secure-listen-address=0.0.0.0:8443
            - --upstream=http://127.0.0.1:8080/
            - --logtostderr=true
            - --v=10
            image: [your_registry_path]/kube-rbac-proxy:v4.8.0
          name: kube-rbac-proxy
          ports:
            - containerPort: 8443
              name: https
        - args:
            - --health-probe-bind-address=:8081
            - --metrics-bind-address=127.0.0.1:8080
            - --leader-elect
          env:
            - name: ACCOP_LOG_LEVEL
              value: "2"
            - name: ACCOP_HELM_INSTALLTIMEOUT
              value: 5m
            image: [your_registry_path]/acc-operator:[version x.y.z]
          imagePullPolicy: IfNotPresent
          livenessProbe:
            httpGet:
              path: /healthz
              port: 8081
              initialDelaySeconds: 15
              periodSeconds: 20
          name: manager
          readinessProbe:
            httpGet:
              path: /readyz

```

```

        port: 8081
        initialDelaySeconds: 5
        periodSeconds: 10
    resources:
        limits:
            cpu: 300m
            memory: 750Mi
        requests:
            cpu: 100m
            memory: 75Mi
    securityContext:
        allowPrivilegeEscalation: false
imagePullSecrets: []
    securityContext:
        runAsUser: 65532
    terminationGracePeriodSeconds: 10

```

### 3. Install the Astra Control Center operator:

```
kubectl apply -f astra_control_center_operator_deploy.yaml
```

#### Sample response:

```

namespace/netapp-acc-operator created
customresourcedefinition.apiextensions.k8s.io/astracontrolcenters.astra.
netapp.io created
role.rbac.authorization.k8s.io/acc-operator-leader-election-role created
clusterrole.rbac.authorization.k8s.io/acc-operator-manager-role created
clusterrole.rbac.authorization.k8s.io/acc-operator-metrics-reader
created
clusterrole.rbac.authorization.k8s.io/acc-operator-proxy-role created
rolebinding.rbac.authorization.k8s.io/acc-operator-leader-election-
rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/acc-operator-manager-
rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/acc-operator-proxy-
rolebinding created
configmap/acc-operator-manager-config created
service/acc-operator-controller-manager-metrics-service created
deployment.apps/acc-operator-controller-manager created

```

### 4. Verify pods are running:

```
kubectl get pods -n netapp-acc-operator
```

## Configure Astra Control Center

1. Edit the Astra Control Center custom resource (CR) file (`astra_control_center.yaml`) to make account, support, registry, and other necessary configurations:

```
vim astra_control_center.yaml
```



An annotated sample YAML follows these steps.

2. Modify or confirm the following settings:

accountName

| Setting     | Guidance  | Type   | Example |
|-------------|---|--------|---------|
| accountName | Change the accountName string to the name you want to associate with the Astra Control Center account. There can be only one accountName. | string | Example |

astraVersion

| Setting      | Guidance  | Type   | Example    |
|--------------|---|--------|------------|
| astraVersion | The version of Astra Control Center to deploy. No action is needed for this setting as the value will be pre-populated. | string | 22.11.0-82 |

| Setting      | Guidance  | Type   | Example                        |
|--------------|---|--------|--------------------------------|
| astraAddress | <p>Change the <code>astraAddress</code> string to the FQDN (recommended) or IP address you want to use in your browser to access Astra Control Center. This address defines how Astra Control Center will be found in your data center and is the same FQDN or IP address you provisioned from your load balancer when you completed <a href="#">Astra Control Center requirements</a>.</p> <p>NOTE: Do not use <code>http://</code> or <code>https://</code> in the address. Copy this FQDN for use in a <a href="#">later step</a>.</p> | string | <code>astra.example.com</code> |

## autoSupport

Your selections in this section determine whether you will participate in NetApp's pro-active support application, NetApp Active IQ, and where data is sent. An internet connection is required (port 442), and all support data is anonymized.

| Setting                           | Use  | Guidance  | Type    | Example   |
|-----------------------------------|--|---|---------|---|
| <code>autoSupport.enrolled</code> | Either <code>enrolled</code> or <code>url</code> fields must be selected | Change <code>enrolled</code> for AutoSupport to <code>false</code> for sites without internet connectivity or retain <code>true</code> for connected sites. A setting of <code>true</code> enables anonymous data to be sent to NetApp for support purposes. The default election is <code>false</code> and indicates no support data will be sent to NetApp. | Boolean | <code>false</code> (this value is the default)  |
| <code>autoSupport.url</code>      | Either <code>enrolled</code> or <code>url</code> fields must be selected | This URL determines where the anonymous data will be sent.  | string  | <a href="https://support.netapp.com/asupprod/post/1.0/postAsup">https://support.netapp.com/asupprod/post/1.0/postAsup</a> |

## email

| Setting            | Guidance   | Type   | Example                        |
|--------------------|--|--------|--------------------------------|
| <code>email</code> | Change the email string to the default initial administrator address. Copy this email address for use in a <a href="#">later step</a> . This email address will be used as the username for the initial account to log in to the UI and will be notified of events in Astra Control. | string | <code>admin@example.com</code> |

firstName

| Setting   | Guidance   | Type   | Example |
|-----------|--|--------|---------|
| firstName | The first name of the default initial administrator associated with the Astra account. The name used here will be visible in a heading in the UI after your first login. | string | SRE     |

LastName

| Setting  | Guidance  | Type   | Example |
|----------|---|--------|---------|
| lastName | The last name of the default initial administrator associated with the Astra account. The name used here will be visible in a heading in the UI after your first login. | string | Admin   |

Your selections in this section define the container image registry that is hosting the Astra application images, Astra Control Center Operator, and Astra Control Center Helm repository.

| Setting              | Use   | Guidance   | Type   | Example                    |
|----------------------|---|--|--------|----------------------------|
| imageRegistry.name   | Required  | The name of the image registry where you pushed the images in the <a href="#">previous step</a> . Do not use <code>http://</code> or <code>https://</code> in the registry name. | string | example.registry.com/astra |
| imageRegistry.secret | Required if the string you entered for <code>imageRegistry.name</code> requires a secret.<br><br>IMPORTANT: If you are using a registry that does not require authorization, you must delete this <code>secret</code> line within <code>imageRegistry</code> or the installation will fail. | The name of the Kubernetes secret used to authenticate with the image registry.  | string | astra-registry-cred        |



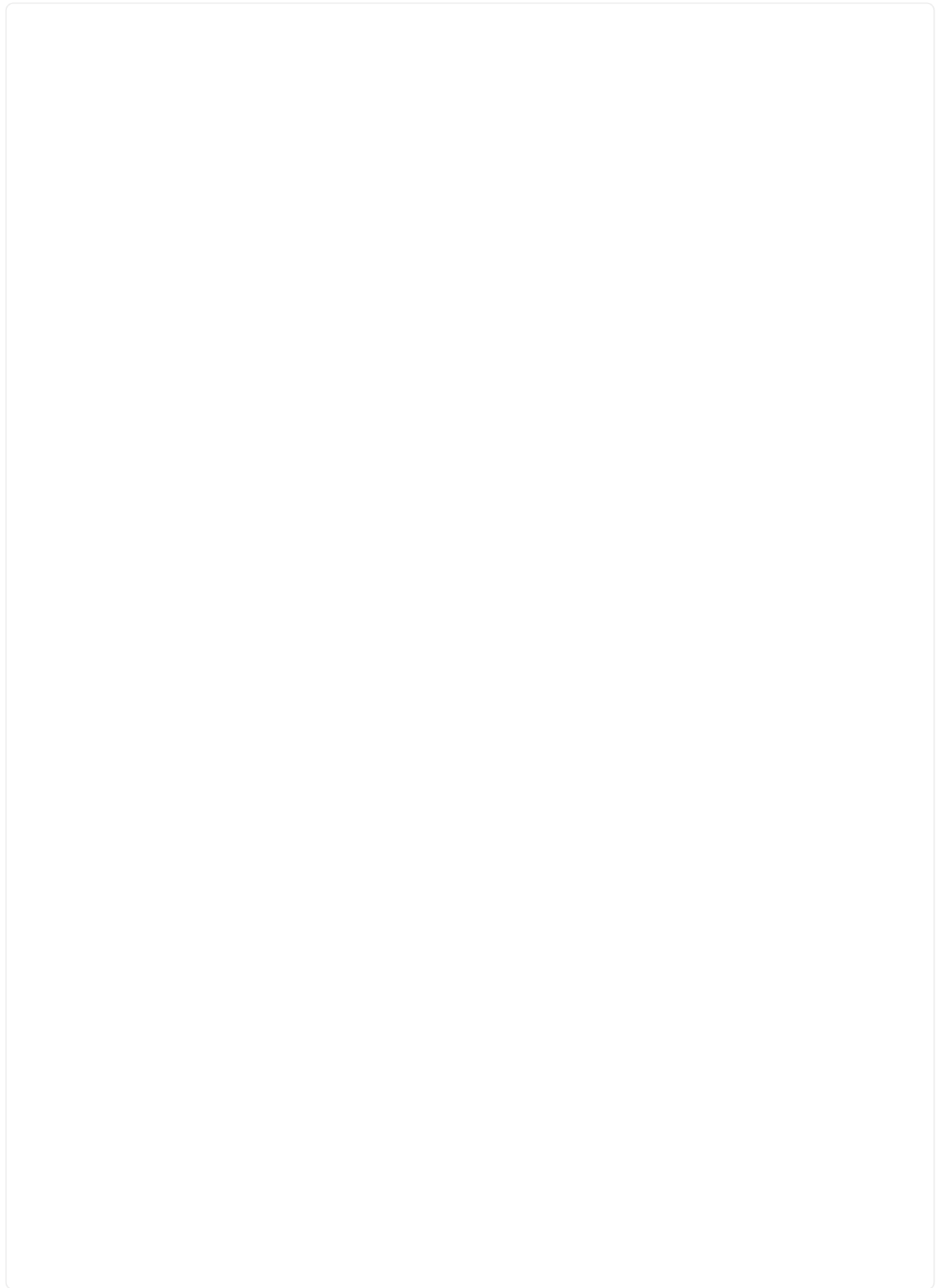
## storageClass

| Setting      | Guidance   | Type   | Example    |
|--------------|--|--------|------------|
| storageClass | <p>Change the storageClass value from ontap-gold to another Trident storageClass resource as required by your installation. Run the command <code>kubectl get sc</code> to determine your existing configured storage classes. One of the Trident-based storage classes must be entered in the manifest file (<code>astra-control-center-&lt;version&gt;.manifest</code>) and will be used for Astra PVs. If it is not set, the default storage class will be used.</p> <p>NOTE: If a default storage class is configured, ensure that it is the only storage class that has the default annotation.</p> | string | ontap-gold |

## volumeReclaimPolicy

| Setting             | Guidance   | Type   | Options   |
|---------------------|--|--------|---|
| volumeReclaimPolicy | <p>This sets the reclaim policy for Astra's PVs. Setting this policy to <code>Retain</code> retains persistent volumes after Astra is deleted. Setting this policy to <code>Delete</code> deletes persistent volumes after astra is deleted. If this value is not set, the PVs are retained.</p> | string | <ul style="list-style-type: none"><li>• Retain (This is the default value)</li><li>• Delete</li></ul> |

ingressType





| Setting     | Guidance  | Type   | Options   |
|-------------|---|--------|---|
| ingressType | <p>Use one of the following ingress types:</p> <p><b>Generic</b><br/> (ingressType:<br/> "Generic") (Default)<br/> Use this option when you have another ingress controller in use or would prefer to use your own ingress controller. After Astra Control Center is deployed, you will need to configure the <a href="#">ingress controller</a> to expose Astra Control Center with a URL.</p> <p><b>AccTraefik</b><br/> (ingressType:<br/> "AccTraefik")<br/> Use this option when you would prefer not to configure an ingress controller. This deploys the Astra Control Center traefik gateway as a Kubernetes LoadBalancer type service.</p> <p>Astra Control Center uses a service of the type "LoadBalancer" (svc/traefik in the Astra Control Center namespace), and requires that it be assigned an accessible external IP address. If load balancers are permitted in your environment and you don't already have one configured, you can use MetalLB or another external service load balancer to assign an external IP address to the service. In the internal DNS server configuration, you should point the chosen</p> | string | <ul style="list-style-type: none"> <li>• Generic (this is the default value)</li> <li>• AccTraefik</li> </ul> |

| Setting                           | Guidance   | Type   | Options   |
|-----------------------------------|--|--------|---|
| <code>astraResourcesScaler</code> | <p>Scaling options for AstraControlCenter Resource limits. By default, Astra Control Center deploys with resource requests set for most of the components within Astra. This configuration allows the Astra Control Center software stack to perform better in environments under increased application load and scale.</p> <p>However, in scenarios using smaller development or test clusters, the CR field <code>astraResourcesScaler</code> may be set to <code>Off</code>. This disables resource requests and allows for deployment on smaller clusters.</p> | string | <ul style="list-style-type: none"><li>• <code>Default</code> (This is the default value)</li><li>• <code>Off</code></li></ul> |

Your selections in this section determine how Astra Control Center should handle CRDs.

| Setting                               | Guidance  | Type    | Example                           |
|---------------------------------------|---|---------|-----------------------------------|
| <code>crds.externalCertManager</code> | <p>If you use an external cert manager, change <code>externalCertManager</code> to <code>true</code>. The default <code>false</code> causes Astra Control Center to install its own cert manager CRDs during installation.</p> <p>CRDs are cluster-wide objects and installing them might have an impact on other parts of the cluster. You can use this flag to signal to Astra Control Center that these CRDs will be installed and managed by the cluster administrator outside of Astra Control Center.</p> | Boolean | False (this value is the default) |
| <code>crds.externalTraefik</code>     | <p>By default, Astra Control Center will install required Traefik CRDs. CRDs are cluster-wide objects and installing them might have an impact on other parts of the cluster. You can use this flag to signal to Astra Control Center that these CRDs will be installed and managed by the cluster administrator outside of Astra Control Center.</p>   | Boolean | False (this value is the default) |

