



Installation overview

Astra Control Center

NetApp
April 26, 2022

Table of Contents

- Installation overview 1
 - Install Astra Control Center using the standard process 1
 - Install Astra Control Center using OpenShift OperatorHub 22
 - Install Astra Control Center with a Cloud Volumes ONTAP storage backend 27

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](#)

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 to install Astra Control Center Operator and Astra Control Center in your environment. You can use this procedure to install Astra Control Center in internet-connected or air-gapped environments.

For Red Hat OpenShift environments, you can also use an [alternative procedure](#) to install Astra Control Center using OpenShift OperatorHub.

What you'll need

- [Before you begin installation, prepare your environment for Astra Control Center deployment.](#)
- Ensure all cluster operators are in a healthy state and available.

OpenShift example:

```
oc get clusteroperators
```

- Ensure all API services are in a healthy state and available:

OpenShift example:

```
oc get apiservices
```

- The Astra FQDN you plan to use needs to be 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.

About this task

The Astra Control Center installation process does the following:

- Installs the Astra components into the `netapp-acc` (or custom-named) namespace.
- Creates a default account.
- Establishes a default administrative user email address and default one-time password of `ACC-<UUID_of_installation>` for this instance of Astra Control Center. This user is assigned the Owner role in the system and is needed for first time login to the UI.
- Helps you determine that all Astra Control Center pods are running.
- Installs the Astra UI.



If you are using Red Hat's Podman instead of Docker Engine, Podman commands can be used in place of Docker commands.



Do not execute the following command during the entirety of the installation process to avoid deleting all Astra Control Center pods: `kubectl delete -f astra_control_center_operator_deploy.yaml`

Steps

To install Astra Control Center, do the following steps:

- [Download the Astra Control Center bundle](#)
- [Unpack the bundle and change directory](#)
- [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 the Astra Control Center bundle

1. Download the Astra Control Center bundle (`astra-control-center-[version].tar.gz`) from the [NetApp Support Site](#).
2. Download the zip of Astra Control Center certificates and keys from the [NetApp Support Site](#).
3. (Optional) Use the following command to verify the signature of the bundle:

```
openssl dgst -sha256 -verify astra-control-center[version].pub  
-signature <astra-control-center[version].sig astra-control-  
center[version].tar.gz
```

Unpack the bundle and change directory

1. Extract the images:

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

2. Change to the Astra directory.

```
cd astra-control-center-[version]
```

Add the images to your local registry

1. Add the files in the Astra Control Center image directory to your local registry.



See sample scripts for the automatic loading of images below.

- a. Log in to your registry:

Docker:

```
docker login [your_registry_path]
```

Podman:

```
podman login [your_registry_path]
```

- b. Use the appropriate script to load the images, tag the images, and push the images to your local registry:

Docker:

```
export REGISTRY=[Docker_registry_path]
for astraImageFile in $(ls images/*.tar) ; do
    # Load to local cache. And store the name of the loaded image
    # trimming the 'Loaded images: '
    astraImage=$(docker load --input ${astraImageFile} | sed 's/Loaded
image: //' )
    astraImage=$(echo ${astraImage} | sed 's!localhost/!!!')
    # Tag with local image repo.
    docker tag ${astraImage} ${REGISTRY}/${astraImage}
    # Push to the local repo.
    docker push ${REGISTRY}/${astraImage}
done
```

Podman:

```

export REGISTRY=[Registry_path]
for astraImageFile in $(ls images/*.tar) ; do
    # Load to local cache. And store the name of the loaded image
    trimming the 'Loaded images: '
    astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image(s): //'')
    astraImage=$(echo ${astraImage} | sed 's!localhost/!!')
    # Tag with local image repo.
    podman tag ${astraImage} ${REGISTRY}/${astraImage}
    # Push to the local repo.
    podman push ${REGISTRY}/${astraImage}
done

```

Set up namespace and secret for registries with auth requirements

1. 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:

```

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
```

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
```

- e. (Optional) If you want the cluster to be automatically managed by Astra Control Center after installation, make sure that you provide the kubeconfig as a secret within the Astra Control Center namespace you intend to deploy into using this command:

```
kubectl create secret generic [acc-kubeconfig-cred or custom secret name] --from-file=<path-to-your-kubeconfig> -n [netapp-acc or custom namespace]
```

Install the Astra Control Center operator

1. 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
```

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

```
imagePullSecrets:
- name: <name_of_secret_with_creds_to_local_registry>
```

- 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](#).
- d. (For installations using Astra Data Store preview) See this known issue regarding [storage class provisioners and additional changes you will need to make to the YAML](#).

```

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
  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
          command:
            - /manager
          env:
            - name: ACCOP_LOG_LEVEL
              value: "2"
            image: [your_registry_path]/acc-operator:[version x.y.z]
          imagePullPolicy: IfNotPresent
      imagePullSecrets: []

```

2. 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
```

Configure Astra Control Center

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



If additional customizations are required for your environment, you can use `astra_control_center.yaml` as an alternative CR. `astra_control_center_min.yaml` is the default CR and is suitable for most installations.

