Installieren Sie Astra Control Center

Astra Control Center

NetApp
June 06, 2024

Inhalt

Installieren Sie das Astra Control Center mithilfe des Standardprozesses

Laden Sie zum Installieren des Astra Control Center das Installationspaket von der NetApp Support Site herunter und führen Sie die folgenden Schritte aus, um Astra Control Center Operator und Astra Control Center in Ihrer Umgebung zu installieren. Mit diesem Verfahren können Sie Astra Control Center in Internet-angeschlossenen oder luftgekapderten Umgebungen installieren.

Für Red hat OpenShift-Umgebungen können Sie ein verwenden "Alternativverfahren" So installieren Sie Astra Control Center mithilfe des OpenShift OperatorHub.

Was Sie benötigen

- "Bevor Sie mit der Installation beginnen, bereiten Sie Ihre Umgebung auf die Implementierung des Astra Control Center vor".
- Wenn Sie POD-Sicherheitsrichtlinien in Ihrer Umgebung konfiguriert haben oder konfigurieren möchten, sollten Sie sich mit den POD-Sicherheitsrichtlinien vertraut machen und wie diese sich auf die Installation des Astra Control Center auswirken. Siehe "Einschränkungen der POD-Sicherheitsrichtlinie verstehen".
- Stellen Sie sicher, dass alle Cluster Operator in einem ordnungsgemäßen Zustand und verfügbar sind.

```
kubectl get clusteroperators
```

- Stellen Sie sicher, dass alle API-Services in einem ordnungsgemäßen Zustand und verfügbar sind:

```
kubectl get apiservices
```

- Stellen Sie sicher, dass der Astra FQDN, den Sie verwenden möchten, für diesen Cluster routingfähig ist. Das bedeutet, dass Sie entweder einen DNS-Eintrag in Ihrem internen DNS-Server haben oder eine bereits registrierte Core URL-Route verwenden.
- Wenn bereits ein Zertifikat-Manager im Cluster vorhanden ist, müssen Sie einen Teil durchführen "Erforderliche Schritte" Damit Astra Control Center nicht seinen eigenen Cert-Manager installiert.

Über diese Aufgabe

Der Astra Control Center-Installationsprozess führt Folgendes aus:

- Installiert die Astra-Komponenten im `netapp-acc` (Oder Name des benutzerdefinierten Namespace).
- Erstellt ein Standardkonto.
- Richtet eine standardmäßige E-Mail-Adresse für Administratorbenutzer und ein Standardpasswort ein. Diesem Benutzer wird die Owner-Rolle im System zugewiesen, die für die erste Anmeldung bei der UI erforderlich ist.
- Hilft Ihnen bei der Ermittlung, dass alle Astra Control Center-Pods ausgeführt werden.
- Installiert die Astra UI

(Gilt nur für die Version des Astra Data Store Early Access Program (EAP). Wenn Sie den Astra Data Store über das Astra Control Center verwalten und VMware-Workflows aktivieren möchten, implementieren Sie Astra Control Center nur auf dem `pcloud` Und nicht auf dem `netapp-acc` Namespace oder ein benutzerdefinierter Namespace, der in den Schritten dieses Verfahrens beschrieben wird.

Führen Sie den folgenden Befehl während der gesamten Installation nicht aus, um zu vermeiden, dass alle Astra Control Center Pods gelöscht werden: `kubectl delete -f astra_control_center_operator_deploy.yaml`

Wenn Sie Podman von Red hat anstelle von Docker Engine verwenden, können Podman-Befehle anstelle von Docker-Befehlen verwendet werden.

Schritte

Gehen Sie wie folgt vor, um Astra Control Center zu installieren:

- Laden Sie das Astra Control Center Bundle herunter und entpacken Sie es aus
- Installieren Sie das NetApp Astra kubectl Plug-in
- Fügen Sie die Bilder Ihrer lokalen Registrierung hinzu
- Einrichten von Namespace und Geheimdienstraum für Registrys mit auth Anforderungen
- Installieren Sie den Operator Astra Control Center
- Konfigurieren Sie Astra Control Center
- Komplette Astra Control Center und Bedienerinstallation
- Überprüfen Sie den Systemstatus
- Eindringen für den Lastenausgleich einrichten
- Melden Sie sich in der UI des Astra Control Center an

Laden Sie das Astra Control Center Bundle herunter und entpacken Sie es aus

1. Laden Sie das Astra Control Center Bundle herunter (`astra-control-center-[version].tar.gz`) Vom "NetApp Support Website".
2. Laden Sie den Zip der Astra Control Center Zertifikate und Schlüssel aus dem herunter "NetApp Support Website".
3. (Optional) Überprüfen Sie mit dem folgenden Befehl die Signatur des Pakets:

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

4. Extrahieren Sie die Bilder:

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

Installieren Sie das NetApp Astra kubectl Plug-in

Der NetApp Astra `kubectl` Kommandozeilen-Plugin spart Zeit bei der Ausführung von Routineaufgaben im Zusammenhang mit der Bereitstellung und dem Upgrade von Astra Control Center.

