安装概述

Astra Control Center

NetApp
November 21, 2023

目录

安装概述

选择并完成以下 Astra 控制中心安装过程之一：

- "使用标准流程安装 Astra 控制中心"
- "（如果使用 Red Hat OpenShift ）使用 OpenShift OperatorHub 安装 Astra 控制中心"
- "使用 Cloud Volumes ONTAP 存储后端安装 Astra 控制中心"

使用标准流程安装 Astra 控制中心

要安装 Astra 控制中心，请从 NetApp 支持站点下载安装包，并执行以下步骤在您的环境中安装 Astra 控制中心操作员和 Astra 控制中心。您可以使用此操作步骤在互联网连接或通风环境中安装 Astra 控制中心。

对于 Red Hat OpenShift 环境，您还可以使用 "备用操作步骤" 使用 OpenShift OperatorHub 安装 Astra Control Center 。

您需要的内容

- "开始安装之前，请为 Astra Control Center 部署准备您的环境"。
- 确保所有集群操作员均处于运行状况良好且可用。

  OpenShift 示例：

  ```
  oc get clusteroperators
  ```

- 确保所有 API 服务均处于运行状况良好且可用：

  OpenShift 示例：

  ```
  oc get apiservices
  ```

- 您计划使用的 Astra FQDN 需要可路由到此集群。这意味着您的内部 DNS 服务器中有一个 DNS 条目，或者您正在使用已注册的核心 URL 路由。

关于此任务

Astra 控制中心安装过程将执行以下操作：

- 将 Astra 组件安装到 `NetApp-Accc` （或自定义命名）命名空间中。
- 创建默认帐户。
- 为此 Astra 控制中心实例建立默认管理用户电子邮件地址和默认一次性密码 `Acc-<UID_of_installation>` 。系统会为此用户分配所有者角色，首次登录到 UI 时需要此用户。
- 帮助您确定所有 Astra 控制中心 Pod 是否正在运行。
- 安装 Astra UI 。

(仅限适用场景 Astra数据存储早期访问计划(EAP)版本)如果要使用控制中心管理Astra数据存储并启用VMware工作流、 仅在`pcloud`命名空间上部署Astra控制中心、而不是在`NetApp-Accc`命名空间或本操作步骤 步骤中所述的自定义命名空间上部署。

请勿在整个安装过程中执行以下命令以避免删除所有 Astra 控制中心 Pod： `kubectl delete -f Astra_control_center_operator_deploy.yaml`

如果您使用的是 Red Hat 的 Podman 而不是 Docker 引擎，则可以使用 Podman 命令代替 Docker 命令。

步骤

要安装 Astra 控制中心，请执行以下步骤：

- 下载并解包Astra Control Center软件包
- 安装NetApp Astra kubectl插件
- [将映像添加到本地注册表]
- [为具有身份验证要求的注册表设置命名空间和密钥]
- 安装 Astra 控制中心操作员
- 配置 Astra 控制中心
- 完成 Astra 控制中心和操作员安装
- [验证系统状态]
- [设置传入以进行负载平衡]
- 登录到 Astra 控制中心 UI

下载并解包Astra Control Center软件包

1. 从下载 Astra 控制中心捆绑包（`Astra-control-center-[version].tar.gz`） "NetApp 支持站点"。
2. 从下载 Astra 控制中心证书和密钥的 zip "NetApp 支持站点"。
3. （可选）使用以下命令验证捆绑包的签名：

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

4. 提取映像：

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

安装NetApp Astra kubectl插件

NetApp Astra `kubectl`命令行插件可在执行与部署和升级Astra控制中心相关的常见任务时节省时间。

您需要的内容

NetApp为不同CPU架构和操作系统的插件提供二进制文件。在执行此任务之前、您需要了解您的CPU和操作系统。在Linux和Mac操作系统上、您可以使用`uname -a`命令收集此信息。

步骤

1. 列出可用的NetApp Astra `kubectl`插件二进制文件、并记下操作系统和CPU架构所需的文件名称：

```
ls kubectl-astra/
```

2. 将此文件复制到与标准`kubectl`实用程序相同的位置。在此示例中、`kubectl`实用程序位于/usr/local/bin`目录中。将`<二进制名称>`替换为所需文件的名称：

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

将映像添加到本地注册表

1. 更改为Astra目录：

```
cd acc
```

2. 将 Astra Control Center 映像目录中的文件添加到本地注册表中。

   有关自动加载映像的信息，请参见下面的示例脚本。

   a. 登录到注册表：

      Docker：

      ```
      docker login [your_registry_path]
      ```

      播客：

      ```
      podman login [your_registry_path]
      ```

   b. 使用适当的脚本加载映像，标记映像，并将这些映像推送到本地注册表：

      Docker：

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

播客：

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

为具有身份验证要求的注册表设置命名空间和密钥

1. 如果您使用的注册表需要身份验证，则需要执行以下操作：

   a. 创建 `NetApp-Acc-operator` 命名空间：

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

   响应：

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

   b. 为 `NetApp-Acc-operator` 命名空间创建一个密钥。添加 Docker 信息并运行以下命令：

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

响应示例：

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

c. 创建 `NetApp-Accc` （或自定义命名）命名空间。

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

响应示例：

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

d. 为 `NetApp-Accc` （或自定义命名）命名空间创建一个密钥。添加 Docker 信息并运行以下命令：

```
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]
```

