



Install Astra Control Center

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

- Install Astra Control Center using the standard process 1
 - Download and extract Astra Control Center. 2
 - Install the NetApp Astra kubectl plugin. 2
 - Add the images to your local registry 3
 - Set up namespace and secret for registries with auth requirements 5
 - Install the Astra Control Center operator 7
 - Configure Astra Control Center 10
 - Complete Astra Control Center and operator installation. 22
 - Verify system status 23
 - Set up ingress for load balancing 27
 - Log in to the Astra Control Center UI 31
 - Troubleshoot the installation 31
 - What’s next. 32

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.

```
kubect1 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 Astra Control Center requirements.</p> <p>NOTE: Do not use <code>http://</code> or <code>https://</code> in the address. Copy this FQDN for use in a later step.</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	https://support.netapp.com/asupprod/post/1.0/postAsup

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 later step . 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 previous step . 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. 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

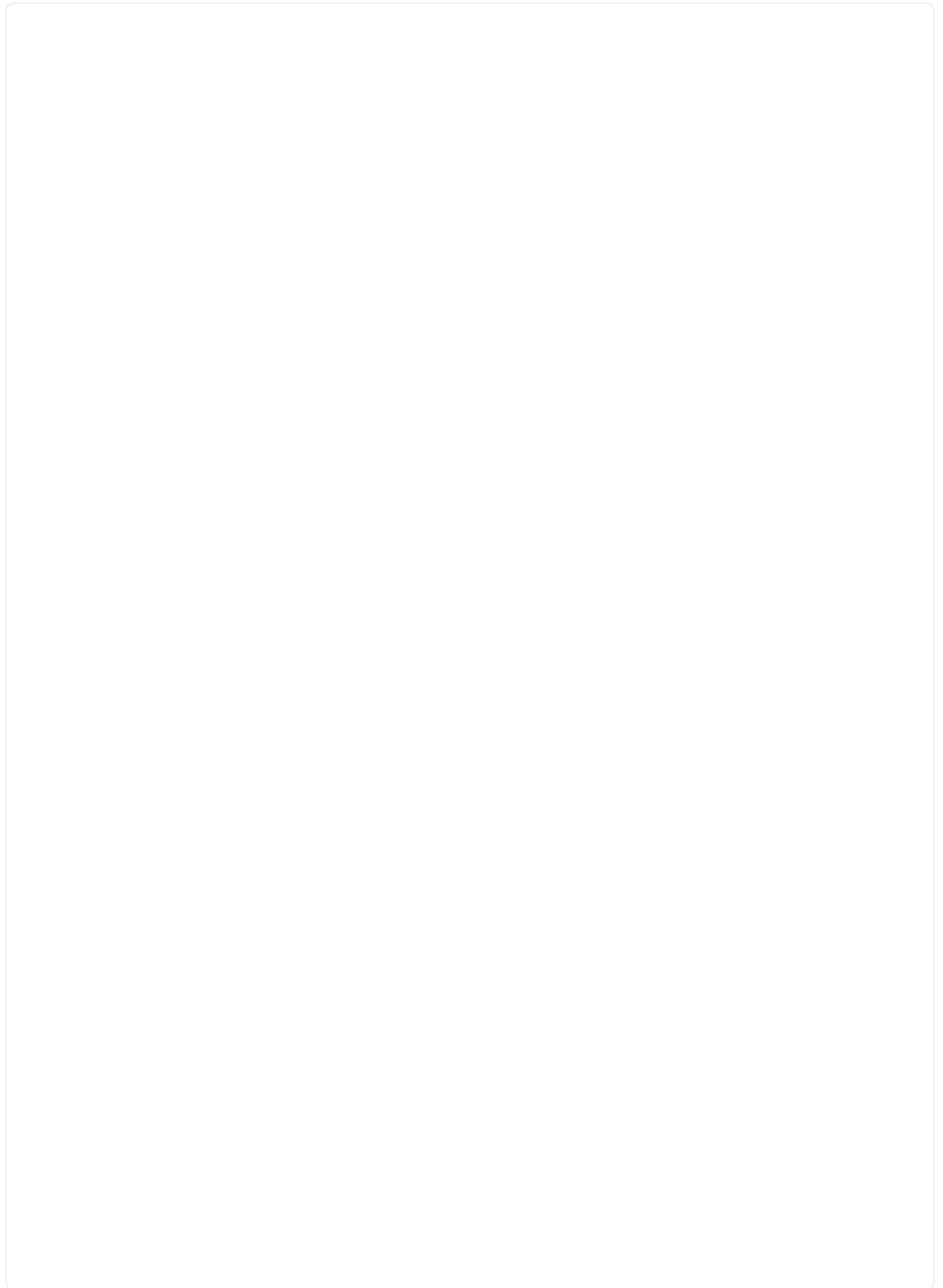
storageClass

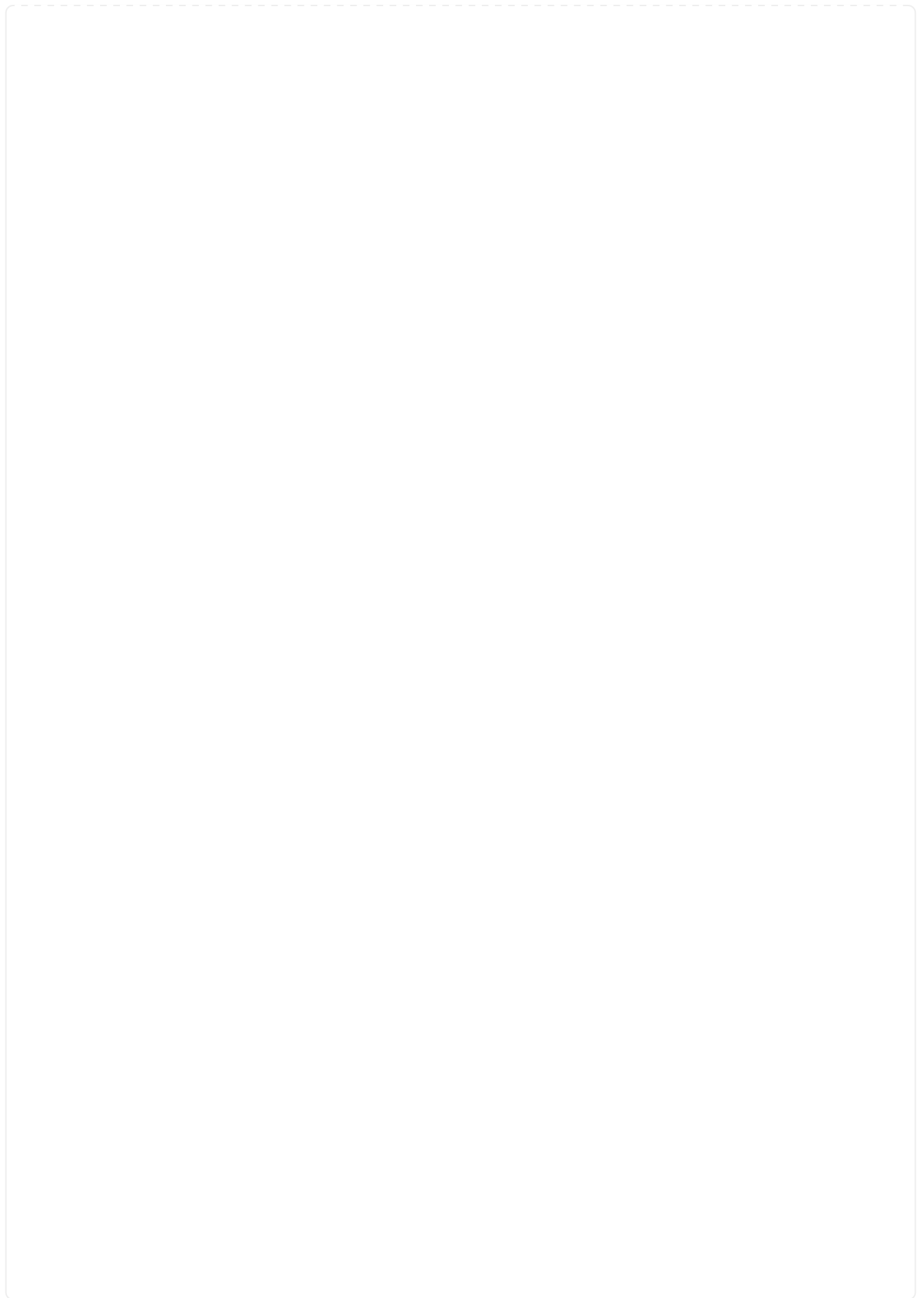
Setting	Guidance	Type	Example