Was Sie benötigen

NetApp bietet Binärdateien für das Plug-in für verschiedene CPU-Architekturen und Betriebssysteme. Sie müssen wissen, welche CPU und welches Betriebssystem Sie haben, bevor Sie diese Aufgabe ausführen. Unter Linux- und Mac-Betriebssystemen können Sie die verwenden `uname -a` Befehl zum Sammeln dieser Informationen.

Schritte

1. Nennen Sie den verfügbaren NetApp Astra `kubectl` Plugin-Binärdateien, und notieren Sie den Namen der Datei, die Sie für Ihr Betriebssystem und CPU-Architektur benötigen:

```
ls kubectl-astra/
```

2. Kopieren Sie die Datei an denselben Speicherort wie der Standard `kubectl` Utility: In diesem Beispiel ist der `kubectl` Das Dienstprogramm befindet sich im `/usr/local/bin` Verzeichnis. Austausch `<binary-name>` Mit dem Namen der benötigten Datei:

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

Fügen Sie die Bilder Ihrer lokalen Registrierung hinzu

1. Führen Sie die entsprechende Schrittfolge für Ihre Container-Engine durch:

Docker

1. Wechseln Sie in das Astra-Verzeichnis:

```
cd acc
```

2. Schieben Sie die Paketbilder im Astra Control Center-Bildverzeichnis in Ihre lokale Registrierung. Führen Sie folgende Ersetzungen durch, bevor Sie den Befehl ausführen:
   - ERSETZEN SIE DIE BUNDLE_FILE durch den Namen der Astra Control Bundle-Datei (z. B. `acc.manifest.yaml`).
   - ERSETZEN SIE MY_REGISTRY durch die URL des Docker Repositorys.
   - ERSETZEN SIE MY_REGISTRY_USER durch den Benutzernamen.
   - ERSETZEN SIE MY_REGISTRY_TOKEN durch ein autorisiertes Token für die Registrierung.

```
kubectl astra packages push-images -m BUNDLE_FILE -r MY_REGISTRY
-u MY_REGISTRY_USER -p MY_REGISTRY_TOKEN
```

Podman

1. Melden Sie sich bei Ihrer Registrierung an:

```
podman login [your_registry_path]
```

2. Führen Sie das folgende Skript aus und machen Sie die Substitution <YOUR_REGISTRY> wie in den Kommentaren angegeben:

```
# You need to be at the root of the tarball.
# You should see these files to confirm correct location:
#   acc.manifest.yaml
#   acc/

# Replace <YOUR_REGISTRY> with your own registry (e.g
registry.customer.com or registry.customer.com/testing, etc..)
export REGISTRY=<YOUR_REGISTRY>
export PACKAGENAME=acc
export PACKAGEVERSION=22.08.1-26
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
  # Load to local cache
  astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image(s): //')

  # Remove path and keep imageName.
  astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')

  # Tag with local image repo.
  podman tag ${astraImage} ${REGISTRY}/netapp/astra/${PACKAGENAME}
/${PACKAGEVERSION}/${astraImageNoPath}

  # Push to the local repo.
  podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

Einrichten von Namespace und Geheimdienstraum für Registrys mit auth Anforderungen

1. Exportieren Sie den KUBECONFIG für den Hostcluster Astra Control Center:

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

2. Wenn Sie eine Registrierung verwenden, für die eine Authentifizierung erforderlich ist, müssen Sie Folgendes tun:

   a. Erstellen Sie die `netapp-acc-operator` Namespace:

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

Antwort:

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

b. Erstellen Sie ein Geheimnis für das `netapp-acc-operator` Namespace. Fügen Sie Docker-Informationen hinzu und führen Sie den folgenden Befehl aus:

> Platzhalter `your_registry_path` Sollte die Position der Bilder, die Sie früher hochgeladen haben, entsprechen (z. B. `[Registry_URL]/netapp/astra/astracc/22.08.1-26`).

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

Beispielantwort:

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

> Wenn Sie den Namespace löschen, nachdem das Geheimnis generiert wurde, müssen Sie das Geheimnis für den Namespace neu generieren, nachdem der Namespace neu erstellt wurde.

c. Erstellen Sie die `netapp-acc` (Oder benutzerdefinierter Name) Namespace

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

Beispielantwort:

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

d. Erstellen Sie ein Geheimnis für das `netapp-acc` (Oder benutzerdefinierter Name) Namespace Fügen Sie Docker-Informationen hinzu und führen Sie den folgenden Befehl aus:

```
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]
```

Antwort

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

a. (Optional) Wenn Sie möchten, dass der Cluster nach der Installation automatisch vom Astra Control Center verwaltet wird, stellen Sie sicher, dass Sie den kubeconfig als Geheimnis innerhalb des Astra Control Center Namespace angeben, in dem Sie diesen Befehl einsetzen möchten:

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

Installieren Sie den Operator Astra Control Center

1. Telefonbuch ändern:

```
cd manifests
```

2. Bearbeiten Sie die YAML-Implementierung des Astra Control Center-Bedieners (`astra_control_center_operator_deploy.yaml`) Zu Ihrem lokalen Register und Geheimnis zu verweisen.