响应

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

a. （可选）如果您希望集群在安装后由 Astra 控制中心自动管理，请确保在您要使用此命令部署到的 Astra 控制中心命名空间中提供 kubeconfig 作为机密：

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

安装 **Astra** 控制中心操作员

1. 编辑 Astra 控制中心操作员部署 YAML （`Astra_control_center_operator_deploy.yaml` ）以参考您的本地注册表和机密。

```
vim astra_control_center_operator_deploy.yaml
```

a. 如果您使用的注册表需要身份验证，请将默认行 `imagePullSecs ： []` 替换为以下内容：

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

b. 将 `Kube-RBAC` 代理 映像的 ` [yor_registry_path]` 更改为将映像推入的注册表路径 上一步。

c. 将 `Acc-operator-controller-manager` 映像的 ` [yor_registry_path]` 更改为在中推送映像的注册表路径 上一步。

d. （对于使用 Astra 数据存储预览版的安装）请参见有关的此已知问题描述 "存储类配置程序以及需要对 YAML 进行的其他更改"。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    control-plane: controller-manager
  name: acc-operator-controller-manager
  namespace: netapp-acc-operator
spec:
  replicas: 1
  selector:
    matchLabels:
      control-plane: controller-manager
  template:
    metadata:
      labels:
        control-plane: controller-manager
    spec:
      containers:
      - args:
        - --secure-listen-address=0.0.0.0:8443
        - --upstream=http://127.0.0.1:8080/
        - --logtostderr=true
        - --v=10
        image: [your_registry_path]/kube-rbac-proxy:v4.8.0
        name: kube-rbac-proxy
        ports:
        - containerPort: 8443
          name: https
      - args:
        - --health-probe-bind-address=:8081
        - --metrics-bind-address=127.0.0.1:8080
        - --leader-elect
        command:
        - /manager
        env:
        - name: ACCOP_LOG_LEVEL
          value: "2"
        image: [your_registry_path]/acc-operator:[version x.y.z]
        imagePullPolicy: IfNotPresent
      imagePullSecrets: []
```

2. 安装 Astra 控制中心操作员：

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

响应示例：

```
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
```

配置 Astra 控制中心

1. 编辑 Astra 控制中心自定义资源（CR）文件（`Astra_control_center_min.yaml`）以进行帐户，AutoSupport，注册表和其他必要配置：

> 如果您的环境需要其他自定义设置，您可以使用 `Astra_control_center.yaml` 作为替代 CR。`Astra_control_center_min.yaml` 是默认 CR，适用于大多数安装。

```
vim astra_control_center_min.yaml
```

> 首次部署 Astra 控制中心后，无法更改 CR 配置的属性。

> 如果您使用的注册表不需要授权，则必须删除 `imageRegistry` 中的 `secret` 行，否则安装将失败。

a. 将 ` [yor_registry_path]` 更改为上一步中用于推送映像的注册表路径。

b. 将 `accountName` 字符串更改为要与帐户关联的名称。

c. 将 `astraAddress` 字符串更改为要在浏览器中使用的 FQDN 以访问 Astra。请勿在此地址中使用 `http：//` 或 `https：//`。复制此 FQDN 以在中使用 后续步骤。

d. 将 `email` 字符串更改为默认的初始管理员地址。复制此电子邮件地址以在中使用 后续步骤。

e. 将 AutoSupport 的 **已注册** 更改为 `false` 对于无 Internet 连接的站点，或者将已连接站点的 `true` 保留。

f. （可选）添加与帐户关联的用户的名字 `firstName` 和姓氏 `lastName` 。您可以在用户界面中立即或稍后执行此步骤。

g. （可选）如果您的安装需要，请将 `storageClass` 值更改为另一个 Trident storageClass 资源。

h. （可选）如果您希望集群在安装后由 Astra 控制中心自动管理，并且您已经这样了 已为此集群创建包含 kubeconfig 的密钥，通过在此 YAML 文件中添加一个名为 `astraKubeConfigSecret` 的新字段来提供此机密的名称： `"Acc-kubeconfig-cred` 或自定义机密名称 `"`

i. 完成以下步骤之一：

- * 其他传入控制器（ ingressType： Generic ）*：这是 Astra 控制中心的默认操作。部署 Astra 控制中心后，您需要配置入口控制器，以便使用 URL 公开 Astra 控制中心。

  默认的 Astra 控制中心安装会将其网关（ `sservice/traefik` ）设置为类型 `ClusterIP` 。此默认安装要求您另外设置一个 Kubernetes IngressController/Ingress ，以便向其路由流量。如果要使用入口，请参见 "设置传入以进行负载平衡"。

- * 服务负载平衡器（ ingressType： AccTraefik ）*：如果您不想安装 IngressController 或创建 Ingress 资源，请将 `ingressType` 设置为 `AccTraefik` 。

  这会将 Astra 控制中心 `traefik` 网关部署为 Kubernetes 负载平衡器类型的服务。

  Astra 控制中心使用类型为 "loadbalancer" 的服务（在 Astra 控制中心命名空间中为 `svC/traefik` ），并要求为其分配可访问的外部 IP 地址。如果您的环境允许使用负载平衡器，但您尚未配置一个平衡器，则可以使用 MetalLB 或其他外部服务负载平衡器为该服务分配外部 IP 地址。在内部 DNS 服务器配置中，您应将为 Astra 控制中心选择的 DNS 名称指向负载平衡的 IP 地址。

  ⓘ 有关 "loadbalancer" 服务类型和入口的详细信息，请参见 "要求"。