storageClass	<p>Change the storageClass value from <code>ontap-gold</code> to another Trident storageClass resource as required by your installation. Run the command <code>kubectl get sc</code> to determine your existing configured storage classes. One of the Trident-based storage classes must be entered in the manifest file (<code>astra-control-center-<version>.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">• Retain (This is the default value)• Delete

ingressType





Setting	Guidance	Type	Options
ingressType	<p>Use one of the following ingress types:</p> <p>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.</p> <p>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.</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"> • Generic (this is the default value) • AccTraefik

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">• <code>Default</code> (This is the default value)• <code>Off</code>

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 14m	1/1	Running	0
activity-6cc67ff9f4-z48mr (8m32s ago) 9m	1/1	Running	2
api-token-authentication-7s67v 8m56s	1/1	Running	0
api-token-authentication-bplb4 8m56s	1/1	Running	0
api-token-authentication-p2c9z 8m56s	1/1	Running	0
asup-6cdfbc6795-md8vn 9m14s	1/1	Running	0
authentication-9477567db-8hnc9 7m4s	1/1	Running	0
bucket-service-f4dbdfcd6-wqzkw 8m48s	1/1	Running	0
cert-manager-bb756c7c4-wm2cv 14m	1/1	Running	0
cert-manager-cainjector-c9bb86786-8wrf5 14m	1/1	Running	0
cert-manager-webhook-dd465db99-j2w4x 14m	1/1	Running	0
certificates-68dff9cdd6-kcvml (8m43s ago) 9m2s	1/1	Running	2
certificates-68dff9cdd6-rsnsb 9m2s	1/1	Running	0
cloud-extension-69d48c956c-2s8dt (8m43s ago) 9m24s	1/1	Running	3