```
vim astra_control_center_operator_deploy.yaml
```

Ein YAML-Beispiel mit Anmerkungen folgt diesen Schritten.

a. Wenn Sie eine Registrierung verwenden, für die eine Authentifizierung erforderlich ist, ersetzen Sie die Standardzeile von `imagePullSecrets: []` Mit folgenden Optionen:

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

b. Ändern `[your_registry_path]` Für das `kube-rbac-proxy` Bild zum Registrierungspfad, in dem Sie die Bilder in ein geschoben haben Vorheriger Schritt.

c. Ändern `[your_registry_path]` Für das `acc-operator-controller-manager` Bild zum Registrierungspfad, in dem Sie die Bilder in ein geschoben haben Vorheriger Schritt.

d. (Für Installationen mit Astra Data Store Vorschau) Siehe dieses bekannte Problem bzgl. "Provisorer der Speicherklasse und zusätzliche Änderungen, die Sie an der YAML vornehmen müssen".

```
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: []
```

3. Installieren Sie den Astra Control Center-Operator:

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

Beispielantwort:

```
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. Überprüfen Sie, ob Pods ausgeführt werden:

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

Konfigurieren Sie Astra Control Center

1. Bearbeiten Sie die Datei Astra Control Center Custom Resource (CR) (`astra_control_center_min.yaml`) Um Konto, AutoSupport, Registrierung und andere notwendige Konfigurationen zu machen:

> `astra_control_center_min.yaml` Ist die Standard-CR und ist für die meisten Installationen geeignet. Machen Sie sich mit allen vertraut "CR-Optionen und ihre potenziellen Werte" Damit Sie Astra Control Center richtig für Ihre Umgebung einsetzen können. Falls für Ihre Umgebung zusätzliche Anpassungen erforderlich sind, können Sie dies verwenden `astra_control_center.yaml` Als Alternative CR.

```
vim astra_control_center_min.yaml
```

> Wenn Sie eine Registrierung verwenden, für die keine Autorisierung erforderlich ist, müssen Sie das löschen `secret` Zeile in `imageRegistry` Oder die Installation schlägt fehl.

   a. Ändern `[your_registry_path]` Zum Registrierungspfad, in dem Sie die Bilder im vorherigen Schritt verschoben haben.

b. Ändern Sie das `accountName` Zeichenfolge an den Namen, den Sie dem Konto zuordnen möchten.

c. Ändern Sie das `astraAddress` Zeichenfolge an den FQDN, den Sie in Ihrem Browser für den Zugriff auf Astra verwenden möchten. Verwenden Sie es nicht `http://` Oder `https://` In der Adresse. Kopieren Sie diesen FQDN zur Verwendung in einem Später Schritt.

d. Ändern Sie das `email` Zeichenfolge zur standardmäßigen ursprünglichen Administratoradresse. Kopieren Sie diese E-Mail-Adresse zur Verwendung in A Später Schritt.

e. Ändern `enrolled` Für AutoSupport bis `false` Für Websites ohne Internetverbindung oder Aufbewahrung `true` Für verbundene Standorte.

f. Wenn Sie einen externen Zertifikaten-Manager verwenden, fügen Sie folgende Zeilen zu hinzu `spec`:

```
spec:
  crds:
    externalCertManager: true
```

g. (Optional) Geben Sie einen Vornamen ein `firstName` Und Nachname `lastName` Des Benutzers, der dem Konto zugeordnet ist. Sie können diesen Schritt jetzt oder später in der Benutzeroberfläche ausführen.

h. (Optional) Ändern Sie den `storageClass` Nutzen Sie bei Bedarf für Ihre Installation einen anderen Trident Storage Class-Mitarbeiter.

i. (Optional) Wenn der Cluster nach der Installation automatisch von Astra Control Center verwaltet werden soll und schon vorhanden ist Schuf das Geheimnis, das den kubeconfig für diesen Cluster enthältGeben Sie den Namen des Geheimnisses an, indem Sie dieser YAML-Datei ein neues Feld hinzufügen `astraKubeConfigSecret: "acc-kubeconfig-cred or custom secret name"`

j. Führen Sie einen der folgenden Schritte aus:

- **Anderer Ingress-Controller (ingressType:Generic)**: Dies ist die Standard-Aktion mit Astra Control Center. Nachdem Astra Control Center bereitgestellt wurde, müssen Sie den Ingress-Controller so konfigurieren, dass Astra Control Center mit einer URL verfügbar ist.

  Die standardmäßige Astra Control Center-Installation stellt das Gateway ein (`service/traefik`) Vom Typ zu sein `ClusterIP`. Bei dieser Standardinstallation müssen Sie zusätzlich einen Kubernetes ProgressController/Ingress einrichten, um den Datenverkehr dorthin zu leiten. Wenn Sie ein Ingress verwenden möchten, lesen Sie "Eindringen für den Lastenausgleich einrichten".