```
<strong>astra_control_center.yaml</strong>
```

```
apiVersion: astra.netapp.io/v1
kind: AstraControlCenter
metadata:
  name: astra
spec:
  accountName: "Example"
  astraVersion: "ASTRA_VERSION"
  astraAddress: "astra.example.com"
  autoSupport:
    enrolled: true
  email: "[admin@example.com]"
  firstName: "SRE"
  lastName: "Admin"
  imageRegistry:
    name: "[your_registry_path]"
    secret: "astra-registry-cred"
  storageClass: "ontap-gold"
  volumeReclaimPolicy: "Retain"
  ingressType: "Generic"
  astraResourcesScaler: "Default"
  additionalValues: {}
  crds:
    externalTraefik: false
    externalCertManager: false
```

## Complete Astra Control Center and operator installation

1. If you didn't already do so in a previous step, create the `netapp-acc` (or custom) namespace:

```
kubectl create ns [netapp-acc or custom namespace]
```

Sample response:

```
namespace/netapp-acc created
```

2. Install Astra Control Center in the `netapp-acc` (or your custom) namespace:

```
kubectl apply -f astra_control_center.yaml -n [netapp-acc or custom namespace]
```

Sample response:

```
astracontrolcenter.astra.netapp.io/astra created
```

## Verify system status

You can verify system status using `kubectl` commands. If you prefer to use OpenShift, you can use comparable `oc` commands for verification steps.

### Steps

1. Verify that all system components installed successfully.

```
kubectl get pods -n [netapp-acc or custom namespace]
```

Each pod should have a status of `Running`. It may take several minutes before the system pods are deployed.



## Sample response

| NAME   | READY | STATUS  |   |
|--|-------|---------|---|
| RESTARTS   | AGE   |         |   |
| acc-helm-repo-76d8d845c9-ggds2<br>14m                        | 1/1   | Running | 0 |
| activity-6cc67ff9f4-z48mr<br>(8m32s ago) 9m                  | 1/1   | Running | 2 |
| api-token-authentication-7s67v<br>8m56s                      | 1/1   | Running | 0 |
| api-token-authentication-bplb4<br>8m56s                      | 1/1   | Running | 0 |
| api-token-authentication-p2c9z<br>8m56s                      | 1/1   | Running | 0 |
| asup-6cdfbc6795-md8vn<br>9m14s                               | 1/1   | Running | 0 |
| authentication-9477567db-8hnc9<br>7m4s                       | 1/1   | Running | 0 |
| bucket-service-f4dbdfcd6-wqzkw<br>8m48s                      | 1/1   | Running | 0 |
| cert-manager-bb756c7c4-wm2cv<br>14m                          | 1/1   | Running | 0 |
| cert-manager-cainjector-c9bb86786-8wrf5<br>14m               | 1/1   | Running | 0 |
| cert-manager-webhook-dd465db99-j2w4x<br>14m                  | 1/1   | Running | 0 |
| certificates-68dff9cdd6-kcvml<br>(8m43s ago) 9m2s            | 1/1   | Running | 2 |
| certificates-68dff9cdd6-rsnsb<br>9m2s                        | 1/1   | Running | 0 |
| cloud-extension-69d48c956c-2s8dt<br>(8m43s ago) 9m24s        | 1/1   | Running | 3 |
| cloud-insights-service-7c4f48b978-7gvlh<br>(8m50s ago) 9m28s | 1/1   | Running | 3 |
| composite-compute-7d9ff5f68-nxbhl<br>8m51s                   | 1/1   | Running | 0 |
| composite-volume-57b4756d64-nl66d<br>9m13s                   | 1/1   | Running | 0 |
| credentials-6dbc55f89f-qpzff<br>11m                          | 1/1   | Running | 0 |
| entitlement-67bfb6d7-gl6kp<br>(8m33s ago) 9m38s              | 1/1   | Running | 4 |
| features-856cc4dccc-mxbdb<br>9m20s                           | 1/1   | Running | 0 |
| fluent-bit-ds-4rtsp<br>6m54s                                 | 1/1   | Running | 0 |

|                                    |     |         |   |
|------------------------------------|-----|---------|---|
| fluent-bit-ds-9rql1                | 1/1 | Running | 0 |
| 6m54s                              |     |         |   |
| fluent-bit-ds-w5mp7                | 1/1 | Running | 0 |
| 6m54s                              |     |         |   |
| graphql-server-7c7cc49776-jz2kn    | 1/1 | Running | 0 |
| 2m29s                              |     |         |   |
| identity-87c59c975-9jpnf           | 1/1 | Running | 0 |
| 9m6s                               |     |         |   |
| influxdb2-0                        | 1/1 | Running | 0 |
| 13m                                |     |         |   |
| keycloak-operator-84ff6d59d4-qcnmc | 1/1 | Running | 0 |
| 7m1s                               |     |         |   |
| krakend-cbf6c7df9-mdtzv            | 1/1 | Running | 0 |
| 2m30s                              |     |         |   |
| license-5b888b78bf-plj6j           | 1/1 | Running | 0 |
| 9m32s                              |     |         |   |
| login-ui-846b4664dd-fz8hv          | 1/1 | Running | 0 |
| 2m24s                              |     |         |   |
| loki-0                             | 1/1 | Running | 0 |
| 13m                                |     |         |   |
| metrics-facade-779cc9774-n26rw     | 1/1 | Running | 0 |
| 9m18s                              |     |         |   |
| monitoring-operator-974db78f-pkspq | 2/2 | Running | 0 |
| 6m58s                              |     |         |   |
| nats-0                             | 1/1 | Running | 0 |
| 13m                                |     |         |   |
| nats-1                             | 1/1 | Running | 0 |
| 13m                                |     |         |   |
| nats-2                             | 1/1 | Running | 0 |
| 13m                                |     |         |   |
| nautilus-7bdc7ddc54-49tfn          | 1/1 | Running | 0 |
| 7m50s                              |     |         |   |
| nautilus-7bdc7ddc54-cwc79          | 1/1 | Running | 0 |
| 9m36s                              |     |         |   |
| openapi-5584ff9f46-gbrdj           | 1/1 | Running | 0 |
| 9m17s                              |     |         |   |
| openapi-5584ff9f46-z9mzk           | 1/1 | Running | 0 |
| 9m17s                              |     |         |   |
| packages-bfc58cc98-lpxq9           | 1/1 | Running | 0 |
| 8m58s                              |     |         |   |
| polaris-consul-consul-server-0     | 1/1 | Running | 0 |
| 13m                                |     |         |   |
| polaris-consul-consul-server-1     | 1/1 | Running | 0 |
| 13m                                |     |         |   |
| polaris-consul-consul-server-2     | 1/1 | Running | 0 |
| 13m                                |     |         |   |

|   |     |         |   |
|---|-----|---------|---|
| polaris-keycloak-0<br>(6m15s ago) 6m56s                 | 1/1 | Running | 3 |
| polaris-keycloak-1<br>4m22s                             | 1/1 | Running | 0 |
| polaris-keycloak-2<br>3m41s                             | 1/1 | Running | 0 |
| polaris-keycloak-db-0<br>6m56s                          | 1/1 | Running | 0 |
| polaris-keycloak-db-1<br>4m23s                          | 1/1 | Running | 0 |
| polaris-keycloak-db-2<br>3m36s                          | 1/1 | Running | 0 |
| polaris-mongodb-0<br>13m                                | 2/2 | Running | 0 |
| polaris-mongodb-1<br>13m                                | 2/2 | Running | 0 |
| polaris-mongodb-2<br>12m                                | 2/2 | Running | 0 |
| polaris-ui-5ccff47897-8rzgh<br>2m33s                    | 1/1 | Running | 0 |
| polaris-vault-0<br>13m                                  | 1/1 | Running | 0 |
| polaris-vault-1<br>13m                                  | 1/1 | Running | 0 |
| polaris-vault-2<br>13m                                  | 1/1 | Running | 0 |
| public-metrics-6cb7bfc49b-p54xm<br>(8m29s ago) 9m31s    | 1/1 | Running | 1 |
| storage-backend-metrics-5c77994586-kjn48<br>8m52s       | 1/1 | Running | 0 |
| storage-provider-769fdc858c-62w54<br>8m54s              | 1/1 | Running | 0 |
| task-service-9ffc484c5-kx9f4<br>(8m44s ago) 9m34s       | 1/1 | Running | 3 |
| telegraf-ds-bphb9<br>6m54s                              | 1/1 | Running | 0 |
| telegraf-ds-rtsm2<br>6m54s                              | 1/1 | Running | 0 |
| telegraf-ds-s9h5h<br>6m54s                              | 1/1 | Running | 0 |
| telegraf-rs-lbpv7<br>6m54s                              | 1/1 | Running | 0 |
| telemetry-service-57cfb998db-zjx78<br>(8m40s ago) 9m26s | 1/1 | Running | 1 |
| tenancy-5d5dfbcf9f-vmbxh<br>9m5s                        | 1/1 | Running | 0 |

|                                   |     |         |   |
|-----------------------------------|-----|---------|---|
| traefik-7b87c4c474-jmcp2          | 1/1 | Running | 0 |
| 2m24s                             |     |         |   |
| traefik-7b87c4c474-t9k8x          | 1/1 | Running | 0 |
| 2m24s                             |     |         |   |
| trident-svc-c78f5b6bd-nwdsq       | 1/1 | Running | 0 |
| 9m22s                             |     |         |   |
| vault-controller-55bbc96668-c6425 | 1/1 | Running | 0 |
| 11m                               |     |         |   |
| vault-controller-55bbc96668-lq9n9 | 1/1 | Running | 0 |
| 11m                               |     |         |   |
| vault-controller-55bbc96668-rfkkg | 1/1 | Running | 0 |
| 11m                               |     |         |   |

2. (Optional) To ensure the installation is completed, you can watch the `acc-operator` logs using the following command.