```
vim astra_control_center_min.yaml
```



Properties configured by the CR cannot be changed after initial Astra Control Center deployment.



If you are using a registry that does not require authorization, you must delete the `secret` line within `imageRegistry` or the installation will fail.

- a. Change `[your_registry_path]` to the registry path where you pushed the images in the previous step.
- b. Change the `accountName` string to the name you want to associate with the account.
- c. Change the `astraAddress` string to the FQDN you want to use in your browser to access Astra. Do not use `http://` or `https://` in the address. Copy this FQDN for use in a [later step](#).

- d. Change the `email` string to the default initial administrator address. Copy this email address for use in a [later step](#).
- e. Change `enrolled for AutoSupport` to `false` for sites without internet connectivity or retain `true` for connected sites.
- f. (Optional) Add a first name `firstName` and last name `lastName` of the user associated with the account. You can perform this step now or later within the UI.
- g. (Optional) Change the `storageClass` value to another Trident `storageClass` resource if required by your installation.
- h. (Optional) If you want the cluster to be automatically managed by Astra Control Center after installation and you have already [created the secret containing the kubeconfig for this cluster](#), provide the name of the secret by adding a new field to this YAML file called `astraKubeConfigSecret`: `"acc-kubeconfig-cred` or `custom secret name"`
- i. Complete one of the following steps:

- **Other ingress controller (`ingressType:Generic`):** This is the default action with Astra Control Center. After Astra Control Center is deployed, you will need to configure the ingress controller to expose Astra Control Center with a URL.

The default Astra Control Center installation sets up its gateway (`service/traefik`) to be of the type `ClusterIP`. This default installation requires you to additionally set up a Kubernetes IngressController/Ingress to route traffic to it. If you want to use an ingress, see [Set up ingress for load balancing](#).

- **Service load balancer (`ingressType:AccTraefik`):** If you don't want to install an IngressController or create an Ingress resource, set `ingressType` to `AccTraefik`.

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](#).

- j. (For installations using Astra Data Store preview) See this known issue for [additional required changes](#) to the YAML.

```
apiVersion: astra.netapp.io/v1
kind: AstraControlCenter
metadata:
  name: astra
spec:
  accountName: "Example"
  astraVersion: "ASTRA_VERSION"
  astraAddress: "astra.example.com"
  astraKubeConfigSecret: "acc-kubeconfig-cred or custom secret name"
  ingressType: "Generic"
  autoSupport:
    enrolled: true
  email: "[admin@example.com]"
  firstName: "SRE"
  lastName: "Admin"
  imageRegistry:
    name: "[your_registry_path]"
    secret: "astra-registry-cred"
  storageClass: "ontap-gold"
```

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_min.yaml -n [netapp-acc or custom namespace]
```

Sample response:

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

Verify system status



If you prefer to use OpenShift, you can use comparable oc commands for verification 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-5f75c5f564-bzqmt 11m	1/1	Running	0
activity-6b8f7cccb9-mlrn4 9m2s	1/1	Running	0
api-token-authentication-6hznt 8m50s	1/1	Running	0
api-token-authentication-qpfqb 8m50s	1/1	Running	0
api-token-authentication-sqnb7 8m50s	1/1	Running	0
asup-5578bbdd57-dxkbp 9m3s	1/1	Running	0
authentication-56bff4f95d-mspmq 7m31s	1/1	Running	0
bucket-service-6f7968b95d-9rrrl 8m36s	1/1	Running	0
cert-manager-5f6cf4bc4b-82khn 6m19s	1/1	Running	0
cert-manager-cainjector-76cf976458-sdrbc 6m19s	1/1	Running	0
cert-manager-webhook-5b7896bfd8-2n45j 6m19s	1/1	Running	0
cloud-extension-749d9f684c-8bdhq 9m6s	1/1	Running	0
cloud-insights-service-7d58687d9-h5tzw 8m56s	1/1	Running	2
composite-compute-968c79cb5-nv7l4 9m11s	1/1	Running	0
composite-volume-7687569985-jg9gg 8m33s	1/1	Running	0
credentials-5c9b75f4d6-nx9cz	1/1	Running	0