- **Service Load Balancer (ingressType:AccTraefik)**: Wenn Sie keinen IngressController installieren oder eine Ingress-Ressource erstellen möchten, stellen Sie ein `ingressType` Bis `AccTraefik`.

  Dies implementiert das Astra Control Center `traefik` Gateway als Service des Typs Kubernetes Load Balancer:

  Astra Control Center nutzt einen Service vom Typ „loadbalancer“ (`svc/traefik` Im Astra Control Center Namespace) und erfordert, dass ihm eine zugängliche externe IP-Adresse zugewiesen wird. Wenn in Ihrer Umgebung Load Balancer zugelassen sind und Sie noch nicht eine konfiguriert haben, können Sie MetalLB oder einen anderen externen Service Load Balancer verwenden, um dem Dienst eine externe IP-Adresse zuzuweisen. In der Konfiguration des internen DNS-Servers sollten Sie den ausgewählten DNS-Namen für Astra Control Center auf die Load-Balanced IP-Adresse verweisen.

Einzelheiten zum Servicetyp von „loadbalancer“ und Ingress finden Sie unter "Anforderungen".

```
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"
```

Komplette Astra Control Center und Bedienerinstallation

1. Wenn Sie dies in einem vorherigen Schritt nicht bereits getan haben, erstellen Sie das `netapp-acc` (Oder benutzerdefinierter) Namespace:

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

Beispielantwort:

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

2. Installieren Sie das Astra Control Center im `netapp-acc` (Oder Ihr individueller) Namespace:

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

Beispielantwort:

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

Überprüfen Sie den Systemstatus

Wenn Sie OpenShift verwenden möchten, können Sie vergleichbare oc-Befehle für Verifizierungsschritte verwenden.

1. Vergewissern Sie sich, dass alle Systemkomponenten erfolgreich installiert wurden.

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

Jeder Pod sollte einen Status von haben `Running`. Es kann mehrere Minuten dauern, bis die System-Pods implementiert sind.

Beispielantwort

```
NAME                                        READY   STATUS    RESTARTS
AGE
acc-helm-repo-6b44d68d94-d8m55              1/1     Running   0
13m
activity-78f99ddf8-hltct                    1/1     Running   0
10m
api-token-authentication-457nl              1/1     Running   0
9m28s
api-token-authentication-dgwsz              1/1     Running   0
9m28s
api-token-authentication-hmqqc              1/1     Running   0
9m28s
asup-75fd554dc6-m6qzh                       1/1     Running   0
9m38s
authentication-6779b4c85d-92gds             1/1     Running   0
8m11s
bucketservice-7cc767f8f8-lqwr8              1/1     Running   0
9m31s
certificates-549fd5d6cb-5kmd6               1/1     Running   0
9m56s
certificates-549fd5d6cb-bkjh9               1/1     Running   0
9m56s
cloud-extension-7bcb7948b-hn8h2             1/1     Running   0
10m
cloud-insights-service-56ccf86647-fgg69     1/1     Running   0
9m46s
composite-compute-677685b9bb-7vgsf          1/1     Running   0
10m
composite-volume-657d6c5585-dnq79           1/1     Running   0
9m49s
credentials-755fd867c8-vrlmt                1/1     Running   0
11m
entitlement-86495cdf5b-nwhh2                1/1     Running   2
10m
features-5684fb8b56-8d6s8                   1/1     Running   0
10m
fluent-bit-ds-rhx7v                         1/1     Running   0
7m48s
fluent-bit-ds-rjms4                         1/1     Running   0
7m48s
fluent-bit-ds-zf5ph                         1/1     Running   0
7m48s
graphql-server-66d895f544-w6hjd             1/1     Running   0
3m29s
```

```
identity-744df448d5-rlcmm                1/1    Running  0
10m
influxdb2-0                              1/1    Running  0
13m
keycloak-operator-75c965cc54-z7csw       1/1    Running  0
8m16s
krakend-798d6df96f-9z2sk                 1/1    Running  0
3m26s
license-5fb7d75765-f8mjg                 1/1    Running  0
9m50s
login-ui-7d5b7df85d-l2s7s                1/1    Running  0
3m20s
loki-0                                   1/1    Running  0
13m
metrics-facade-599b9d7fcc-gtmgl          1/1    Running  0
9m40s
monitoring-operator-67cc74f844-cdplp     2/2    Running  0
8m11s
nats-0                                   1/1    Running  0
13m
nats-1                                   1/1    Running  0
13m
nats-2                                   1/1    Running  0
12m
nautilus-769f5b74cd-k5jxm                1/1    Running  0
9m42s
nautilus-769f5b74cd-kd9gd                1/1    Running  0
8m59s
openapi-84f6ccd8ff-76kvp                 1/1    Running  0
9m34s
packages-6f59fc67dc-4g2f5                1/1    Running  0
9m52s
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                       1/1    Running  0
8m7s
polaris-keycloak-1                       1/1    Running  0
5m49s
polaris-keycloak-2                       1/1    Running  0
5m15s
polaris-keycloak-db-0                    1/1    Running  0
8m6s
```

```
polaris-keycloak-db-1                     1/1    Running  0
5m49s
polaris-keycloak-db-2                     1/1    Running  0
4m57s
polaris-mongodb-0                         2/2    Running  0
13m
polaris-mongodb-1                         2/2    Running  0
12m
polaris-mongodb-2                         2/2    Running  0
12m
polaris-ui-565f56bf7b-zwr8b               1/1    Running  0
3m19s
polaris-vault-0                           1/1    Running  0
13m
polaris-vault-1                           1/1    Running  0
13m
polaris-vault-2                           1/1    Running  0
13m
public-metrics-6d86d66444-2wbzl           1/1    Running  0
9m30s
storage-backend-metrics-77c5d98dcd-dbhg5 1/1    Running  0
9m44s
storage-provider-78c885f57c-6zcv4         1/1    Running  0
9m36s
telegraf-ds-2l2m9                         1/1    Running  0
7m48s
telegraf-ds-qfzgh                         1/1    Running  0
7m48s
telegraf-ds-shrms                         1/1    Running  0
7m48s
telegraf-rs-bjpkt                         1/1    Running  0
7m48s
telemetry-service-6684696c64-qzfdf        1/1    Running  0
10m
tenancy-6596b6c54d-vmpsm                  1/1    Running  0
10m
traefik-7489dc59f9-6mnst                  1/1    Running  0
3m19s
traefik-7489dc59f9-xrkgg                  1/1    Running  0
3m4s
trident-svc-6c8dc458f5-jswcl              1/1    Running  0
10m
vault-controller-6b954f9b76-gz9nm         1/1    Running  0
11m
```