cloud-insights-service-7c4f48b978-7gvlh (8m50s ago) 9m28s	1/1	Running	3
composite-compute-7d9ff5f68-nxbhl 8m51s	1/1	Running	0
composite-volume-57b4756d64-nl66d 9m13s	1/1	Running	0
credentials-6dbc55f89f-qpzff 11m	1/1	Running	0
entitlement-67bfb6d7-gl6kp (8m33s ago) 9m38s	1/1	Running	4
features-856cc4dccc-mxbdb 9m20s	1/1	Running	0
fluent-bit-ds-4rtsp 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 (6m15s ago) 6m56s	1/1	Running	3
polaris-keycloak-1 4m22s	1/1	Running	0
polaris-keycloak-2 3m41s	1/1	Running	0
polaris-keycloak-db-0 6m56s	1/1	Running	0
polaris-keycloak-db-1 4m23s	1/1	Running	0
polaris-keycloak-db-2 3m36s	1/1	Running	0
polaris-mongodb-0 13m	2/2	Running	0
polaris-mongodb-1 13m	2/2	Running	0
polaris-mongodb-2 12m	2/2	Running	0
polaris-ui-5ccff47897-8rzgh 2m33s	1/1	Running	0
polaris-vault-0 13m	1/1	Running	0
polaris-vault-1 13m	1/1	Running	0
polaris-vault-2 13m	1/1	Running	0
public-metrics-6cb7bfc49b-p54xm (8m29s ago) 9m31s	1/1	Running	1
storage-backend-metrics-5c77994586-kjn48 8m52s	1/1	Running	0
storage-provider-769fdc858c-62w54 8m54s	1/1	Running	0
task-service-9ffc484c5-kx9f4 (8m44s ago) 9m34s	1/1	Running	3
telegraf-ds-bphb9 6m54s	1/1	Running	0
telegraf-ds-rtsm2 6m54s	1/1	Running	0
telegraf-ds-s9h5h 6m54s	1/1	Running	0
telegraf-rs-lbpv7 6m54s	1/1	Running	0
telemetry-service-57cfb998db-zjx78 (8m40s ago) 9m26s	1/1	Running	1
tenancy-5d5dfbcf9f-vmbxh 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-rfkqg	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](#).

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.