8m42s			
entitlement-6c96fd8b78-zt7f8	1/1	Running	0
8m28s			
features-5f7bfc9f68-gsjnl	1/1	Running	0
8m57s			
fluent-bit-ds-h88p7	1/1	Running	0
7m22s			
fluent-bit-ds-krhnj	1/1	Running	0
7m23s			
fluent-bit-ds-l5bjj	1/1	Running	0
7m22s			
fluent-bit-ds-lrclb	1/1	Running	0
7m23s			
fluent-bit-ds-s5t4n	1/1	Running	0
7m23s			
fluent-bit-ds-zpr6v	1/1	Running	0
7m22s			
graphql-server-5f5976f4bd-vbb4z	1/1	Running	0
7m13s			
identity-56f78b8f9f-8h9p9	1/1	Running	0
8m29s			
influxdb2-0	1/1	Running	0
11m			
krakend-6f8d995b4d-5khkl	1/1	Running	0
7m7s			
license-5b5db87c97-jmxzc	1/1	Running	0
9m			
login-ui-57b57c74b8-6xtv7	1/1	Running	0
7m10s			
loki-0	1/1	Running	0
11m			
monitoring-operator-9dbc9c76d-8znck	2/2	Running	0
7m33s			
nats-0	1/1	Running	0
11m			
nats-1	1/1	Running	0
10m			
nats-2	1/1	Running	0
10m			
nautilus-6b9d88bc86-h8kfb	1/1	Running	0
8m6s			
nautilus-6b9d88bc86-vn68r	1/1	Running	0
8m35s			
openapi-b87d77dd8-5dz9h	1/1	Running	0
9m7s			
polaris-consul-consul-5ljfb	1/1	Running	0

11m			
polaris-consul-consul-s5d5z	1/1	Running	0
11m			
polaris-consul-consul-server-0	1/1	Running	0
11m			
polaris-consul-consul-server-1	1/1	Running	0
11m			
polaris-consul-consul-server-2	1/1	Running	0
11m			
polaris-consul-consul-twmpq	1/1	Running	0
11m			
polaris-mongodb-0	2/2	Running	0
11m			
polaris-mongodb-1	2/2	Running	0
10m			
polaris-mongodb-2	2/2	Running	0
10m			
polaris-ui-84dc87847f-zrg8w	1/1	Running	0
7m12s			
polaris-vault-0	1/1	Running	0
11m			
polaris-vault-1	1/1	Running	0
11m			
polaris-vault-2	1/1	Running	0
11m			
public-metrics-657698b66f-67pgt	1/1	Running	0
8m47s			
storage-backend-metrics-6848b9fd87-w7x8r	1/1	Running	0
8m39s			
storage-provider-5ff5868cd5-r9hj7	1/1	Running	0
8m45s			
telegraf-ds-dw4hg	1/1	Running	0
7m23s			
telegraf-ds-k92gn	1/1	Running	0
7m23s			
telegraf-ds-mmxml	1/1	Running	0
7m23s			
telegraf-ds-nhs8s	1/1	Running	0
7m23s			
telegraf-ds-rj7lw	1/1	Running	0
7m23s			
telegraf-ds-tqrkb	1/1	Running	0
7m23s			
telegraf-rs-9mwgj	1/1	Running	0
7m23s			
telemetry-service-56c49d689b-ffrzx	1/1	Running	0

8m42s	tenancy-767c77fb9d-g9ctv	1/1	Running	0
8m52s	traefik-5857d87f85-7pmx8	1/1	Running	0
6m49s	traefik-5857d87f85-cpxgv	1/1	Running	0
5m34s	traefik-5857d87f85-lvmlb	1/1	Running	0
4m33s	traefik-5857d87f85-t2x1k	1/1	Running	0
4m33s	traefik-5857d87f85-v9wpf	1/1	Running	0
7m3s	trident-svc-595f84dd78-zb816	1/1	Running	0
8m54s	vault-controller-86c94fbf4f-krttq	1/1	Running	0
9m24s				

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 installation success by retrieving the `AstraControlCenter` instance installed by the Astra Control Center Operator.

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

4. In the YAML, check the `status.deploymentState` field in the response for the `Deployed` value. If deployment was unsuccessful, an error message appears instead.
5. To get the one-time password you will use when you log in to Astra Control Center, copy the `status.uuid` value. The password is `ACC-` followed by the UUID value (`ACC-[UUID]` or, in this example, `ACC-9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f`).

