Install Astra Control Center

Astra Control Center

Dave Bagwell, amitha
September 28, 2021

Table of Contents

Install Astra Control Center

To install Astra Control Center, do the following steps:

- Install Astra Control Center
- Log in to the Astra Control Center UI

Install Astra Control Center

To install Astra Control Center, download the installation bundle from the NetApp Support Site and perform a series a commands 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.

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
```

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.

Podman commands can be used in place of Docker commands if you are using Red Hat's Podman repository.

Steps

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
```

4. Extract the images:

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

5. Change to the Astra directory.

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

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

See a sample script for the automatic loading of images below.

a. Log in to your Docker registry:

```
docker login [Docker_registry_path]
```

b. Load the images into Docker.

c. Tag the images.

d. Push the images to your local registry.

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

7. (For registries with auth requirements only) 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=[Docker_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]
```

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] --docker-server=[Docker_registry_path] --docker
-username=[username] --docker-password=[token]
```

Response

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

8. 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: astra-registry-cred
```

b. Change `[Docker_registry_path]` for the `kube-rbac-prox` image to the registry path where you pushed the images in a previous step.

c. Change `[Docker_registry_path]` for the `acc-operator-controller-manager` image to the registry path where you pushed the images in a previous step.

```
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: [Docker_registry_path]/kube-rbac-proxy:v0.5.0
        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: [Docker_registry_path]/acc-operator:[version x.y.z]
        imagePullPolicy: IfNotPresent
      imagePullSecrets: []
```

9. Edit the Astra Control Center custom resource (CR) file (`astra_control_center_min.yaml`):

```
vim astra_control_center_min.yaml
```

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.

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

a. Change `[Docker_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. If you are not using a registry that requires authorization, delete the `secret` line.

```
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: "[Docker_registry_path]"
    secret: "astra-registry-cred"
    storageClass: "ontap-gold"
```

10. 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
```

11. 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]
```

Sample response:

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

12. Run the following patch to correct cluster role binding.
13. 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]
```

Sample response:

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

14. Verify that all system components installed successfully.

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

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-5fdfff786f-gkv6z            1/1     Running   0
4m58s
activity-649f869bf7-jn5gs                 1/1     Running   0
3m14s
asup-79846b5fdc-s9s97                     1/1     Running   0
3m10s
authentication-84c78f5cf4-qhx9t           1/1     Running   0
118s
billing-9b8496787-v8rzv                   1/1     Running   0
2m54s
bucketservice-5fb876d9d5-wkfvz            1/1     Running   0
3m26s
cloud-extension-f9f4f59c6-dz6s6           1/1     Running   0
3m
cloud-insights-service-5676b8c6d4-6q7lv   1/1     Running   0
2m52s
composite-compute-7dcc9c6d6c-lxdr6        1/1     Running   0
2m50s
composite-volume-74dbfd7577-cd42b         1/1     Running   0
3m2s
credentials-75dbf46f9d-5qm2b              1/1     Running   0
3m32s
entitlement-6cf875cb48-gkvhp              1/1     Running   0
3m12s
features-74fd97bb46-vss2n                 1/1     Running   0
3m6s
fluent-bit-ds-2g9jb                       1/1     Running   0
113s
fluent-bit-ds-5tg5h                       1/1     Running   0
113s
fluent-bit-ds-qfxb8                       1/1     Running   0
113s
graphql-server-7769f98b86-p4qrv           1/1     Running   0
90s
identity-566c566cd5-ntfj6                 1/1     Running   0
3m16s
```