```
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"
```

完成 Astra 控制中心和操作员安装

1. 如果您在上一步中尚未创建，请创建 `NetApp-Accc` （或自定义）命名空间：

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

响应示例：

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

2. 在 `NetApp-Accc` （或您的自定义）命名空间中安装 Astra Control Center ：

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

响应示例：

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

验证系统状态

> 如果您更喜欢使用 OpenShift ，则可以使用同等的 oc 命令执行验证步骤。

1. 验证是否已成功安装所有系统组件。

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

每个 POD 的状态应为 `running` 。部署系统 Pod 可能需要几分钟的时间。

响应示例：

```
NAME                                      READY   STATUS    RESTARTS
AGE
acc-helm-repo-5f75c5f564-bzqmt            1/1     Running   0
11m
activity-6b8f7cccb9-mlrn4                 1/1     Running   0
9m2s
api-token-authentication-6hznt            1/1     Running   0
8m50s
api-token-authentication-qpfgb            1/1     Running   0
```

```
8m50s
api-token-authentication-sqnb7             1/1     Running   0
8m50s
asup-5578bbdd57-dxkbp                      1/1     Running   0
9m3s
authentication-56bff4f95d-mspmq            1/1     Running   0
7m31s
bucketservice-6f7968b95d-9rrrl             1/1     Running   0
8m36s
cert-manager-5f6cf4bc4b-82khn              1/1     Running   0
6m19s
cert-manager-cainjector-76cf976458-sdrbc   1/1     Running   0
6m19s
cert-manager-webhook-5b7896bfd8-2n45j      1/1     Running   0
6m19s
cloud-extension-749d9f684c-8bdhq           1/1     Running   0
9m6s
cloud-insights-service-7d58687d9-h5tzw     1/1     Running   2
8m56s
composite-compute-968c79cb5-nv7l4          1/1     Running   0
9m11s
composite-volume-7687569985-jg9gg          1/1     Running   0
8m33s
credentials-5c9b75f4d6-nx9cz               1/1     Running   0
8m42s
entitlement-6c96fd8b78-zt7f8               1/1     Running   0
8m28s
features-5f7bfc9f68-gsjnl                  1/1     Running   0
8m57s
fluent-bit-ds-h88p7                        1/1     Running   0
7m22s
fluent-bit-ds-krhnj                        1/1     Running   0
7m23s
fluent-bit-ds-l5bjj                        1/1     Running   0
7m22s
fluent-bit-ds-lrclb                        1/1     Running   0
7m23s
fluent-bit-ds-s5t4n                        1/1     Running   0
7m23s
fluent-bit-ds-zpr6v                        1/1     Running   0
7m22s
graphql-server-5f5976f4bd-vbb4z            1/1     Running   0
7m13s
identity-56f78b8f9f-8h9p9                  1/1     Running   0
8m29s
influxdb2-0                                1/1     Running   0
```