Sample YAML Details

```
name: astra
  namespace: netapp-acc
  resourceVersion: "104424560"
  selfLink: /apis/astra.netapp.io/v1/namespaces/netapp-acc/astracontrolcenters/astra
  uid: 9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f
spec:
  accountName: Example
  astraAddress: astra.example.com
  astraVersion: 21.12.60
  autoSupport:
    enrolled: true
    url: https://support.netapp.com/asupprod/post/1.0/postAsup
  crds: {}
  email: admin@example.com
  firstName: SRE
  imageRegistry:
    name: registry_name/astra
    secret: astra-registry-cred
  lastName: Admin
status:
  accConditionHistory:
    items:
      - astraVersion: 21.12.60
        condition:
          lastTransitionTime: "2021-11-23T02:23:59Z"
          message: Deploying is currently in progress.
          reason: InProgress
          status: "False"
          type: Ready
        generation: 2
    observedSpec:
      accountName: Example
      astraAddress: astra.example.com
      astraVersion: 21.12.60
      autoSupport:
        enrolled: true
        url: https://support.netapp.com/asupprod/post/1.0/postAsup
      crds: {}
      email: admin@example.com
      firstName: SRE
      imageRegistry:
        name: registry_name/astra
        secret: astra-registry-cred
```



```

    lastName: Admin
    timestamp: "2021-11-23T02:23:59Z"
- astraVersion: 21.12.60
  condition:
    lastTransitionTime: "2021-11-23T02:23:59Z"
    message: Deploying is currently in progress.
    reason: InProgress
    status: "True"
    type: Deploying
  generation: 2
  observedSpec:
    accountName: Example
    astraAddress: astra.example.com
    astraVersion: 21.12.60
    autoSupport:
      enrolled: true
      url: https://support.netapp.com/asupprod/post/1.0/postAsup
    crds: {}
    email: admin@example.com
    firstName: SRE
    imageRegistry:
      name: registry_name/astra
      secret: astra-registry-cred
    lastName: Admin
    timestamp: "2021-11-23T02:23:59Z"
- astraVersion: 21.12.60
  condition:
    lastTransitionTime: "2021-11-23T02:29:41Z"
    message: Post Install was successful
    observedGeneration: 2
    reason: Complete
    status: "True"
    type: PostInstallComplete
  generation: 2
  observedSpec:
    accountName: Example
    astraAddress: astra.example.com
    astraVersion: 21.12.60
    autoSupport:
      enrolled: true
      url: https://support.netapp.com/asupprod/post/1.0/postAsup
    crds: {}
    email: admin@example.com
    firstName: SRE
    imageRegistry:
      name: registry_name/astra

```

```

    secret: astra-registry-cred
    lastName: Admin
    timestamp: "2021-11-23T02:29:41Z"
- astraVersion: 21.12.60
  condition:
    lastTransitionTime: "2021-11-23T02:29:41Z"
    message: Deploying succeeded.
    reason: Complete
    status: "False"
    type: Deploying
  generation: 2
  observedGeneration: 2
  observedSpec:
    accountName: Example
    astraAddress: astra.example.com
    astraVersion: 21.12.60
    autoSupport:
      enrolled: true
      url: https://support.netapp.com/asupprod/post/1.0/postAsup
    crds: {}
    email: admin@example.com
    firstName: SRE
    imageRegistry:
      name: registry_name/astra
      secret: astra-registry-cred
      lastName: Admin
    observedVersion: 21.12.60
    timestamp: "2021-11-23T02:29:41Z"
- astraVersion: 21.12.60
  condition:
    lastTransitionTime: "2021-11-23T02:29:41Z"
    message: Astra is deployed
    reason: Complete
    status: "True"
    type: Deployed
  generation: 2
  observedGeneration: 2
  observedSpec:
    accountName: Example
    astraAddress: astra.example.com
    astraVersion: 21.12.60
    autoSupport:
      enrolled: true
      url: https://support.netapp.com/asupprod/post/1.0/postAsup
    crds: {}
    email: admin@example.com

```

```

    firstName: SRE
    imageRegistry:
      name: registry_name/astra
      secret: astra-registry-cred
    lastName: Admin
    observedVersion: 21.12.60
    timestamp: "2021-11-23T02:29:41Z"
- astraVersion: 21.12.60
  condition:
    lastTransitionTime: "2021-11-23T02:29:41Z"
    message: Astra is deployed
    reason: Complete
    status: "True"
    type: Ready
  generation: 2
  observedGeneration: 2
  observedSpec:
    accountName: Example
    astraAddress: astra.example.com
    astraVersion: 21.12.60
    autoSupport:
      enrolled: true
      url: https://support.netapp.com/asupprod/post/1.0/postAsup
    crds: {}
    email: admin@example.com
    firstName: SRE
    imageRegistry:
      name: registry_name/astra
      secret: astra-registry-cred
    lastName: Admin
    observedVersion: 21.12.60
    timestamp: "2021-11-23T02:29:41Z"
certManager: deploy
cluster:
  type: OCP
  vendorVersion: 4.7.5
  version: v1.20.0+bafe72f
conditions:
- lastTransitionTime: "2021-12-08T16:19:55Z"
  message: Astra is deployed
  reason: Complete
  status: "True"
  type: Ready
- lastTransitionTime: "2021-12-08T16:19:55Z"
  message: Deploying succeeded.
  reason: Complete

```

```

    status: "False"
    type: Deploying
  - lastTransitionTime: "2021-12-08T16:19:53Z"
    message: Post Install was successful
    observedGeneration: 2
    reason: Complete
    status: "True"
    type: PostInstallComplete
  - lastTransitionTime: "2021-12-08T16:19:55Z"
    message: Astra is deployed
    reason: Complete
    status: "True"
    type: Deployed
deploymentState: Deployed
observedGeneration: 2
observedSpec:
  accountName: Example
  astraAddress: astra.example.com
  astraVersion: 21.12.60
  autoSupport:
    enrolled: true
    url: https://support.netapp.com/asupprod/post/1.0/postAsup
  crds: {}
  email: admin@example.com
  firstName: SRE
  imageRegistry:
    name: registry_name/astra
    secret: astra-registry-cred
  lastName: Admin
  observedVersion: 21.12.60
  postInstall: Complete
  uuid: 9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f
kind: List
metadata:
  resourceVersion: ""
  selfLink: ""

```

Set up ingress for load balancing

You can set up a Kubernetes ingress controller that manages external access to services, such as load balancing in a cluster.