```
influxdb2-0                                  1/1     Running   0
4m43s
krakend-5cb8d56978-44q66                     1/1     Running   0
93s
license-66cbbc6f48-27kgf                     1/1     Running   0
3m4s
login-ui-584f7fd84b-dmdrp                    1/1     Running   0
87s
loki-0                                       1/1     Running   0
4m44s
metrics-ingestion-service-6dcfddf45f-mhnvh   1/1     Running   0
3m8s
monitoring-operator-78d67b4d4-nxs6v          2/2     Running   0
116s
nats-0                                       1/1     Running   0
4m40s
nats-1                                       1/1     Running   0
4m26s
nats-2                                       1/1     Running   0
4m15s
nautilus-9b664bc55-rn9t8                     1/1     Running   0
2m56s
openapi-dc5ddfb7d-6q8vh                      1/1     Running   0
3m20s
polaris-consul-consul-5tjs7                  1/1     Running   0
4m43s
polaris-consul-consul-5wbnx                  1/1     Running   0
4m43s
polaris-consul-consul-bfvl7                  1/1     Running   0
4m43s
polaris-consul-consul-server-0               1/1     Running   0
4m43s
polaris-consul-consul-server-1               1/1     Running   0
4m43s
polaris-consul-consul-server-2               1/1     Running   0
4m43s
polaris-mongodb-0                            2/2     Running   0
4m49s
polaris-mongodb-1                            2/2     Running   0
4m22s
polaris-mongodb-arbiter-0                    1/1     Running   0
4m49s
polaris-ui-6648875998-75d98                  1/1     Running   0
92s
polaris-vault-0                              1/1     Running   0
4m41s
```

```
polaris-vault-1                              1/1     Running   0
4m41s
polaris-vault-2                              1/1     Running   0
4m41s
storage-backend-metrics-69546f4fc8-m7lfj     1/1     Running   0
3m22s
storage-provider-5d46f755b-qfv89             1/1     Running   0
3m30s
support-5dc579865c-z4pwq                     1/1     Running   0
3m18s
telegraf-ds-4452f                            1/1     Running   0
113s
telegraf-ds-gnqxl                            1/1     Running   0
113s
telegraf-ds-jhw74                            1/1     Running   0
113s
telegraf-rs-gg6m4                            1/1     Running   0
113s
telemetry-service-6dcc875f98-zft26           1/1     Running   0
3m24s
tenancy-7f7f77f699-q7l6w                     1/1     Running   0
3m28s
traefik-769d846f9b-c9crt                     1/1     Running   0
83s
traefik-769d846f9b-l9n4k                     1/1     Running   0
67s
trident-svc-8649c8bfc5-pdj79                 1/1     Running   0
2m57s
vault-controller-745879f98b-49c5v            1/1     Running   0
4m51s
```

15. (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
```

16. When all the pods are running, verify installation success by retrieving the AstraControlCenter instance installed by the ACC Operator.

```
kubectl get acc -o yaml -n netapp-acc
```

17. Check the `status.deploymentState` field in the response for the `Deployed` value. If deployment was unsuccessful, an error message appears instead.

You will use the `uuid` in the next step.

```
apiVersion: v1
items:
- apiVersion: astra.netapp.io/v1
  kind: AstraControlCenter
  metadata:
    creationTimestamp: "2021-07-28T21:36:49Z"
    finalizers:
    - astracontrolcenter.netapp.io/finalizer
   generation: 1
    name: astra
    namespace: netapp-acc
    resourceVersion: "27797604"
    selfLink: /apis/astra.netapp.io/v1/namespaces/netapp-
acc/astracontrolcenters/astra
    uid: 61cd8b65-047b-431a-ba35-510afcb845f1
  spec:
    accountName: Example
    astraAddress: astra.example.com
    astraResourcesScaler: "Off"
    astraVersion: 21.08.52
    autoSupport:
      enrolled: false
    email: admin@example.com
    firstName: SRE
    lastName: Admin
    imageRegistry:
      name: registry_name/astra
  status:
    certManager: deploy
    deploymentState: Deployed
    observedGeneration: 1
    observedVersion: 21.08.52
    postInstall: Complete
    uuid: c49008a5-4ef1-4c5d-a53e-830daf994116
kind: List
metadata:
  resourceVersion: ""
  selfLink: ""
```

18. To get the one-time password you will use when you log in to Astra Control Center, copy the `status.uuid` value from the response in the previous step. The password is `ACC-` followed by the UUID value (`ACC-[UUID]` or, in this example, `ACC-c49008a5-4ef1-4c5d-a53e-830daf994116`).

Log in to the Astra Control Center UI

After installing ACC, you will change the password for the default administrator and log in to the ACC 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 ACC.
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 ACC, 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.