2. (Optional) um sicherzustellen, dass die Installation abgeschlossen ist, können Sie sich die ansehen `acc-operator` Protokolle mit dem folgenden Befehl

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

> `accHost` Die Cluster-Registrierung ist einer der letzten Vorgänge, und bei Ausfall wird die Implementierung nicht fehlschlagen. Sollte ein Cluster-Registrierungsfehler in den Protokollen gemeldet werden, können Sie die Registrierung erneut durch den Add-Cluster-Workflow versuchen "In der UI" Oder API.

3. Wenn alle Pods ausgeführt werden, überprüfen Sie, ob die Installation erfolgreich war (`READY` Ist `True`) Und holen Sie sich das einmalige Passwort, das Sie verwenden, wenn Sie sich bei Astra Control Center:

```
kubectl get AstraControlCenter -n netapp-acc
```

Antwort:

```
NAME    UUID                                    VERSION     ADDRESS
READY
astra   ACC-9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f  22.08.1-26
10.111.111.111  True
```

> Den UUID-Wert kopieren. Das Passwort lautet `ACC-` Anschließend der UUID-Wert (`ACC-[UUID]` Oder in diesem Beispiel `ACC-9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f`).

Eindringen für den Lastenausgleich einrichten

Sie können einen Kubernetes Ingress-Controller einrichten, der den externen Zugriff auf Services, wie etwa den Lastausgleich in einem Cluster, managt.

Dieses Verfahren erklärt, wie ein Ingress-Controller eingerichtet wird (`ingressType:Generic`). Dies ist die Standardaktion mit Astra Control Center. Nachdem Astra Control Center bereitgestellt wurde, müssen Sie den Ingress-Controller so konfigurieren, dass Astra Control Center mit einer URL verfügbar ist.

> Wenn Sie keinen Ingress-Controller einrichten möchten, können Sie ihn einstellen `ingressType:AccTraefik`). Astra Control Center nutzt einen Service vom Typ „loadbalancer“ (`svc/traefik` Im Astra Control Center Namespace) und erfordert, dass ihm eine zugängliche externe IP-Adresse zugewiesen wird. Wenn in Ihrer Umgebung Load Balancer zugelassen sind und Sie noch nicht eine konfiguriert haben, können Sie MetalLB oder einen anderen externen Service Load Balancer verwenden, um dem Dienst eine externe IP-Adresse zuzuweisen. In der Konfiguration des internen DNS-Servers sollten Sie den ausgewählten DNS-Namen für Astra Control Center auf die Load-Balanced IP-Adresse verweisen. Einzelheiten zum Servicetyp von „loadbalancer“ und Ingress finden Sie unter "Anforderungen".

Die Schritte unterscheiden sich je nach Art des Ingress-Controllers, den Sie verwenden:

- Istio Ingress
- Nginx-Ingress-Controller
- OpenShift-Eingangs-Controller

Was Sie benötigen

- Erforderlich "Eingangs-Controller" Sollte bereits eingesetzt werden.
- Der "Eingangsklasse" Entsprechend der Eingangs-Steuerung sollte bereits erstellt werden.
- Sie verwenden Kubernetes-Versionen zwischen und v1.19 und v1.22.

Schritte für Istio Ingress

1. Konfigurieren Sie Istio Ingress.

   Bei diesem Verfahren wird davon ausgegangen, dass Istio mithilfe des Konfigurationsprofils „Standard“ bereitgestellt wird.