```
11m
krakend-6f8d995b4d-5khkl                    1/1     Running   0
7m7s
license-5b5db87c97-jmxzc                    1/1     Running   0
9m
login-ui-57b57c74b8-6xtv7                   1/1     Running   0
7m10s
loki-0                                      1/1     Running   0
11m
monitoring-operator-9dbc9c76d-8znck         2/2     Running   0
7m33s
nats-0                                      1/1     Running   0
11m
nats-1                                      1/1     Running   0
10m
nats-2                                      1/1     Running   0
10m
nautilus-6b9d88bc86-h8kfb                   1/1     Running   0
8m6s
nautilus-6b9d88bc86-vn68r                   1/1     Running   0
8m35s
openapi-b87d77dd8-5dz9h                     1/1     Running   0
9m7s
polaris-consul-consul-5ljfb                 1/1     Running   0
11m
polaris-consul-consul-s5d5z                 1/1     Running   0
11m
polaris-consul-consul-server-0              1/1     Running   0
11m
polaris-consul-consul-server-1              1/1     Running   0
11m
polaris-consul-consul-server-2              1/1     Running   0
11m
polaris-consul-consul-twmpq                 1/1     Running   0
11m
polaris-mongodb-0                           2/2     Running   0
11m
polaris-mongodb-1                           2/2     Running   0
10m
polaris-mongodb-2                           2/2     Running   0
10m
polaris-ui-84dc87847f-zrg8w                 1/1     Running   0
7m12s
polaris-vault-0                             1/1     Running   0
11m
polaris-vault-1                             1/1     Running   0
```

```
11m
polaris-vault-2                              1/1     Running   0
11m
public-metrics-657698b66f-67pgt              1/1     Running   0
8m47s
storage-backend-metrics-6848b9fd87-w7x8r     1/1     Running   0
8m39s
storage-provider-5ff5868cd5-r9hj7            1/1     Running   0
8m45s
telegraf-ds-dw4hg                            1/1     Running   0
7m23s
telegraf-ds-k92gn                            1/1     Running   0
7m23s
telegraf-ds-mmxjl                            1/1     Running   0
7m23s
telegraf-ds-nhs8s                            1/1     Running   0
7m23s
telegraf-ds-rj7lw                            1/1     Running   0
7m23s
telegraf-ds-tqrkb                            1/1     Running   0
7m23s
telegraf-rs-9mwgj                            1/1     Running   0
7m23s
telemetry-service-56c49d689b-ffrzx           1/1     Running   0
8m42s
tenancy-767c77fb9d-g9ctv                     1/1     Running   0
8m52s
traefik-5857d87f85-7pmx8                     1/1     Running   0
6m49s
traefik-5857d87f85-cpxgv                     1/1     Running   0
5m34s
traefik-5857d87f85-lvmlb                     1/1     Running   0
4m33s
traefik-5857d87f85-t2xlk                     1/1     Running   0
4m33s
traefik-5857d87f85-v9wpf                     1/1     Running   0
7m3s
trident-svc-595f84dd78-zb8l6                 1/1     Running   0
8m54s
vault-controller-86c94fbf4f-krttq            1/1     Running   0
9m24s
```

2. （可选）为确保安装完成，您可以使用以下命令查看 `Acc-operator` 日志。

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

> AccHost 集群注册是最后一项操作，如果失败，发生原因 部署不会失败。如果日志中指示集群注册失败，您可以通过添加集群工作流再次尝试注册 "在 UI 中" 或 API 。

3. 当所有 Pod 运行时，通过检索 Astra 控制中心操作员安装的 AstraControlCenter 实例来验证安装是否成功。

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

4. 在 YAML 中，` 响应中的 status.deploymentState 字段以查看 `Deploy 值。如果部署失败，则会显示一条错误消息。
5. 要获取登录到 Astra 控制中心时要使用的一次性密码，请复制 status.uuid 值。密码为 Acc- ，后跟 UUID 值（Acc-UUID 或在此示例中为 Acc-9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f ）。

YAML 详细信息示例

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

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

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

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

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

设置传入以进行负载平衡

您可以设置 Kubernetes 入口控制器，用于管理对服务的外部访问，例如集群中的负载平衡。

此操作步骤 介绍了如何设置入口控制器（`ingressType ： Generic`）。这是 Astra 控制中心的默认操作。部署 Astra 控制中心后，您需要配置入口控制器，以便使用 URL 公开 Astra 控制中心。

> 如果您不想设置入口控制器，可以设置 `ingressType ： AccTraefik ）` 。Astra 控制中心使用类型为 "loadbalancer" 的服务（在 Astra 控制中心命名空间中为 `svC/traefik ）`，并要求为其分配可访问的外部 IP 地址。如果您的环境允许使用负载平衡器，但您尚未配置一个平衡器，则可以使用 MetalLB 或其他外部服务负载平衡器为该服务分配外部 IP 地址。在内部 DNS 服务器配置中，您应将为 Astra 控制中心选择的 DNS 名称指向负载平衡的 IP 地址。有关 "loadbalancer" 服务类型和入口的详细信息，请参见 "要求"。

根据您使用的入口控制器类型，步骤会有所不同：

- nginx 入口控制器
- OpenShift 入口控制器

您需要的内容

- 所需 "入口控制器" 应已部署。
- 。 "入口类" 应已创建与入口控制器对应的。
- 您使用的是介于 v1.19 和 v1.22 之间的 Kubernetes 版本，包括 v1.19 和 v1.22 。

nginx 入口控制器的步骤

1. 创建类型的密钥 "8a637503539b25b68130b6e8003579d9" 用于 `NetApp-Accc` （或自定义命名）命名空间中的 TLS 专用密钥和证书，如中所述 "TLS 密钥"。
2. 使用`v1beta1`(在Kubernetes版本低于或1.22的情况下已弃用)或`v1`资源类型为已弃用或新模式在`NetApp-Accc`(或自定义命名)命名空间中部署入站资源：
   a. 对于 `v1bea1` 已弃用的架构，请遵循以下示例：

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

b. 对于 `v1` 新架构，请遵循以下示例：