This procedure explains how to set up an ingress controller (`ingressType:Generic`). This is the default action with Astra Control Center. After Astra Control Center is deployed, you will need to configure the ingress controller to expose Astra Control Center with a URL.



If you don't want to set up an ingress controller, you can set `ingressType:AccTraefik`. 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](#).

The steps differ depending on the type of ingress controller you use:

- 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.
- You are using Kubernetes versions between and including v1.19 and v1.22.

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 either the `v1beta1` (deprecated in Kubernetes version less than or 1.22) or `v1` resource type for either a deprecated or a new schema:
 - a. For a `v1beta1` deprecated schema, follow this sample:

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: ingress-acc
  namespace: [netapp-acc or custom namespace]
  annotations:
    kubernetes.io/ingress.class: [class name for nginx controller]
spec:
  tls:
    - hosts:
        - <ACC address>
      secretName: [tls secret name]
  rules:
    - host: [ACC address]
      http:
        paths:
          - backend:
              serviceName: traefik
              servicePort: 80
            pathType: ImplementationSpecific
```

b. For the v1 new schema, follow this sample:

```

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

```

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 you used in the `astraAddress` in the `astra_control_center_min.yaml` CR when [you installed Astra Control Center](#).
2. Accept the self-signed certificates when 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_min.yaml` CR when [you installed Astra Control Center](#), followed by the one-time 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 --follow -n netapp-acc-operator $(kubectl get pods -n netapp-acc-operator -o name) -c manager
```

What's next

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

- [Before you begin installation, prepare your environment for Astra Control Center deployment](#).
- From your OpenShift cluster, ensure all cluster operators are in a healthy state (`available` is `true`):

```
oc get clusteroperators
```


- From your OpenShift cluster, ensure all API services are in a healthy state (`available is true`):

```
oc get apiservices
```

- You have created an FQDN address for Astra Control Center in your data center.
- You have the necessary permissions and access to the Red Hat OpenShift Container Platform to perform the installation steps described.

Steps

- [Download the Astra Control Center bundle](#)
- [Unpack the bundle and change directory](#)
- [Add the images to your local registry](#)
- [Find the operator install page](#)
- [Install the operator](#)
- [Install Astra Control Center](#)

Download the Astra Control Center bundle

1. Download the Astra Control Center bundle (`astra-control-center-[version].tar.gz`) from the [NetApp Support Site](#).
2. Download the zip of Astra Control Center certificates and keys from [NetApp Support Site](#).
3. (Optional) Use the following command to verify the signature of the bundle:

```
openssl dgst -sha256 -verify astra-control-center[version].pub  
-signature <astra-control-center[version].sig astra-control-  
center[version].tar.gz
```

Unpack the bundle and change directory

1. Extract the images:

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

2. Change to the Astra directory.

```
cd astra-control-center-[version]
```

Add the images to your local registry

1. Add the files in the Astra Control Center image directory to your local registry.



See sample scripts for the automatic loading of images below.

a. Log in to your registry:

Docker:

```
docker login [your_registry_path]
```

Podman:

```
podman login [your_registry_path]
```

b. Use the appropriate script to load the images, tag the images, and push the images to your local registry:

Docker:

```
export REGISTRY=[Docker_registry_path]
for astraImageFile in $(ls images/*.tar) ; do
    # Load to local cache. And store the name of the loaded image
    # trimming the 'Loaded images: '
    astraImage=$(docker load --input ${astraImageFile} | sed 's/Loaded
image: //' )
    astraImage=$(echo ${astraImage} | sed 's!localhost/!!')
    # Tag with local image repo.
    docker tag ${astraImage} ${REGISTRY}/${astraImage}
    # Push to the local repo.
    docker push ${REGISTRY}/${astraImage}
done
```