2. Sammeln oder erstellen Sie die gewünschte Zertifikatdatei und die private Schlüsseldatei für das Ingress Gateway.

   Sie können ein CA-signiertes oder selbstsigniertes Zertifikat verwenden. Der allgemeine Name muss die Astra-Adresse (FQDN) sein.

   Beispielbefehl:

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

3. Erstellen Sie ein Geheimnis `tls secret name` Vom Typ `kubernetes.io/tls` Für einen privaten TLS-Schlüssel und ein Zertifikat im `istio-system namespace` Wie in TLS Secrets beschrieben.

   Beispielbefehl:

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

   Der Name des Geheimnisses sollte mit dem übereinstimmen `spec.tls.secretName` Verfügbar in `istio-ingress.yaml` Datei:

4. Bereitstellung einer Ingress-Ressource in `netapp-acc` (Oder Custom-Name) Namespace mit entweder dem v1beta1 (veraltet in Kubernetes Version weniger als oder 1.22) oder v1 Ressourcentyp für entweder ein deprecated oder ein neues Schema:

   Ausgabe:

```
apiVersion: networking.k8s.io/v1beta1
kind: IngressClass
metadata:
  name: istio
spec:
  controller: istio.io/ingress-controller
---
apiVersion: networking.k8s.io/v1beta1
kind: Ingress
metadata:
  name: ingress
  namespace: istio-system
spec:
  ingressClassName: istio
  tls:
  - hosts:
    - <ACC addess>
    secretName: [tls secret name]
  rules:
  - host: [ACC addess]
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          serviceName: traefik
          servicePort: 80
```

Für das neue Schema v1 gehen Sie wie folgt vor:

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

Ausgabe:

```
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: istio-system
spec:
  ingressClassName: istio
  tls:
  - hosts:
    - <ACC addess>
    secretName: [tls secret name]
  rules:
  - host: [ACC addess]
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: traefik
            port:
              number: 80
```

5. Implementieren Sie wie gewohnt Astra Control Center.
6. Überprüfen Sie den Status des Eingangs:

```
kubectl get ingress -n netapp-acc
```

Antwort:

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

Schritte für Nginx Ingress Controller

1. Erstellen Sie ein Geheimnis des Typs[`kubernetes.io/tls`] Für einen privaten TLS-Schlüssel und ein Zertifikat in `netapp-acc` (Oder Custom-Name) Namespace wie in beschrieben "TLS-Geheimnisse".

2. Bereitstellung einer Ingress-Ressource in `netapp-acc` (Oder Custom-Name) Namespace mit entweder dem `v1beta1` (Veraltet in Kubernetes Version kleiner als oder 1.22) oder `v1` Ressourcentyp für ein deprecated oder ein neues Schema:

   a. Für A `v1beta1` Veraltete Schemas, folgen Sie diesem Beispiel:

```
apiVersion: extensions/v1beta1
Kind: IngressClass
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. Für das `v1` Neues Schema, folgen Sie diesem Beispiel:

```
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 addess>
    http:
      paths:
        - path:
          backend:
            service:
              name: traefik
              port:
                number: 80
          pathType: ImplementationSpecific
```

Schritte für OpenShift-Eingangs-Controller

1. Beschaffen Sie Ihr Zertifikat, und holen Sie sich die Schlüssel-, Zertifikat- und CA-Dateien für die OpenShift-Route bereit.
2. Erstellen Sie die 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
```

Melden Sie sich in der UI des Astra Control Center an

Nach der Installation von Astra Control Center ändern Sie das Passwort für den Standardadministrator und melden sich im Astra Control Center UI Dashboard an.

Schritte

1. Geben Sie in einem Browser den FQDN ein, den Sie in verwendet haben `astraAddress` Im `astra_control_center_min.yaml` CR, wenn Sie haben das Astra Control Center installiert.
2. Akzeptieren Sie die selbstsignierten Zertifikate, wenn Sie dazu aufgefordert werden.

Sie können nach der Anmeldung ein benutzerdefiniertes Zertifikat erstellen.

3. Geben Sie auf der Anmeldeseite des Astra Control Center den Wert ein, den Sie für verwendet haben `email` In `astra_control_center_min.yaml` CR, wenn Sie haben das Astra Control Center installiert, Gefolgt von dem Einzeitkennwort (`ACC-[UUID]`).

Wenn Sie dreimal ein falsches Passwort eingeben, wird das Administratorkonto 15 Minuten lang gesperrt.

4. Wählen Sie **Login**.
5. Ändern Sie das Passwort, wenn Sie dazu aufgefordert werden.

Wenn es sich um Ihre erste Anmeldung handelt und Sie das Passwort vergessen haben und noch keine anderen Administratorkonten erstellt wurden, wenden Sie sich an den NetApp Support, um Unterstützung bei der Passwortwiederherstellung zu erhalten.

6. (Optional) Entfernen Sie das vorhandene selbst signierte TLS-Zertifikat und ersetzen Sie es durch ein "Benutzerdefiniertes TLS-Zertifikat, signiert von einer Zertifizierungsstelle (CA)".