```
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
```

OpenShift 入口控制器的步骤

1. 获取证书并获取密钥，证书和 CA 文件，以供 OpenShift 路由使用。
2. 创建 OpenShift 路由：

```
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
```

登录到 **Astra** 控制中心 **UI**

安装 Astra 控制中心后，您将更改默认管理员的密码并登录到 Astra 控制中心 UI 信息板。

步骤

1. 在浏览器中，输入在 `Astra_control_center_min.YAML` CR when 的 `AstraAddress` 中使用的 FQDN 您安装了 Astra 控制中心。
2. 出现提示时接受自签名证书。

> 您可以在登录后创建自定义证书。

3. 在 Astra Control Center 登录页面上，在 `Astra_control_center_min.yaml` CR when 中输入您用于 `email` 的值 您安装了 Astra 控制中心，后跟一次性密码（`Acc-UUID`）。

> 如果您输入的密码三次不正确，管理员帐户将锁定 15 分钟。

4. 选择 * 登录 * 。
5. 根据提示更改密码。

> 如果您是首次登录，但忘记了密码，并且尚未创建任何其他管理用户帐户，请联系 NetApp 支持部门以获得密码恢复帮助。

6. （可选）删除现有自签名 TLS 证书并将其替换为 "由证书颁发机构（ CA ）签名的自定义 TLS 证书"。

对安装进行故障排除

如果任何服务处于 `Error` 状态，您可以检查日志。查找 400 到 500 范围内的 API 响应代码。这些信息表示发生故障的位置。

步骤

1. 要检查 Astra 控制中心操作员日志，请输入以下内容：

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

下一步行动

执行以完成部署 "设置任务"。

使用 OpenShift OperatorHub 安装 Astra 控制中心

如果您使用的是 Red Hat OpenShift ，则可以使用 Red Hat 认证操作员安装 Astra Control Center 。使用此操作步骤从安装 Astra 控制中心 "Red Hat 生态系统目录" 或使用 Red Hat OpenShift 容器平台。

完成此操作步骤后，您必须返回到安装操作步骤以完成 "剩余步骤" 以验证安装是否成功并登录。

您需要的内容

- "开始安装之前，请为 Astra Control Center 部署准备您的环境"。
- 在 OpenShift 集群中，确保所有集群操作员均处于运行状况良好的状态（`Available` is `true` ）：

```
oc get clusteroperators
```

- 在 OpenShift 集群中，确保所有 API 服务均处于运行状况良好的状态（`Available` is `true` ）：

```
oc get apiservices
```

- 您已在数据中心为 Astra 控制中心创建 FQDN 地址。
- 您拥有对 Red Hat OpenShift 容器平台执行所述安装步骤所需的权限和访问权限。

步骤

- 下载并解包Astra Control Center软件包
- 安装NetApp Astra kubectl插件
- [将映像添加到本地注册表]
- [找到操作员安装页面]
- [安装操作员]
- 安装 Astra 控制中心

下载并解包Astra Control Center软件包

1. 从下载 Astra 控制中心捆绑包（`Astra-control-center-[version].tar.gz` ） "NetApp 支持站点"。
2. 从下载 Astra 控制中心证书和密钥的 zip "NetApp 支持站点"。
3. （可选）使用以下命令验证捆绑包的签名：

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

4. 提取映像：

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

安装NetApp Astra kubectl插件

NetApp Astra `kubectl`命令行插件可在执行与部署和升级Astra控制中心相关的常见任务时节省时间。

您需要的内容

NetApp为不同CPU架构和操作系统的插件提供二进制文件。在执行此任务之前、您需要了解您的CPU和操作系统。在Linux和Mac操作系统上、您可以使用`uname -a`命令收集此信息。

步骤

1. 列出可用的NetApp Astra `kubectl`插件二进制文件、并记下操作系统和CPU架构所需的文件名称：

```
ls kubectl-astra/
```

2. 将此文件复制到与标准`kubectl`实用程序相同的位置。在此示例中、`kubectl`实用程序位于/usr/local/bin`目录中。将`<二进制名称>`替换为所需文件的名称：

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

将映像添加到本地注册表

1. 更改为Astra目录：

```
cd acc
```

2. 将 Astra Control Center 映像目录中的文件添加到本地注册表中。

> 有关自动加载映像的信息，请参见下面的示例脚本。

a. 登录到注册表：

Docker：