```
kubectl logs deploy/acc-operator-controller-manager -n netapp-acc-operator -c manager -f
```



`accHost` cluster registration is one of the last operations, and if it fails it will not cause deployment to fail. In the event of a cluster registration failure indicated in the logs, you can attempt registration again through the [Add cluster workflow in the UI](#) or API.

3. When all the pods are running, verify that the installation was successful (`READY` is `True`) and get the initial setup password you will use when you log in to Astra Control Center:

```
kubectl get AstraControlCenter -n [netapp-acc or custom namespace]
```

Response:

| NAME  | UUID                                 | VERSION    | ADDRESS        |
|-------|--------------------------------------|------------|----------------|
| READY |                                      |            |                |
| astra | 9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f | 22.11.0-82 | 10.111.111.111 |
| True  |                                      |            |                |



Copy the UUID value. The password is `ACC-` followed by the UUID value (`ACC-[UUID]` or, in this example, `ACC-9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f`).

## Set up ingress for load balancing

You can set up a Kubernetes ingress controller that manages external access to services. These procedures give setup examples for an ingress controller if you used the default of `ingressType: "Generic"` in the

Astra Control Center custom resource (`astra_control_center.yaml`). You do not need to use this procedure if you specified `ingressType: "AccTraefik"` in the Astra Control Center custom resource (`astra_control_center.yaml`).

After Astra Control Center is deployed, you will need to configure the ingress controller to expose Astra Control Center with a URL.

Setup steps differ depending on the type of ingress controller you use. Astra Control Center supports many ingress controller types. These setup procedures provide example steps for the following ingress controller types:

- Istio ingress
- Nginx ingress controller
- OpenShift ingress controller

### What you'll need

- The required [ingress controller](#) should already be deployed.
- The [ingress class](#) corresponding to the ingress controller should already be created.

### Steps for Istio ingress

1. Configure Istio ingress.



This procedure assumes that Istio is deployed using the "default" configuration profile.

2. Gather or create the desired certificate and private key file for the Ingress Gateway.

You can use a CA-signed or self-signed certificate. The common name must be the Astra address (FQDN).

Sample command:

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout tls.key -out  
tls.crt
```

3. Create a secret `tls` secret name of type `kubernetes.io/tls` for a TLS private key and certificate in the `istio-system` namespace as described in [TLS secrets](#).

Sample command:

```
kubectl create secret tls [tls secret name] --key="tls.key"  
--cert="tls.crt" -n istio-system
```



The name of the secret should match the `spec.tls.secretName` provided in `istio-ingress.yaml` file.

4. Deploy an ingress resource in the `netapp-acc` (or custom-named) namespace using the `v1` resource type for a schema (`istio-Ingress.yaml` is used in this example):

```

apiVersion: networking.k8s.io/v1
kind: IngressClass
metadata:
  name: istio
spec:
  controller: istio.io/ingress-controller
---
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: ingress
  namespace: [netapp-acc or custom namespace]
spec:
  ingressClassName: istio
  tls:
  - hosts:
    - <ACC address>
    secretName: [tls secret name]
  rules:
  - host: [ACC address]
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: traefik
            port:
              number: 80

```

##### 5. Apply the changes:

```
kubectl apply -f istio-Ingress.yaml
```

##### 6. Check the status of the ingress:

```
kubectl get ingress -n [netapp-acc or custom namespace]
```

##### Response:

| NAME    | CLASS | HOSTS             | ADDRESS        | PORTS   | AGE |
|---------|-------|-------------------|----------------|---------|-----|
| ingress | istio | astra.example.com | 172.16.103.248 | 80, 443 | 1h  |

## 7. Finish Astra Control Center installation.

### Steps for Nginx ingress controller

1. Create a secret of type `kubernetes.io/tls` for a TLS private key and certificate in `netapp-acc` (or custom-named) namespace as described in [TLS secrets](#).
2. Deploy an ingress resource in `netapp-acc` (or custom-named) namespace using the v1 resource type for a schema (`nginx-Ingress.yaml` is used in this example):

```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: netapp-acc-ingress
  namespace: [netapp-acc or custom namespace]
spec:
  ingressClassName: [class name for nginx controller]
  tls:
  - hosts:
    - <ACC address>
    secretName: [tls secret name]
  rules:
  - host: <ACC address>
    http:
      paths:
      - path:
        backend:
          service:
            name: traefik
            port:
              number: 80
        pathType: ImplementationSpecific
```

3. Apply the changes:

```
kubectl apply -f nginx-Ingress.yaml
```



NetApp recommends installing the nginx controller as a deployment rather than a daemonSet.

### Steps for OpenShift ingress controller

1. Procure your certificate and get the key, certificate, and CA files ready for use by the OpenShift route.
2. Create the OpenShift route:

```
oc create route edge --service=traefik --port=web -n [netapp-acc or
custom namespace] --insecure-policy=Redirect --hostname=<ACC address>
--cert=cert.pem --key=key.pem
```

## Log in to the Astra Control Center UI

After installing Astra Control Center, you will change the password for the default administrator and log in to the Astra Control Center UI dashboard.

### Steps

1. In a browser, enter the FQDN (including the `https://` prefix) you used in the `astraAddress` in the `astra_control_center.yaml` CR when [you installed Astra Control Center](#).
2. Accept the self-signed certificates if prompted.



You can create a custom certificate after login.

3. At the Astra Control Center login page, enter the value you used for email in `astra_control_center.yaml` CR when [you installed Astra Control Center](#), followed by the initial setup password (ACC-[UUID]).



If you enter an incorrect password three times, the admin account will be locked for 15 minutes.

4. Select **Login**.
5. Change the password when prompted.



If this is your first login and you forget the password and no other administrative user accounts have yet been created, contact [NetApp Support](#) for password recovery assistance.

6. (Optional) Remove the existing self-signed TLS certificate and replace it with a [custom TLS certificate signed by a Certificate Authority \(CA\)](#).

## Troubleshoot the installation

If any of the services are in `Error` status, you can inspect the logs. Look for API response codes in the 400 to 500 range. Those indicate the place where a failure happened.

### Steps

1. To inspect the Astra Control Center operator logs, enter the following:

```
kubectl logs deploy/acc-operator-controller-manager -n netapp-acc-
operator -c manager -f
```



## What's next

- (Optional) Depending on your environment, complete post-installation [configuration steps](#).
- Complete the deployment by performing [setup tasks](#).

## Install Astra Control Center using OpenShift OperatorHub

If you use Red Hat OpenShift, you can install Astra Control Center using the Red Hat certified operator. Use this procedure to install Astra Control Center from the [Red Hat Ecosystem Catalog](#) or using the Red Hat OpenShift Container Platform.

After you complete this procedure, you must return to the installation procedure to complete the [remaining steps](#) to verify installation success and log on.

### What you'll need

- **Environmental prerequisites met:** [Before you begin installation, prepare your environment for Astra Control Center deployment](#).