Beheben Sie die Fehlerbehebung für die Installation

Wenn einer der Dienstleistungen in ist `Error` Status, können Sie die Protokolle überprüfen. Suchen Sie nach API-Antwortcodes im Bereich von 400 bis 500. Diese geben den Ort an, an dem ein Fehler aufgetreten ist.

Schritte

1. Um die Bedienerprotokolle des Astra Control Center zu überprüfen, geben Sie Folgendes ein:

```
kubectl logs --follow -n netapp-acc-operator $(kubectl get pods -n
netapp-acc-operator -o name) -c manager
```

Wie es weiter geht

Führen Sie die Implementierung durch "Setup-Aufgaben".

=
:allow-uri-read:

Einschränkungen der POD-Sicherheitsrichtlinie verstehen

Astra Control Center unterstützt die Einschränkung von Berechtigungen durch POD-Sicherheitsrichtlinien (PSPs). Mithilfe der POD-Sicherheitsrichtlinien können Sie begrenzen, welche Benutzer oder Gruppen Container ausführen können und welche Berechtigungen diese Container haben können.

Einige Kubernetes Distributionen, wie z. B. RKE2, verfügen über eine Standard-Pod-Sicherheitsrichtlinie, die zu restriktiv ist und bei der Installation von Astra Control Center Probleme verursacht.

Anhand der hier enthaltenen Informationen und Beispiele können Sie die von Astra Control Center erstellten POD-Sicherheitsrichtlinien verstehen und die Richtlinien für die POD-Sicherheit konfigurieren, die den

erforderlichen Schutz bieten, ohne die Funktionen des Astra Control Center zu beeinträchtigen.

PSPs, die vom Astra Control Center installiert werden

Astra Control Center erstellt während der Installation mehrere POD-Sicherheitsrichtlinien. Einige davon sind dauerhaft, und einige von ihnen werden während bestimmter Operationen erstellt und werden entfernt, sobald der Vorgang abgeschlossen ist.

PSPs, die während der Installation erstellt wurden

Bei der Installation von Astra Control Center installiert der Astra Control Center-Operator eine benutzerdefinierte POD-Sicherheitsrichtlinie, ein Rollenobjekt und ein rollenbindendes Objekt, um die Implementierung von Astra Control Center-Diensten im Astra Control Center-Namespace zu unterstützen.

Die neue Richtlinie und die neuen Objekte haben folgende Attribute:

```
kubectl get psp

NAME                            PRIV    CAPS          SELINUX    RUNASUSER
FSGROUP     SUPGROUP    READONLYROOTFS   VOLUMES
avp-psp                         false                 RunAsAny   RunAsAny
RunAsAny    RunAsAny    false            *
netapp-astra-deployment-psp     false                 RunAsAny   RunAsAny
RunAsAny    RunAsAny    false            *

kubectl get role

NAME                                    CREATED AT
netapp-astra-deployment-role            2022-06-27T19:34:58Z

kubectl get rolebinding

NAME                                    ROLE
AGE
netapp-astra-deployment-rb              Role/netapp-astra-deployment-role
32m
```

Während des Backup-Betriebs erstellte PSPs

Astra Control Center erstellt während des Backup-Betriebs eine dynamische Pod-Sicherheitsrichtlinie, ein ClusterRollenobjekt und ein rollenbindendes Objekt. Diese unterstützen den Backup-Prozess, der in einem separaten Namespace geschieht.

Die neue Richtlinie und die neuen Objekte haben folgende Attribute:

```
kubectl get psp

NAME                             PRIV    CAPS
SELINUX    RUNASUSER          FSGROUP     SUPGROUP    READONLYROOTFS
VOLUMES
netapp-astra-backup             false   DAC_READ_SEARCH
RunAsAny   RunAsAny           RunAsAny    RunAsAny    false            *

kubectl get role

NAME                   CREATED AT
netapp-astra-backup   2022-07-21T00:00:00Z

kubectl get rolebinding

NAME                   ROLE                          AGE
netapp-astra-backup   Role/netapp-astra-backup   62s
```

PSPs, die während des Clustermanagements erstellt wurden

Wenn Sie einen Cluster verwalten, installiert Astra Control Center den netapp Monitoring Operator im Managed Cluster. Dieser Operator erstellt eine Pod-Sicherheitsrichtlinie, ein ClusterRole-Objekt und ein RoleBending-Objekt, um Telemetrieservices im Namespace von Astra Control Center bereitzustellen.