Podman:

```

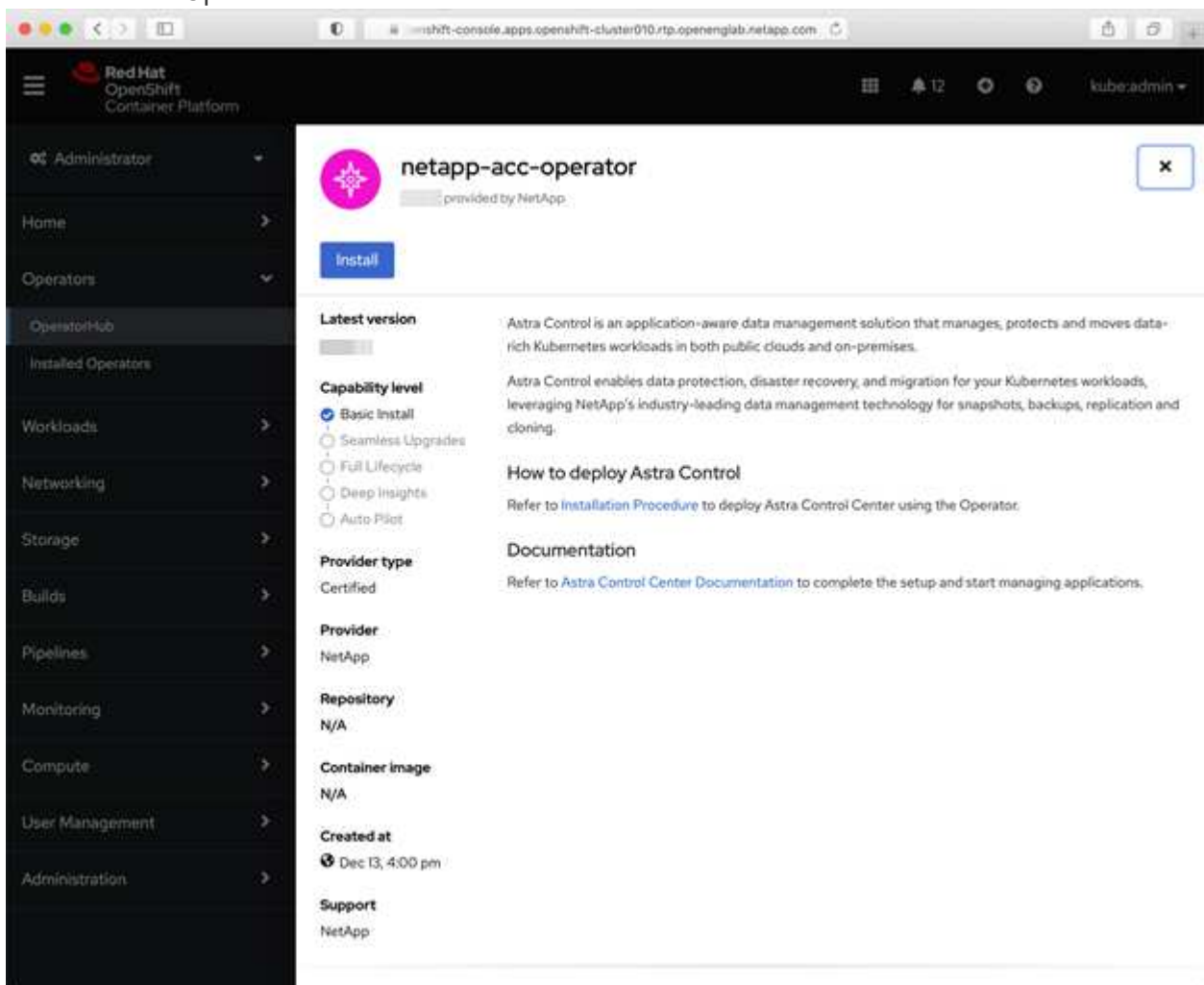
export REGISTRY=[Registry_path]
for astraImageFile in $(ls images/*.tar) ; do
    # Load to local cache. And store the name of the loaded image
    trimming the 'Loaded images: '
    astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image(s): //'')
    astraImage=$(echo ${astraImage} | sed 's!localhost/!!')
    # Tag with local image repo.
    podman tag ${astraImage} ${REGISTRY}/${astraImage}
    # Push to the local repo.
    podman push ${REGISTRY}/${astraImage}
done