- **Healthy cluster operators and API services:**

- From your OpenShift cluster, ensure all cluster operators are in a healthy state:

```
oc get clusteroperators
```

- From your OpenShift cluster, ensure all API services are in a healthy state:

```
oc get apiservices
```

- **FQDN address:** Obtain an FQDN address for Astra Control Center in your data center.
- **Openshift Permissions:** Obtain the necessary permissions and access to the Red Hat OpenShift Container Platform to perform the installation steps described.
- **cert manager configured:** If a cert manager already exists in the cluster, you need to perform some [prerequisite steps](#) so that Astra Control Center does not install its own cert manager. By default, Astra Control Center installs its own cert manager during installation.
- **Kubernetes ingress controller:** If you have a Kubernetes ingress controller that manages external access to services, such as load balancing in a cluster, you need to set it up for use with Astra Control Center:
  - a. Create the operator namespace:

```
oc create namespace netapp-acc-operator
```

- b. [Complete setup](#) for your ingress controller type.

### Steps

- [Download and extract Astra Control Center](#)
- [Install the NetApp Astra kubectl plugin](#)

- [Add the images to your local registry](#)
- [Find the operator install page](#)
- [Install the operator](#)
- [Install Astra Control Center](#)

## Download and extract Astra Control Center

1. Go to the [Astra Control Center Evaluation downloads page](#) on the NetApp Support Site.
2. Download the bundle containing Astra Control Center (`astra-control-center-[version].tar.gz`).
3. (Recommended but optional) Download the certificates and signatures bundle for Astra Control Center (`astra-control-center-certs-[version].tar.gz`) to verify the signature of the bundle:

```
tar -vxzf astra-control-center-certs-[version].tar.gz
```

```
openssl dgst -sha256 -verify certs/AstraControlCenter-public.pub  
-signature certs/astra-control-center-[version].tar.gz.sig astra-  
control-center-[version].tar.gz
```

The output will show `Verified OK` after successful verification.

4. Extract the images from the Astra Control Center bundle:

```
tar -vxzf astra-control-center-[version].tar.gz
```

## Install the NetApp Astra kubectl plugin

The NetApp Astra kubectl command line plugin saves time when performing common tasks associated with deploying and upgrading Astra Control Center.

### What you'll need

NetApp provides plugin binaries for different CPU architectures and operating systems. You need to know which CPU and operating system you have before you perform this task.

### Steps

1. List the available NetApp Astra kubectl plugin binaries, and note the name of the file you need for your operating system and CPU architecture:



The kubectl plugin library is part of the tar bundle and is extracted into the folder `kubectl-astra`.

```
ls kubectl-astra/
```

2. Move the correct binary into the current path and rename it to `kubectl-astra`:

```
cp kubectl-astra/<binary-name> /usr/local/bin/kubectl-astra
```

## Add the images to your local registry

1. Complete the appropriate step sequence for your container engine:

## Docker

- a. Change to the root directory of the tarball. You should see this file and directory:

```
acc.manifest.bundle.yaml
acc/
```

- b. Push the package images in the Astra Control Center image directory to your local registry. Make the following substitutions before running the `push-images` command:

- Replace `<BUNDLE_FILE>` with the name of the Astra Control bundle file (`acc.manifest.bundle.yaml`).
- Replace `<MY_FULL_REGISTRY_PATH>` with the URL of the Docker repository; for example, `"https://<docker-registry>"`.
- Replace `<MY_REGISTRY_USER>` with the user name.
- Replace `<MY_REGISTRY_TOKEN>` with an authorized token for the registry.

```
kubectrl astra packages push-images -m <BUNDLE_FILE> -r
<MY_FULL_REGISTRY_PATH> -u <MY_REGISTRY_USER> -p
<MY_REGISTRY_TOKEN>
```

## Podman

- a. Change to the root directory of the tarball. You should see this file and directory:

```
acc.manifest.bundle.yaml
acc/
```

- b. Log in to your registry:

```
podman login <YOUR_REGISTRY>
```

- c. Prepare and run one of the following scripts that is customized for the version of Podman you use. Substitute `<MY_FULL_REGISTRY_PATH>` with the URL of your repository that includes any sub-directories.

```
<strong>Podman 4</strong>
```

```

export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=22.11.0-82
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image: //'')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/${
PACKAGEVERSION}/${astraImageNoPath}
done

```

<strong>Podman 3</strong>

```

export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=22.11.0-82
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image: //'')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/${
PACKAGEVERSION}/${astraImageNoPath}
done

```



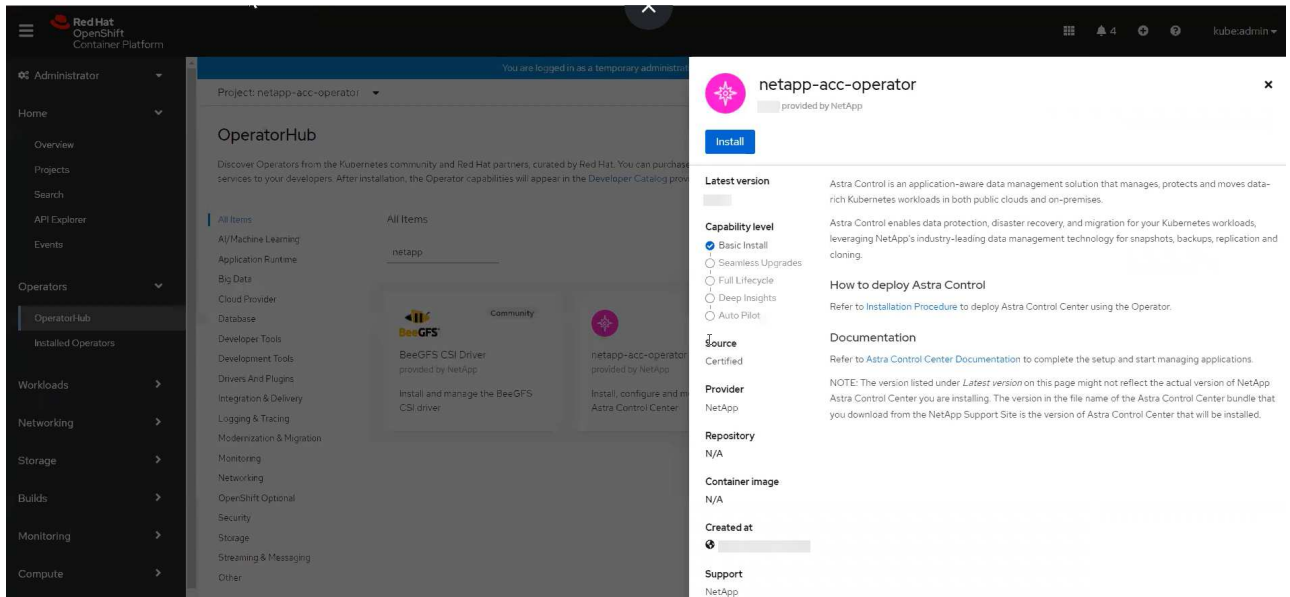
The image path the script creates should resemble the following, depending on your registry configuration: <https://netappdownloads.jfrog.io/docker-astra-control-prod/netapp/astra/acc/22.11.0-82/image:version>

## Find the operator install page

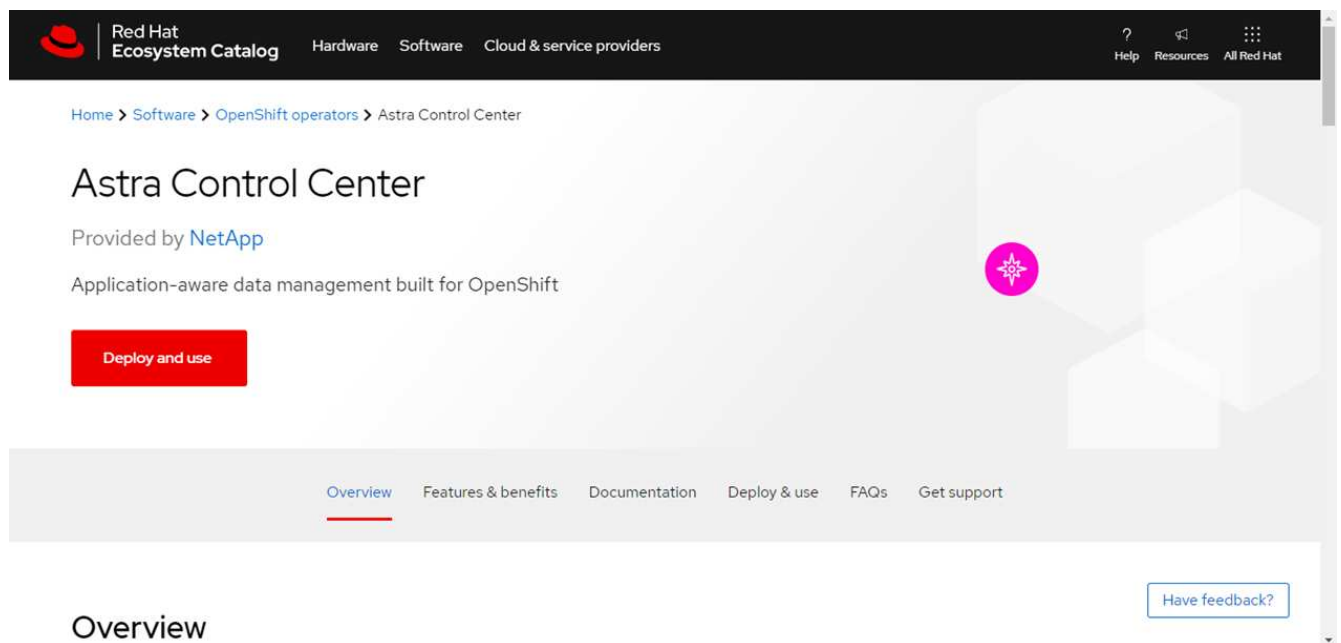
1. Complete one of the following procedures to access the operator install page:

- From Red Hat Openshift web console:
  - a. Log in to the OpenShift Container Platform UI.
  - b. From the side menu, select **Operators > OperatorHub**.

c. Search for and select the NetApp Astra Control Center operator.



- From Red Hat Ecosystem Catalog:
  - a. Select the NetApp Astra Control Center [operator](#).
  - b. Select **Deploy and Use**.



## Overview

## Install the operator

1. Complete the **Install Operator** page and install the operator:



The operator will be available in all cluster namespaces.

- a. Select the operator namespace or `netapp-acc-operator` namespace will be created automatically as part of the operator installation.