```
docker login [your_registry_path]
```

播客：

```
podman login [your_registry_path]
```

b. 使用适当的脚本加载映像，标记映像，并将这些映像推送到本地注册表：

Docker：

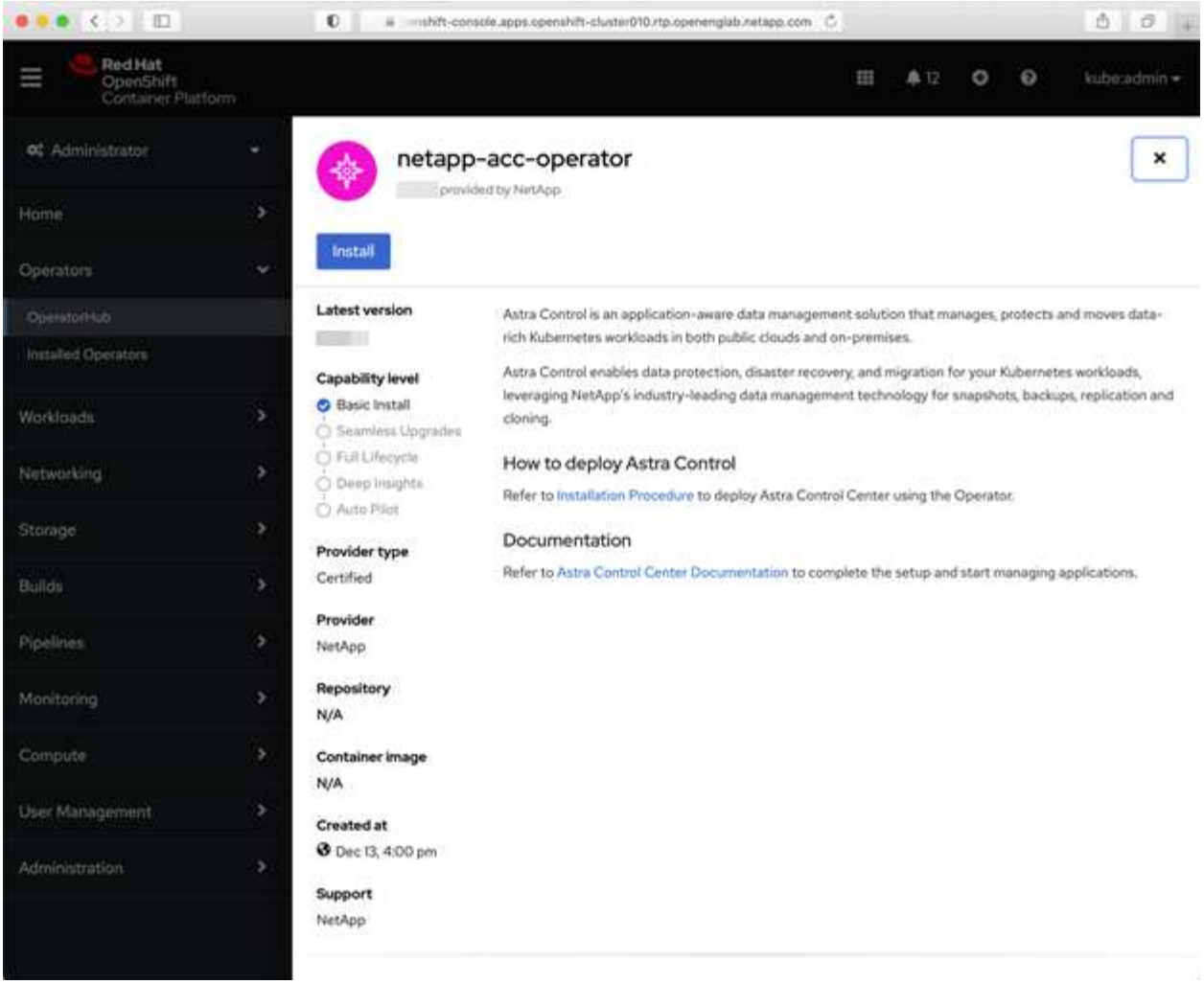
```
export REGISTRY=[Docker_registry_path]
for astraImageFile in $(ls images/*.tar) ; do
  # Load to local cache. And store the name of the loaded image
trimming the 'Loaded images: '
  astraImage=$(docker load --input ${astraImageFile} | sed 's/Loaded
image: //')
  astraImage=$(echo ${astraImage} |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