Die neue Richtlinie und die neuen Objekte haben folgende Attribute:

```
kubectl get psp

NAME                             PRIV    CAPS
SELINUX    RUNASUSER          FSGROUP     SUPGROUP    READONLYROOTFS
VOLUMES
netapp-monitoring-psp-nkmo     true    AUDIT_WRITE,NET_ADMIN,NET_RAW
RunAsAny   RunAsAny           RunAsAny    RunAsAny    false            *

kubectl get role

NAME                                          CREATED AT
netapp-monitoring-role-privileged             2022-07-21T00:00:00Z

kubectl get rolebinding

NAME                                                ROLE
AGE
netapp-monitoring-role-binding-privileged           Role/netapp-
monitoring-role-privileged             2m5s
```

Aktivieren der Netzwerkkommunikation zwischen Namespaces

Einige Umgebungen verwenden NetworkPolicy-Konstrukte, um den Datenverkehr zwischen Namespaces zu beschränken. Der Astra Control Center Operator, Astra Control Center und das Astra Plugin für VMware vSphere sind allesamt in verschiedenen Namespaces. Die Dienste in diesen verschiedenen Namespaces müssen in der Lage sein, miteinander zu kommunizieren. Gehen Sie wie folgt vor, um diese Kommunikation zu aktivieren.

Schritte

1. Löschen Sie alle im Astra Control Center Namespace vorhandenen NetworkPolicy-Ressourcen:

```
kubectl get networkpolicy -n netapp-acc
```

2. Verwenden Sie für jedes NetworkPolicy-Objekt, das vom vorhergehenden Befehl zurückgegeben wird, den folgenden Befehl, um es zu löschen. Ersetzen Sie <OBJECT_NAME> durch den Namen des zurückgegebenen Objekts:

```
kubectl delete networkpolicy <OBJECT_NAME> -n netapp-acc
```

3. Wenden Sie die folgende Ressourcendatei an, um das ACC-avp-Netzwerk-Policy-Objekt zu konfigurieren, damit das Astra Plugin für VMware vSphere Services Anfragen an die Astra Control Center Services stellen kann. Ersetzen Sie die Informationen in Klammern <> durch Informationen aus Ihrer Umgebung:

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: acc-avp-network-policy
  namespace: <ACC_NAMESPACE_NAME> # REPLACE THIS WITH THE ASTRA CONTROL
CENTER NAMESPACE NAME
spec:
  podSelector: {}
  policyTypes:
    - Ingress
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              kubernetes.io/metadata.name: <PLUGIN_NAMESPACE_NAME> #
REPLACE THIS WITH THE ASTRA PLUGIN FOR VMWARE VSPHERE NAMESPACE NAME
```

4. Wenden Sie die folgende Ressourcendatei an, um das ACC-Operator-Network-Policy-Objekt so zu konfigurieren, dass der Astra Control Center-Operator mit den Astra Control Center-Diensten kommunizieren kann. Ersetzen Sie die Informationen in Klammern <> durch Informationen aus Ihrer Umgebung:

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: acc-operator-network-policy
  namespace: <ACC_NAMESPACE_NAME> # REPLACE THIS WITH THE ASTRA CONTROL
CENTER NAMESPACE NAME
spec:
  podSelector: {}
  policyTypes:
    - Ingress
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              kubernetes.io/metadata.name: <NETAPP-ACC-OPERATOR> #
REPLACE THIS WITH THE OPERATOR NAMESPACE NAME
```

Ressourceneinschränkungen entfernen

In einigen Umgebungen werden die Objekte ResourceQuotas und LimitRanges verwendet, um zu verhindern, dass die Ressourcen in einem Namespace alle verfügbaren CPUs und Speicher im Cluster verbrauchen. Das Astra Control Center stellt keine Höchstgrenzen ein, sodass diese Ressourcen nicht eingehalten werden. Sie müssen sie aus den Namespaces entfernen, in denen Sie Astra Control Center installieren möchten.

Sie können folgende Schritte verwenden, um diese Kontingente und Grenzen abzurufen und zu entfernen. In diesen Beispielen wird die Befehlsausgabe direkt nach dem Befehl angezeigt.

Schritte

1. Holen Sie sich die Ressourcenkontingente im netapp-ACC Namespace:

```
kubectl get quota -n netapp-acc
```

Antwort:

```
NAME          AGE   REQUEST                                              LIMIT
pods-high     16s   requests.cpu: 0/20, requests.memory: 0/100Gi
limits.cpu: 0/200, limits.memory: 0/1000Gi
pods-low      15s   requests.cpu: 0/1, requests.memory: 0/1Gi
limits.cpu: 0/2, limits.memory: 0/2Gi
pods-medium   16s   requests.cpu: 0/10, requests.memory: 0/20Gi
limits.cpu: 0/20, limits.memory: 0/200Gi
```

2. Alle Ressourcen-Kontingente nach Namen löschen:

```
kubectl delete resourcequota  pods-high -n netapp-acc
```

```
kubectl delete resourcequota  pods-low -n netapp-acc
```

```
kubectl delete resourcequota  pods-medium -n netapp-acc
```

3. Grenzbereiche im netapp-ACC Namespace abrufen:

```
kubectl get limits -n netapp-acc
```

Antwort:

```
NAME              CREATED AT
cpu-limit-range   2022-06-27T19:01:23Z
```

4. Grenzwerte nach Namen löschen:

```
kubectl delete limitrange cpu-limit-range -n netapp-acc
```

=
:allow-uri-read:

Copyright-Informationen

Markeninformationen