- b. Select a manual or automatic approval strategy.



Manual approval is recommended. You should only have a single operator instance running per cluster.

- c. Select **Install**.



If you selected a manual approval strategy, you will be prompted to approve the manual install plan for this operator.

2. From the console, go to the OperatorHub menu and confirm that the operator installed successfully.

## Install Astra Control Center

1. From the console within the **Astra Control Center** tab of the Astra Control Center operator, select **Create AstraControlCenter**.

The screenshot shows the Astra Control Center console interface. At the top, there's a header with 'Project: netapp-acc-operator' and a dropdown arrow. Below this is a breadcrumb trail 'Installed Operators > Operator details'. The main content area has a header for 'netapp-acc-operator 22.8.1 provided by NetApp' and an 'Actions' dropdown. Below the header is a tabbed interface with 'Details', 'YAML', 'Subscription', 'Events', and 'Astra Control Center'. The 'Astra Control Center' tab is active, showing a section titled 'AstraControlCenters' with a 'Create AstraControlCenter' button. Below this, there's a message 'No operands found' and a note 'Operands are declarative components used to define the behavior of the application.'

2. Complete the `Create AstraControlCenter` form field:
  - a. Keep or adjust the Astra Control Center name.
  - b. Add labels for the Astra Control Center.
  - c. Enable or disable Auto Support. Retaining Auto Support functionality is recommended.
  - d. Enter the Astra Control Center FQDN or IP address. Do not enter `http://` or `https://` in the address field.
  - e. Enter the Astra Control Center version; for example, 22.04.1.
  - f. Enter an account name, email address, and admin last name.
  - g. Choose a volume reclaim policy of `Retain`, `Recycle`, or `Delete`. The default value is `Retain`.
  - h. Select the ingress type:

- **Generic** (`ingressType: "Generic"`) (Default)

Use this option when you have another ingress controller in use or would prefer to use your own ingress controller. After Astra Control Center is deployed, you will need to configure the [ingress controller](#) to expose Astra Control Center with a URL.

- **AccTraefik** (`ingressType: "AccTraefik"`)

Use this option when you would prefer not to configure an ingress controller. This deploys the Astra Control Center `traefik` gateway as a Kubernetes "LoadBalancer" type service.

Astra Control Center uses a service of the type "LoadBalancer" (`svc/traefik` in the Astra Control Center namespace), and requires that it be assigned an accessible external IP address. If load balancers are permitted in your environment and you don't already have one configured, you can use MetalLB or another external service load balancer to assign an external IP address to the service. In the internal DNS server configuration, you should point the chosen DNS name for Astra Control Center to the load-balanced IP address.



For details about the service type of "LoadBalancer" and ingress, see [Requirements](#).

- i. In **Image Registry**, enter your local container image registry path. Do not enter `http://` or `https://` in the address field.
- j. If you use an image registry that requires authentication, enter the image secret.



If you use a registry that requires authentication, [create a secret on the cluster](#).

- k. Enter the admin first name.
- l. Configure resources scaling.
- m. Provide the default storage class.



If a default storage class is configured, ensure that it is the only storage class that has the default annotation.

- n. Define CRD handling preferences.
3. Select the YAML view to review the settings you have selected.
4. Select **Create**.

## Create a registry secret

If you use a registry that requires authentication, create a secret on the OpenShift cluster and enter the secret name in the `Create AstraControlCenter` form field.

1. Create a namespace for the Astra Control Center operator:

```
oc create ns [netapp-acc-operator or custom namespace]
```

2. Create a secret in this namespace:

```
oc create secret docker-registry astra-registry-cred n [netapp-acc-operator or custom namespace] --docker-server=[your_registry_path] --docker-username=[username] --docker-password=[token]
```



Astra Control supports Docker registry secrets only.



3. Complete the remaining fields in [the Create AstraControlCenter form field](#).

## What's next

Complete the [remaining steps](#) to verify that Astra Control Center installed successfully, set up an ingress controller (optional), and log in to the UI. Additionally, you will need to perform [setup tasks](#) after completing installation.

## Install Astra Control Center with a Cloud Volumes ONTAP storage backend

With Astra Control Center, you can manage your apps in a hybrid cloud environment with self-managed Kubernetes clusters and Cloud Volumes ONTAP instances. You can deploy Astra Control Center in your on-premise Kubernetes clusters or in one of the self-managed Kubernetes clusters in the cloud environment.

With one of these deployments, you can perform app data management operations using Cloud Volumes ONTAP as a storage backend. You can also configure an S3 bucket as the backup target.

To install Astra Control Center in Amazon Web Services (AWS), Google Cloud Platform (GCP) and Microsoft Azure with a Cloud Volumes ONTAP storage backend, perform the following steps depending on your cloud environment.

- [Deploy Astra Control Center in Amazon Web Services](#)
- [Deploy Astra Control Center in Google Cloud Platform](#)
- [Deploy Astra Control Center in Microsoft Azure](#)

You can manage your apps in distributions with self-managed Kubernetes clusters, such with OpenShift Container Platform (OCP). Only self-managed OCP clusters are validated for deploying Astra Control Center.

## Deploy Astra Control Center in Amazon Web Services

You can deploy Astra Control Center on a self-managed Kubernetes cluster hosted on an Amazon Web Services (AWS) public cloud.

### What you'll need for AWS

Before you deploy Astra Control Center in AWS, you will need the following items:

- Astra Control Center license. See [Astra Control Center licensing requirements](#).
- [Meet Astra Control Center requirements](#).
- NetApp Cloud Central account
- If using OCP, Red Hat OpenShift Container Platform (OCP) permissions (on namespace level to create pods)
- AWS credentials, Access ID and Secret Key with permissions that enable you to create buckets and connectors
- AWS account Elastic Container Registry (ECR) access and login
- AWS hosted zone and Route 53 entry required to access the Astra Control UI

## Operational environment requirements for AWS


Astra Control Center requires the following operational environment for AWS:

- Red Hat OpenShift Container Platform 4.8



Ensure that the operating environment you choose to host Astra Control Center meets the basic resource requirements outlined in the environment's official documentation.

Astra Control Center requires the following resources in addition to the environment's resource requirements:

| Component   | Requirement   |
|---|---|
| <b>Backend NetApp Cloud Volumes ONTAP storage capacity</b>  | At least 300GB available  |
| <b>Worker nodes (AWS EC2 requirement)</b>   | At least 3 worker nodes total, with 4 vCPU cores and 12GB RAM each  |
| <b>Load balancer</b>  | Service type "LoadBalancer" available for ingress traffic to be sent to services in the operational environment cluster   |
| <b>FQDN</b>   | A method for pointing the FQDN of Astra Control Center to the load balanced IP address  |
| <b>Astra Trident (installed as part of the Kubernetes cluster discovery in NetApp BlueXP, formerly Cloud Manager)</b> | Astra Trident 21.04 or newer installed and configured and NetApp ONTAP version 9.5 or newer as a storage backend  |
| <b>Image registry</b>   | <p>You must have an existing private registry, such as AWS Elastic Container Registry, to which you can push Astra Control Center build images. You need to provide the URL of the image registry where you will upload the images.</p> <div><p>The Astra Control Center hosted cluster and the managed cluster must have access to the same image registry to be able to back up and restore apps using the Restic-based image.</p></div> |

| Component                                  | Requirement   |
|--|---|
| <b>Astra Trident / ONTAP configuration</b> | <p>Astra Control Center requires that a storage class be created and set as the default storage class. Astra Control Center supports the following ONTAP Kubernetes storage classes that are created when you import your Kubernetes cluster into NetApp BlueXP (formerly Cloud Manager). These are provided by Astra Trident:</p> <ul style="list-style-type: none"> <li>• <code>vsaworkingenvironment-&lt;&gt;-ha-nas</code><br/><code>csi.trident.netapp.io</code></li> <li>• <code>vsaworkingenvironment-&lt;&gt;-ha-san</code><br/><code>csi.trident.netapp.io</code></li> <li>• <code>vsaworkingenvironment-&lt;&gt;-single-nas</code><br/><code>csi.trident.netapp.io</code></li> <li>• <code>vsaworkingenvironment-&lt;&gt;-single-san</code><br/><code>csi.trident.netapp.io</code></li> </ul> |



These requirements assume that Astra Control Center is the only application running in the operational environment. If the environment is running additional applications, adjust these minimum requirements accordingly.



The AWS registry token expires in 12 hours, after which you will have to renew the Docker image registry secret.

## Overview of deployment for AWS

Here is an overview of the process to install Astra Control Center for AWS with Cloud Volumes ONTAP as a storage backend.

Each of these steps is explained in more detail below.

1. [Ensure that you have sufficient IAM permissions.](#)
2. [Install a RedHat OpenShift cluster on AWS.](#)
3. [Configure AWS.](#)
4. [Configure NetApp BlueXP for AWS.](#)
5. [Install Astra Control Center for AWS.](#)

### Ensure that you have sufficient IAM permissions

Ensure that you have sufficient IAM roles and permissions that enable you to install a RedHat OpenShift cluster and a NetApp BlueXP (formerly Cloud Manager) Connector.

See [Initial AWS credentials](#).

### Install a RedHat OpenShift cluster on AWS

Install a RedHat OpenShift Container Platform cluster on AWS.

For installation instructions, see [Installing a cluster on AWS in OpenShift Container Platform](#).

## Configure AWS

Next, configure AWS to create a virtual network, set up EC2 compute instances, create an AWS S3 bucket, create an Elastic Container Register (ECR) to host the Astra Control Center images, and push the images to this registry.

Follow the AWS documentation to complete the following steps. See [AWS installation documentation](#).

1. Create an AWS virtual network.
2. Review the EC2 compute instances. This can be a bare metal server or VMs in AWS.
3. If the instance type does not already match the Astra minimum resource requirements for master and worker nodes, change the instance type in AWS to meet the Astra requirements. See [Astra Control Center requirements](#).
4. Create at least one AWS S3 bucket to store your backups.
5. Create an AWS Elastic Container Registry (ECR) to host all the ACC images.