播客：

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

找到操作员安装页面

1. 要访问操作员安装页面，请完成以下过程之一：
   - 从 Red Hat OpenShift Web 控制台



：

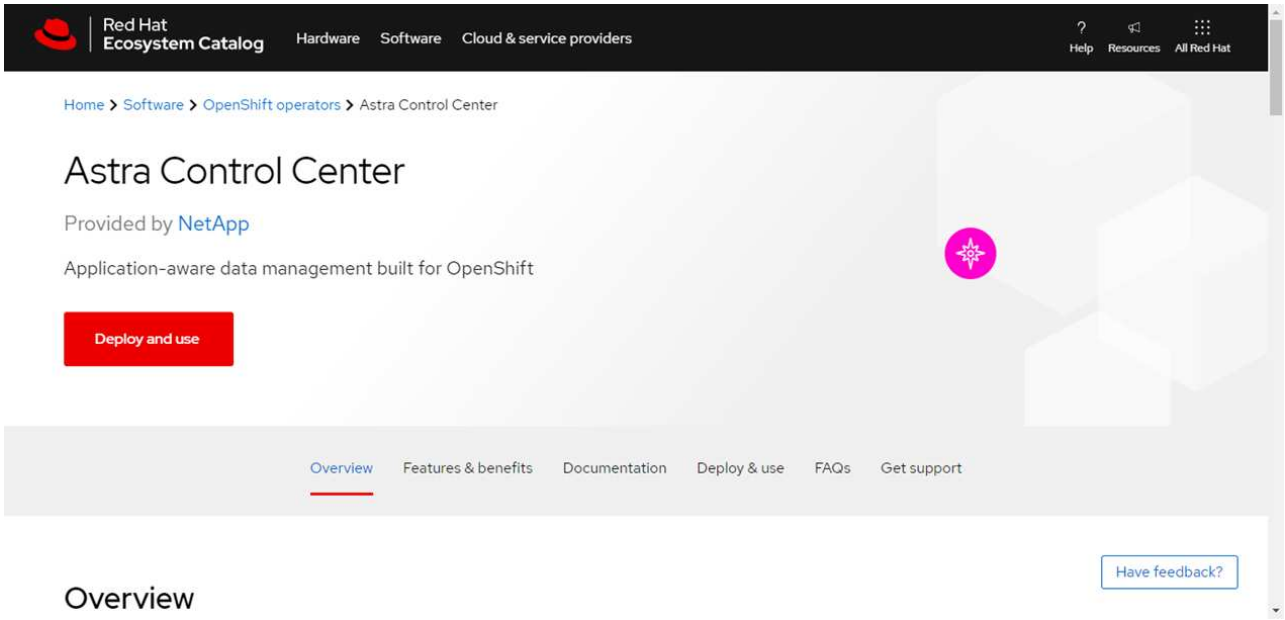
i. 登录到 OpenShift 容器平台 UI 。

ii. 从侧面菜单中，选择 * 运算符 > OperatorHub * 。

iii. 选择 NetApp Astra Control Center 操作员。

iv. 选择 * 安装 * 。

- 从 Red Hat 生态系统目录
：



i. 选择 NetApp Astra 控制中心 "运算符"。

ii. 选择 * 部署并使用 * 。

安装操作员

1. 完成 * 安装操作员 * 页面并安装操作员：

   > 操作员将在所有集群命名空间中可用。

   a. 选择运算符命名空间或 `netapp-ac-operator` namespace will be created automatically as part of the operator install.

   b. 选择手动或自动批准策略。

   > 建议手动批准。每个集群只能运行一个操作员实例。

   c. 选择 * 安装 * 。

   > 如果您选择了手动批准策略，系统将提示您批准此操作员的手动安装计划。

2. 从控制台中，转到 OperatorHub 菜单并确认操作员已成功安装。

安装 Astra 控制中心

1. 在 Astra 控制中心操作员的详细信息视图的控制台中，在提供的 API 部分中选择 `Create instance` 。
2. 填写 `Create AstraControlCenter` Form 字段：

   a. 保留或调整 Astra 控制中心名称。

   b. （可选）启用或禁用自动支持。建议保留自动支持功能。

   c. 输入 Astra 控制中心地址。请勿在此地址中输入 `http ： //` 或 `https ： //` 。

d. 输入 Astra 控制中心版本；例如 21.12.60 。
e. 输入帐户名称，电子邮件地址和管理员姓氏。
f. 保留默认卷回收策略。
g. 在 * 映像注册表 * 中，输入本地容器映像注册表路径。请勿在此地址中输入 `http ： //` 或 `https ： //` 。
h. 如果您使用的注册表需要身份验证，请输入密钥。
i. 输入管理员的名字。
j. 配置资源扩展。
k. 保留默认存储类。
l. 定义 CRD 处理首选项。

3. 选择 `Create` 。

下一步行动

验证是否已成功安装 Astra 控制中心并完成 "剩余步骤" 登录。此外，您还可以通过执行来完成部署 "设置任务"。

使用 Cloud Volumes ONTAP 存储后端安装 Astra 控制中心

借助 Astra 控制中心，您可以使用自管理的 Kubernetes 集群和 Cloud Volumes ONTAP 实例在混合云环境中管理应用程序。您可以在内部 Kubernetes 集群或云环境中的一个自管理 Kubernetes 集群中部署 Astra Control Center 。