```

Find the operator install page

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

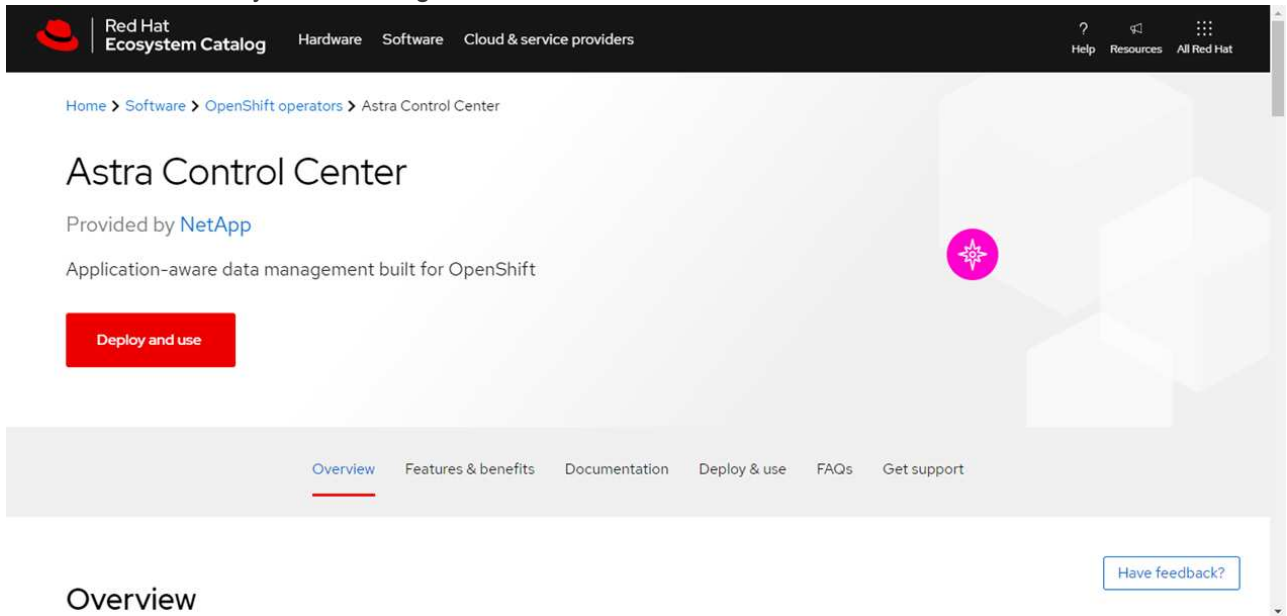
- From Red Hat OpenShift web console:



a. Log in to the OpenShift Container Platform UI.

- b. From the side menu, select **Operators > OperatorHub**.
- c. Select the NetApp Astra Control Center operator.
- d. Select **Install**.

° From Red Hat Ecosystem Catalog:



- a. Select the NetApp Astra Control Center [operator](#).
- b. Select **Deploy and Use**.

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 details view of the Astra Control Center operator, select `Create instance` in the Provided APIs section.

2. Complete the `Create AstraControlCenter` form field:
 - a. Keep or adjust the Astra Control Center name.
 - b. (Optional) Enable or disable Auto Support. Retaining Auto Support functionality is recommended.
 - c. Enter the Astra Control Center address. Do not enter `http://` or `https://` in the address.
 - d. Enter the Astra Control Center version; for example, 21.12.60.
 - e. Enter an account name, email address, and admin last name.
 - f. Retain the default volume reclaim policy.
 - g. In **Image Registry**, enter your local container image registry path. Do not enter `http://` or `https://` in the address.
 - h. If you use a registry that requires authentication, enter the secret.
 - i. Enter the admin first name.
 - j. Configure resources scaling.
 - k. Retain the default storage class.
 - l. Define CRD handling preferences.
3. Select `Create`.

What's next

Verify the successful installation of Astra Control Center and complete the [remaining steps](#) to log in. Additionally, you will complete the deployment by also performing [setup tasks](#).

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) 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 Microsoft Azure](#)

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.

Only self-managed OpenShift Container Platform (OCP) clusters are supported for deploying Astra Control Center.

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
- 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
Backend NetApp Cloud Volumes ONTAP storage capacity	At least 300GB available
Worker nodes (AWS EC2 requirement)	At least 3 worker nodes total, with 4 vCPU cores and 12GB RAM each
Load balancer	Service type "LoadBalancer" available for ingress traffic to be sent to services in the operational environment cluster
FQDN	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 Cloud Manager)	Astra Trident 21.04 or newer installed and configured and NetApp ONTAP version 9.5 or newer as a storage backend

Component	Requirement
Image registry	<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>
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 Cloud Manager. These are provided by Astra Trident:</p> <ul style="list-style-type: none"> • <code>vsaworkingenvironment-<>-ha-nas</code> <code>csi.trident.netapp.io</code> • <code>vsaworkingenvironment-<>-ha-san</code> <code>csi.trident.netapp.io</code> • <code>vsaworkingenvironment-<>-single-nas</code> <code>csi.trident.netapp.io</code> • <code>vsaworkingenvironment-<>-single-san</code> <code>csi.trident.netapp.io</code>



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 Cloud Manager.](#)
5. [Install Astra Control Center.](#)

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 Cloud Volumes ONTAP 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 private cloud, 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 private cloud.
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 CVO 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 (CVO) 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 Cloud Manager

Using Cloud Manager, create a workspace, add a connector to AWS, create a working environment, and import the cluster.