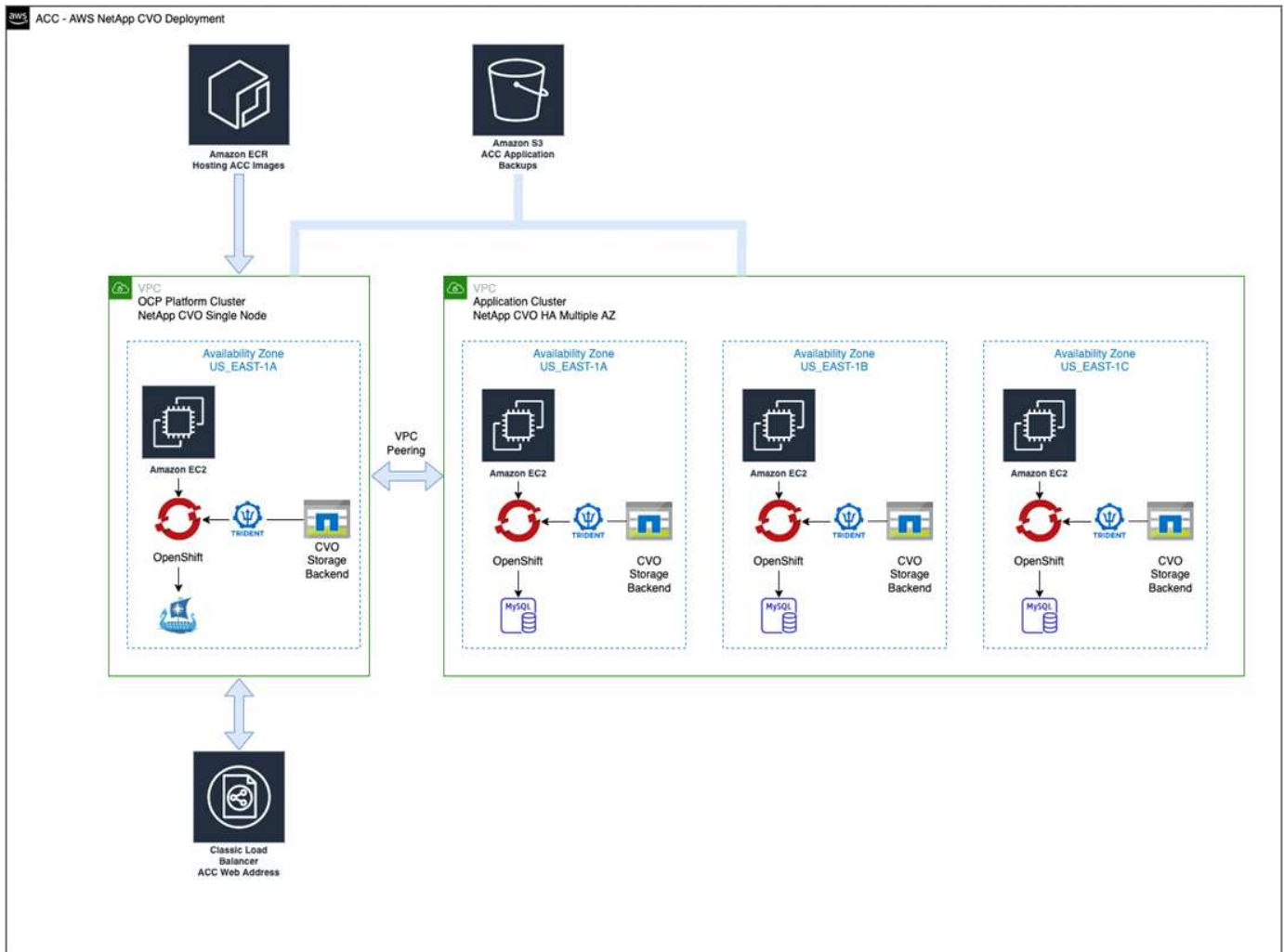
If you do not create the ECR, Astra Control Center cannot access monitoring data from a cluster containing Cloud Volumes ONTAP with an AWS backend. The issue is caused when the cluster you try to discover and manage using Astra Control Center does not have AWS ECR access.

6. Push the ACC images to your defined registry.



The AWS Elastic Container Registry (ECR) token expires after 12 hours and causes cross-cluster clone operations to fail. This issue occurs when managing a storage backend from Cloud Volumes ONTAP configured for AWS. To correct this issue, authenticate with the ECR again and generate a new secret for clone operations to resume successfully.

Here's an example of an AWS deployment:



## Configure NetApp BlueXP for AWS

Using NetApp BlueXP (formerly Cloud Manager), create a workspace, add a connector to AWS, create a working environment, and import the cluster.

Follow the BlueXP documentation to complete the following steps. See the following:

- [Getting started with Cloud Volumes ONTAP in AWS.](#)
- [Create a connector in AWS using BlueXP](#)

### Steps

1. Add your credentials to BlueXP.
2. Create a workspace.
3. Add a connector for AWS. Choose AWS as the Provider.
4. Create a working environment for your cloud environment.
  - a. Location: "Amazon Web Services (AWS)"
  - b. Type: "Cloud Volumes ONTAP HA"
5. Import the OpenShift cluster. The cluster will connect to the working environment you just created.
  - a. View the NetApp cluster details by selecting **K8s** > **Cluster list** > **Cluster Details**.

- b. In the upper right corner, note the Trident version.
- c. Note the Cloud Volumes ONTAP cluster storage classes showing NetApp as the provisioner.

This imports your Red Hat OpenShift cluster and assigns it a default storage class. You select the storage class.

Trident is automatically installed as part of the import and discovery process.

- 6. Note all the persistent volumes and volumes in this Cloud Volumes ONTAP deployment.



Cloud Volumes ONTAP can operate as a single node or in High Availability. If HA is enabled, note the HA status and node deployment status running in AWS.

## Install Astra Control Center for AWS

Follow the standard [Astra Control Center installation instructions](#).



AWS uses the Generic S3 bucket type.

## Deploy Astra Control Center in Google Cloud Platform

You can deploy Astra Control Center on a self-managed Kubernetes cluster hosted on a Google Cloud Platform (GCP) public cloud.

### What you'll need for GCP

Before you deploy Astra Control Center in GCP, you will need the following items:

- Astra Control Center license. See [Astra Control Center licensing requirements](#).
- [Meet Astra Control Center requirements](#).
- NetApp Cloud Central account
- If using OCP, Red Hat OpenShift Container Platform (OCP) 4.10
- If using OCP, Red Hat OpenShift Container Platform (OCP) permissions (on namespace level to create pods)
- GCP Service Account with permissions that enable you to create buckets and connectors


### Operational environment requirements for GCP



Ensure that the operating environment you choose to host Astra Control Center meets the basic resource requirements outlined in the environment's official documentation.

Astra Control Center requires the following resources in addition to the environment's resource requirements:

| Component  | Requirement  |
|--|--|
| <b>Backend NetApp Cloud Volumes ONTAP storage capacity</b> | At least 300GB available   |
| <b>Worker nodes (GCP compute requirement)</b>              | At least 3 worker nodes total, with 4 vCPU cores and 12GB RAM each |

| Component  | Requirement  |
|--|--|
| Load balancer  | Service type "LoadBalancer" available for ingress traffic to be sent to services in the operational environment cluster  |
| FQDN (GCP DNS zone)  | A method for pointing the FQDN of Astra Control Center to the load balanced IP address   |
| Astra Trident (installed as part of the Kubernetes cluster discovery in NetApp BlueXP, formerly Cloud Manager) | Astra Trident 21.04 or newer installed and configured and NetApp ONTAP version 9.5 or newer as a storage backend   |
| Image registry   | <p>You must have an existing private registry, such as Google Container Registry, to which you can push Astra Control Center build images. You need to provide the URL of the image registry where you will upload the images.</p> <div>  <p>You need to enable anonymous access to pull Restic images for backups.</p> </div>  |
| Astra Trident / ONTAP configuration  | <p>Astra Control Center requires that a storage class be created and set as the default storage class. Astra Control Center supports the following ONTAP Kubernetes storage classes that are created when you import your Kubernetes cluster into NetApp BlueXP. These are provided by Astra Trident:</p> <ul style="list-style-type: none"> <li>• <code>vsaworkingenvironment-&lt;&gt;-ha-nas</code><br/><code>csi.trident.netapp.io</code></li> <li>• <code>vsaworkingenvironment-&lt;&gt;-ha-san</code><br/><code>csi.trident.netapp.io</code></li> <li>• <code>vsaworkingenvironment-&lt;&gt;-single-nas</code><br/><code>csi.trident.netapp.io</code></li> <li>• <code>vsaworkingenvironment-&lt;&gt;-single-san</code><br/><code>csi.trident.netapp.io</code></li> </ul> |



These requirements assume that Astra Control Center is the only application running in the operational environment. If the environment is running additional applications, adjust these minimum requirements accordingly.

## Overview of deployment for GCP

Here is an overview of the process to install Astra Control Center on a self-managed OCP cluster in GCP with Cloud Volumes ONTAP as a storage backend.

Each of these steps is explained in more detail below.

1. [Install a RedHat OpenShift cluster on GCP.](#)
2. [Create a GCP Project and Virtual Private Cloud.](#)

3. [Ensure that you have sufficient IAM permissions.](#)
4. [Configure GCP.](#)
5. [Configure NetApp BlueXP for GCP.](#)
6. [Install Astra Control Center for GCP.](#)

## Install a RedHat OpenShift cluster on GCP

The first step is to install a RedHat OpenShift cluster on GCP.

For installation instructions, see the following:

- [Installing an OpenShift cluster in GCP](#)
- [Creating a GCP Service Account](#)

## Create a GCP Project and Virtual Private Cloud

Create at least one GCP Project and Virtual Private Cloud (VPC).



OpenShift might create its own resource groups. In addition to these, you should also define a GCP VPC. Refer to OpenShift documentation.

You might want to create a platform cluster resource group and a target app OpenShift cluster resource group.

## Ensure that you have sufficient IAM permissions

Ensure that you have sufficient IAM roles and permissions that enable you to install a RedHat OpenShift cluster and a NetApp BlueXP (formerly Cloud Manager) Connector.

See [Initial GCP credentials and permissions.](#)

## Configure GCP

Next, configure GCP to create a VPC, set up compute instances, create a Google Cloud Object Storage, create an Google Container Register to host the Astra Control Center images, and push the images to this registry.

Follow the GCP documentation to complete the following steps. See [Installing OpenShift cluster in GCP.](#)

1. Create a GCP Project and VPC in the GCP that you plan on using for the OCP cluster with CVO backend.
2. Review the compute instances. This can be a bare metal server or VMs in GCP.
3. If the instance type does not already match the Astra minimum resource requirements for master and worker nodes, change the instance type in GCP to meet the Astra requirements. See [Astra Control Center requirements.](#)
4. Create at least one GCP Cloud Storage Bucket to store your backups.
5. Create a secret, which is required for bucket access.
6. Create a Google Container Registry to host all the Astra Control Center images.
7. Set up Google Container Registry access for Docker push/pull for all the Astra Control Center images.

Example: ACC images can be pushed to this registry by entering the following script:



```
gcloud auth activate-service-account <service account email address>
--key-file=<GCP Service Account JSON file>
```

This script requires an Astra Control Center manifest file and your Google Image Registry location.

Example:

```
manifestfile=astra-control-center-<version>.manifest
GCP_CR_REGISTRY=<target image repository>
ASTRA_REGISTRY=<source ACC image repository>

while IFS= read -r image; do
    echo "image: $ASTRA_REGISTRY/$image $GCP_CR_REGISTRY/$image"
    root_image=${image%:*}
    echo $root_image
    docker pull $ASTRA_REGISTRY/$image
    docker tag $ASTRA_REGISTRY/$image $GCP_CR_REGISTRY/$image
    docker push $GCP_CR_REGISTRY/$image
done < astra-control-center-22.04.41.manifest
```

## 8. Set up DNS zones.

### Configure NetApp BlueXP for GCP

Using NetApp BlueXP (formerly Cloud Manager), create a workspace, add a connector to GCP, create a working environment, and import the cluster.

Follow the BlueXP documentation to complete the following steps. See [Getting started with Cloud Volumes ONTAP in GCP](#).

#### What you'll need

- Access to the GCP Service Account with the required IAM permissions and roles

#### Steps

1. Add your credentials to BlueXP. See [Adding GCP accounts](#).
2. Add a connector for GCP.
  - a. Choose "GCP" as the Provider.
  - b. Enter GCP credentials. See [Creating a connector in GCP from BlueXP](#).
  - c. Ensure that the connector is running and switch to that connector.
3. Create a working environment for your cloud environment.
  - a. Location: "GCP"
  - b. Type: "Cloud Volumes ONTAP HA"
4. Import the OpenShift cluster. The cluster will connect to the working environment you just created.
  - a. View the NetApp cluster details by selecting **K8s > Cluster list > Cluster Details**.

- b. In the upper right corner, note the Trident version.
- c. Note the Cloud Volumes ONTAP cluster storage classes showing "NetApp" as the provisioner.

This imports your Red Hat OpenShift cluster and assigns it a default storage class. You select the storage class.

Trident is automatically installed as part of the import and discovery process.

5. Note all the persistent volumes and volumes in this Cloud Volumes ONTAP deployment.



Cloud Volumes ONTAP can operate as a single node or in High Availability (HA). If HA is enabled, note the HA status and node deployment status running in GCP.

## Install Astra Control Center for GCP

Follow the standard [Astra Control Center installation instructions](#).



GCP uses the Generic S3 bucket type.

1. Generate the Docker Secret to pull images for the Astra Control Center installation:

```
kubectl create secret docker-registry <secret name> --docker
-server=<Registry location> --docker-username=_json_key --docker
-password="$(cat <GCP Service Account JSON file>)" --namespace=pcloud
```

## Deploy Astra Control Center in Microsoft Azure

You can deploy Astra Control Center on a self-managed Kubernetes cluster hosted on a Microsoft Azure public cloud.

### What you'll need for Azure

Before you deploy Astra Control Center in Azure, you will need the following items:


- Astra Control Center license. See [Astra Control Center licensing requirements](#).
- [Meet Astra Control Center requirements](#).
- NetApp Cloud Central account
- If using OCP, Red Hat OpenShift Container Platform (OCP) 4.8
- If using OCP, Red Hat OpenShift Container Platform (OCP) permissions (on namespace level to create pods)
- Azure credentials with permissions that enable you to create buckets and connectors

### Operational environment requirements for Azure

Ensure that the operating environment you choose to host Astra Control Center meets the basic resource requirements outlined in the environment's official documentation.

Astra Control Center requires the following resources in addition to the environment's resource requirements:

See [Astra Control Center operational environment requirements](#).

| Component   | Requirement   |
|---|---|
| <b>Backend NetApp Cloud Volumes ONTAP storage capacity</b>                                    | At least 300GB available  |
| <b>Worker nodes (Azure compute requirement)</b>   | At least 3 worker nodes total, with 4 vCPU cores and 12GB RAM each  |
| <b>Load balancer</b>  | Service type "LoadBalancer" available for ingress traffic to be sent to services in the operational environment cluster   |
| <b>FQDN (Azure DNS zone)</b>  | A method for pointing the FQDN of Astra Control Center to the load balanced IP address  |
| <b>Astra Trident (installed as part of the Kubernetes cluster discovery in NetApp BlueXP)</b> | Astra Trident 21.04 or newer installed and configured and NetApp ONTAP version 9.5 or newer will be used as a storage backend   |
| <b>Image registry</b>   | <p>You must have an existing private registry, such as Azure Container Registry (ACR), to which you can push Astra Control Center build images. You need to provide the URL of the image registry where you will upload the images.</p> <div> You need to enable anonymous access to pull Restic images for backups.</div>   |
| <b>Astra Trident / ONTAP configuration</b>  | <p>Astra Control Center requires that a storage class be created and set as the default storage class. Astra Control Center supports the following ONTAP Kubernetes storage classes that are created when you import your Kubernetes cluster into NetApp BlueXP. These are provided by Astra Trident:</p> <ul style="list-style-type: none"><li>• <code>vsaworkingenvironment-&lt;&gt;-ha-nas csi.trident.netapp.io</code></li><li>• <code>vsaworkingenvironment-&lt;&gt;-ha-san csi.trident.netapp.io</code></li><li>• <code>vsaworkingenvironment-&lt;&gt;-single-nas csi.trident.netapp.io</code></li><li>• <code>vsaworkingenvironment-&lt;&gt;-single-san csi.trident.netapp.io</code></li></ul> |



These requirements assume that Astra Control Center is the only application running in the operational environment. If the environment is running additional applications, adjust these minimum requirements accordingly.

## Overview of deployment for Azure

Here is an overview of the process to install Astra Control Center for Azure.

Each of these steps is explained in more detail below.

1. [Install a RedHat OpenShift cluster on Azure.](#)
2. [Create Azure resource groups.](#)
3. [Ensure that you have sufficient IAM permissions.](#)
4. [Configure Azure.](#)
5. [Configure NetApp BlueXP \(formerly Cloud Manager\) for Azure.](#)
6. [Install and configure Astra Control Center for Azure.](#)

## Install a RedHat OpenShift cluster on Azure

The first step is to install a RedHat OpenShift cluster on Azure.

For installation instructions, see the following:

- [Installing OpenShift cluster on Azure.](#)
- [Installing an Azure account.](#)

## Create Azure resource groups

Create at least one Azure resource group.



OpenShift might create its own resource groups. In addition to these, you should also define Azure resource groups. Refer to OpenShift documentation.

You might want to create a platform cluster resource group and a target app OpenShift cluster resource group.

## Ensure that you have sufficient IAM permissions

Ensure that you have sufficient IAM roles and permissions that enable you to install a RedHat OpenShift cluster and a NetApp BlueXP Connector.

See [Azure credentials and permissions.](#)

## Configure Azure

Next, configure Azure to create a virtual network, set up compute instances, create an Azure Blob container, create an Azure Container Register (ACR) to host the Astra Control Center images, and push the images to this registry.

Follow the Azure documentation to complete the following steps. See [Installing OpenShift cluster on Azure.](#)

1. Create an Azure virtual network.
2. Review the compute instances. This can be a bare metal server or VMs in Azure.
3. If the instance type does not already match the Astra minimum resource requirements for master and worker nodes, change the instance type in Azure to meet the Astra requirements. See [Astra Control Center requirements.](#)
4. Create at least one Azure Blob container to store your backups.
5. Create a storage account. You will need a storage account to create a container to be used as a bucket in Astra Control Center.

6. Create a secret, which is required for bucket access.
7. Create an Azure Container Registry (ACR) to host all the Astra Control Center images.
8. Set up ACR access for Docker push/pull all the Astra Control Center images.
9. Push the ACC images to this registry by entering the following script:

```
az acr login -n <AZ ACR URL/Location>
This script requires ACC manifest file and your Azure ACR location.
```

**Example:**

```
manifestfile=astra-control-center-<version>.manifest
AZ_ACR_REGISTRY=<target image repository>
ASTRA_REGISTRY=<source ACC image repository>

while IFS= read -r image; do
    echo "image: $ASTRA_REGISTRY/$image $AZ_ACR_REGISTRY/$image"
    root_image=${image%:*}
    echo $root_image
    docker pull $ASTRA_REGISTRY/$image
    docker tag $ASTRA_REGISTRY/$image $AZ_ACR_REGISTRY/$image
    docker push $AZ_ACR_REGISTRY/$image
done < astra-control-center-22.04.41.manifest
```

10. Set up DNS zones.

### Configure NetApp BlueXP (formerly Cloud Manager) for Azure

Using BlueXP (formerly Cloud Manager), create a workspace, add a connector to Azure, create a working environment, and import the cluster.

Follow the BlueXP documentation to complete the following steps. See [Getting started with BlueXP in Azure](#).