Follow the Cloud Manager documentation to complete the following steps. See the following:

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

Steps

1. Add your credentials to Cloud Manager.
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 CVO 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

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

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
- Red Hat OpenShift Container Platform (OCP) 4.8
- 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
Backend NetApp Cloud Volumes ONTAP storage capacity	At least 300GB available
Worker nodes (Azure compute requirement)	At least 3 worker nodes total, with 4 vCPU cores and 12GB RAM each
Load balancer	Service type "LoadBalancer" available for ingress traffic to be sent to services in the operational environment cluster

Component	Requirement
FQDN (Azure 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 Cloud Manager)	Astra Trident 21.04 or newer installed and configured and NetApp ONTAP version 9.5 or newer will be used as a storage backend
Image registry	<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>  <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 Cloud Manager. These are provided by Astra Trident:</p> <ul style="list-style-type: none"> • <code>vsaworkingenvironment-<>-ha-nas</code> <code>csi.trident.netapp.io</code> • <code>vsaworkingenvironment-<>-ha-san</code> <code>csi.trident.netapp.io</code> • <code>vsaworkingenvironment-<>-single-nas</code> <code>csi.trident.netapp.io</code> • <code>vsaworkingenvironment-<>-single-san</code> <code>csi.trident.netapp.io</code>



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 Cloud Manager.](#)

6. [Install and configure Astra Control Center.](#)

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 Cloud Volumes ONTAP connector.

See [Azure credentials and permissions.](#)

Configure Azure

Next, configure Azure to create a virtual private cloud, set up compute instances, create an Azure Blob S3 bucket, 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 private cloud.
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 S3 bucket 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 Cloud Manager

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

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

What you'll need

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

Steps

1. Add your credentials to Cloud Manager.
2. Add a connector for Azure. See [Cloud Manager 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 Cloud Manager](#).

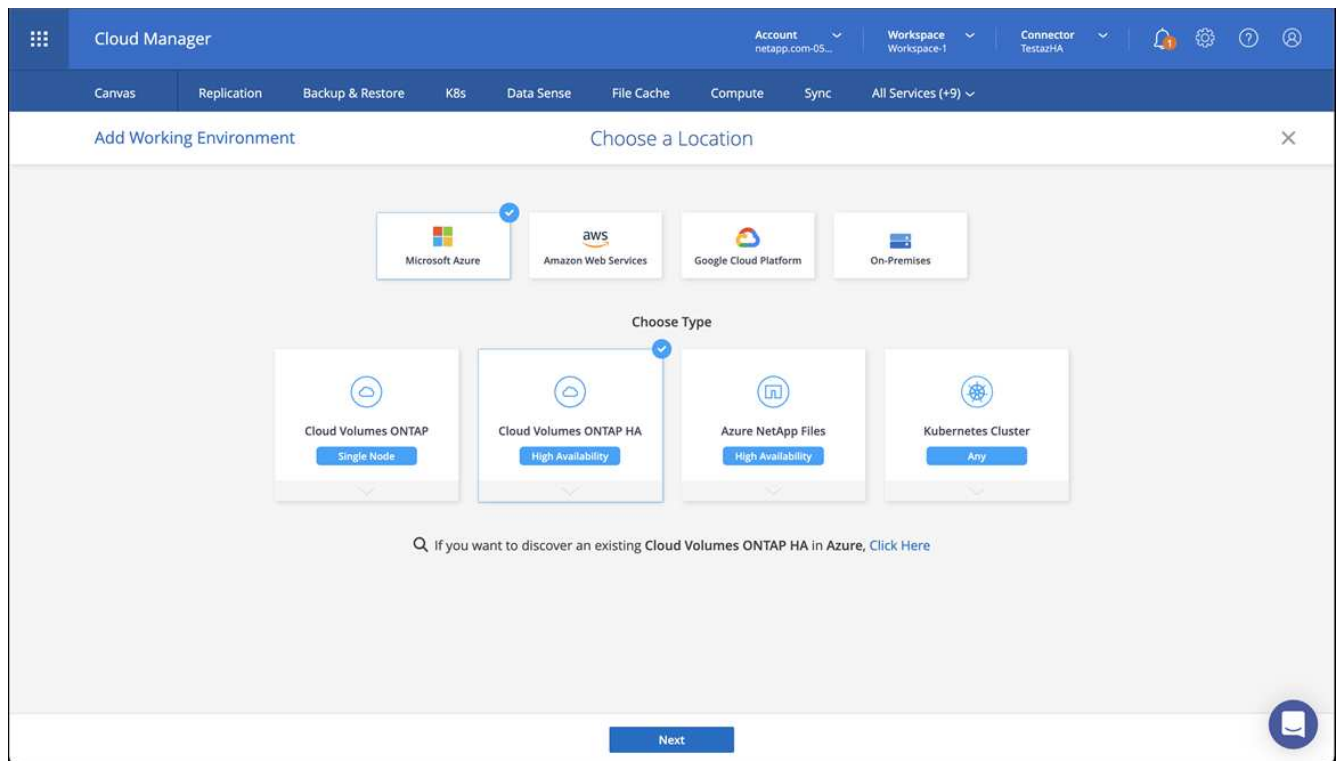
3. Ensure that the connector is running and switch to that connector.



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 CVO 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 CVO 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

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 © 2022 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.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

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.