#### What you'll need

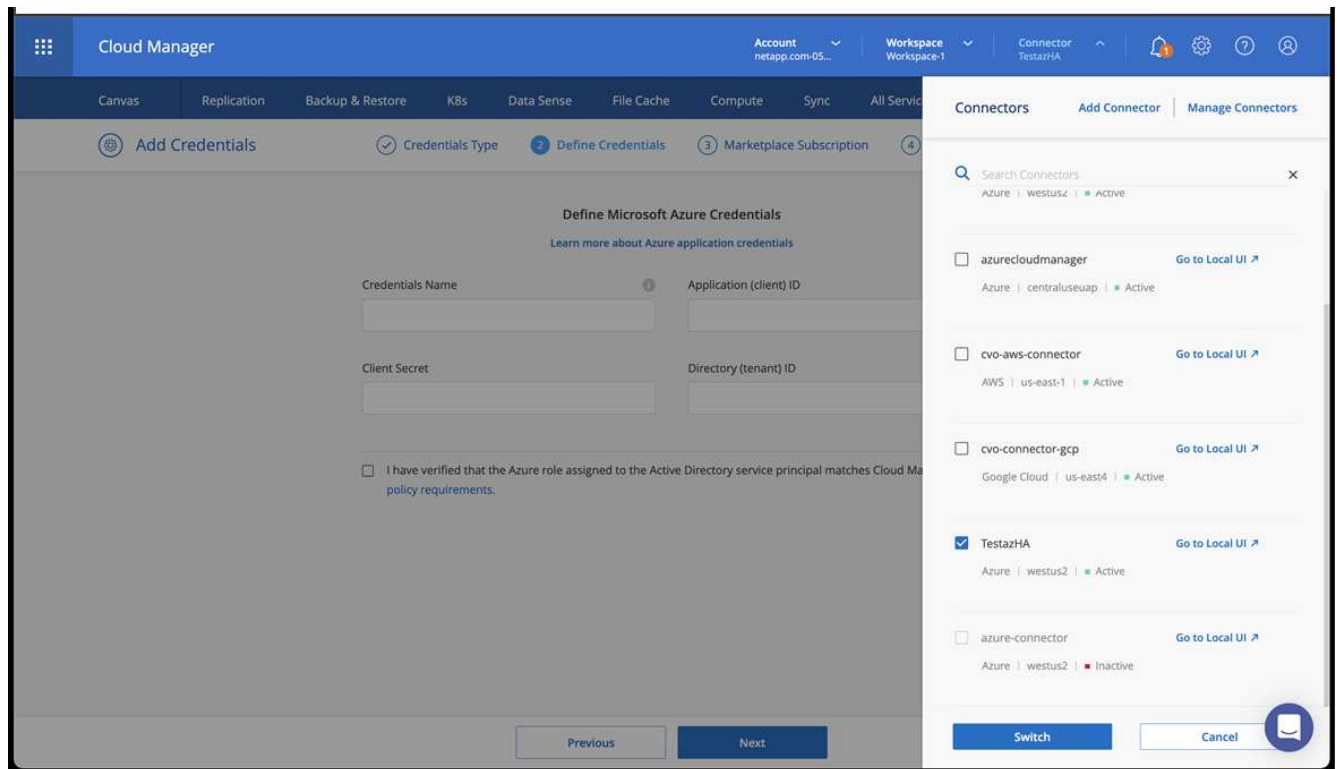
Access to the Azure account with the required IAM permissions and roles

#### Steps

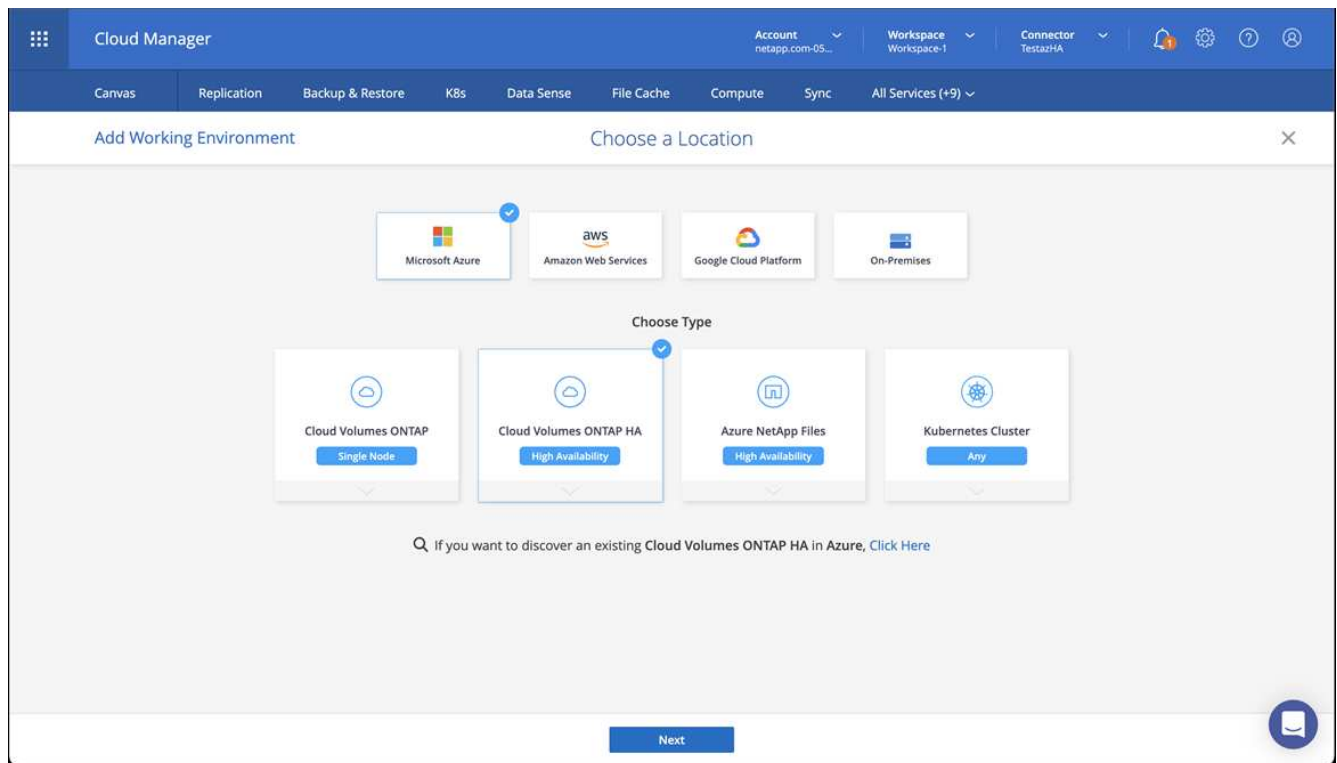
1. Add your credentials to BlueXP.
2. Add a connector for Azure. See [BlueXP policies](#).
  - a. Choose **Azure** as the Provider.
  - b. Enter Azure credentials, including the application ID, client secret, and directory (tenant) ID.

See [Creating a connector in Azure from BlueXP](#).

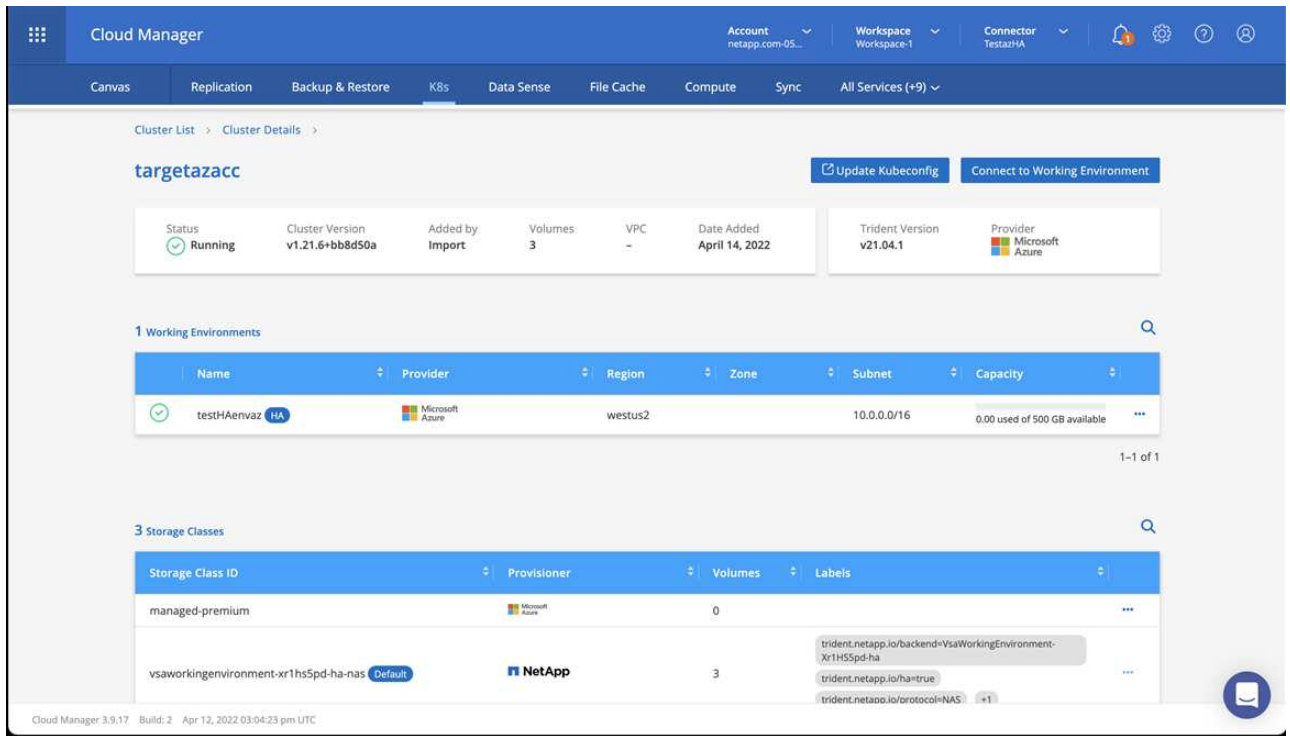
3. Ensure that the connector is running and switch to that connector.



4. Create a working environment for your cloud environment.
  - a. Location: "Microsoft Azure".
  - b. Type: "Cloud Volumes ONTAP HA".



5. Import the OpenShift cluster. The cluster will connect to the working environment you just created.
  - a. View the NetApp cluster details by selecting **K8s > Cluster list > Cluster Details**.



b. In the upper right corner, note the Trident version.

c. Note the Cloud Volumes ONTAP cluster storage classes showing NetApp as the provisioner.

This imports your Red Hat OpenShift cluster and assigns a default storage class. You select the storage class.

Trident is automatically installed as part of the import and discovery process.

6. Note all the persistent volumes and volumes in this Cloud Volumes ONTAP deployment.

7. Cloud Volumes ONTAP can operate as a single node or in High Availability. If HA is enabled, note the HA status and node deployment status running in Azure.

## Install and configure Astra Control Center for Azure

Install Astra Control Center with the standard [installation instructions](#).

Using Astra Control Center, add an Azure bucket. See [Set up Astra Control Center and add buckets](#).

## Copyright information

Copyright © 2024 NetApp, Inc. All Rights Reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data -Noncommercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, nonsublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

## Trademark information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.