在其中一种部署中，您可以使用 Cloud Volumes ONTAP 作为存储后端来执行应用程序数据管理操作。您还可以将 S3 存储分段配置为备份目标。

要在 Amazon Web Services （AWS）和 Microsoft Azure 中使用 Cloud Volumes ONTAP 存储后端安装 Astra 控制中心，请根据您的云环境执行以下步骤。

- 在 Amazon Web Services 中部署 Astra 控制中心
- 在 Microsoft Azure 中部署 Astra 控制中心

在 Amazon Web Services 中部署 Astra 控制中心

您可以在 Amazon Web Services （AWS）公有 云上托管的自管理 Kubernetes 集群上部署 Astra 控制中心。

部署 Astra 控制中心仅支持自管理 OpenShift 容器平台（OCP）集群。

AWS所需的功能

在 AWS 中部署 Astra 控制中心之前，您需要满足以下条件：

- Astra Control Center 许可证。请参见 "Astra 控制中心许可要求"。
- "满足 Astra 控制中心的要求"。

- NetApp Cloud Central account
- Red Hat OpenShift Container Platform （ OCP ）权限（在命名空间级别用于创建 Pod ）
- AWS 凭据，访问 ID 和机密密钥，具有用于创建存储分段和连接器的权限
- AWS 帐户弹性容器注册（ Elastic Container Registry ， ECR ）访问和登录
- 要访问 Astra Control UI ，需要 AWS 托管分区和 Route 53 条目

AWS 的操作环境要求

Astra 控制中心需要以下 AWS 操作环境：

- Red Hat OpenShift 容器平台 4.8

> 确保您选择托管 Astra 控制中心的操作环境符合环境官方文档中概述的基本资源要求。

除了环境的资源要求之外， Astra 控制中心还需要以下资源：

组件	要求
后端 **NetApp Cloud Volumes ONTAP** 存储容量	至少 300 GB 可用
工作节点（**AWS EC2** 要求）	总共至少 3 个辅助节点，每个节点有 4 个 vCPU 核心和 12 GB RAM
负载平衡器	服务类型 "loadbalancer" 可用于将传入流量发送到操作环境集群中的服务
FQDN	一种将 Astra 控制中心的 FQDN 指向负载平衡 IP 地址的方法
Astra Trident （在 **NetApp Cloud Manager** 中发现 **Kubernetes** 集群时安装）	安装并配置了 Astra Trident 21.04 或更高版本，并将 NetApp ONTAP 9.5 或更高版本作为存储后端
映像注册表	您必须拥有一个现有的私有注册表，例如 AWS 弹性容器注册表，您可以将 Astra Control Center 构建映像推送到该注册表。您需要提供要将映像上传到的映像注册表的 URL 。 Astra 控制中心托管的集群和受管集群必须能够访问同一映像注册表，才能使用基于 Restic 的映像备份和还原应用程序。

组件	要求
Astra Trident / ONTAP 配置	Astra 控制中心要求创建一个存储类并将其设置为默认存储类。Astra 控制中心支持以下 ONTAP Kubernetes 存储类，这些存储类是在将 Kubernetes 集群导入到 NetApp Cloud Manager 中时创建的。这些功能由 Astra Trident 提供： • `vsaworkingenvironment-<>-ha-nas csi.trident.netapp.io` • `vsaworkingenvironment-<>-ha-san csi.trident.netapp.io` • `vsaworkingenvironment-<>-Singal-NAS csi.trident.netapp.io` • `vsaworkingenvironment-<>-Singon-san csi.trident.netapp.io`

这些要求假定 Astra 控制中心是运行环境中唯一运行的应用程序。如果环境运行的是其他应用程序，请相应地调整这些最低要求。

AWS 注册表令牌将在 12 小时后过期，之后您必须续订 Docker 映像注册表密钥。

AWS 部署概述

下面简要介绍了将 Cloud Volumes ONTAP 作为存储后端安装适用于 AWS 的 Astra 控制中心的过程。

下面详细介绍了其中每个步骤。

1. 确保您具有足够的 IAM 权限。
2. 在 AWS 上安装 RedHat OpenShift 集群。
3. 配置 AWS。
4. 配置 NetApp Cloud Manager。
5. 安装 Astra 控制中心。

确保您具有足够的 **IAM** 权限

确保您具有足够的IAM角色和权限、可以安装RedHat OpenShift集群和NetApp Cloud Manager Connector。

请参见 "初始 AWS 凭据"。

在 **AWS** 上安装 **RedHat OpenShift** 集群

在 AWS 上安装 RedHat OpenShift 容器平台集群。

有关安装说明，请参见 "在 OpenShift 容器平台中的 AWS 上安装集群"。

配置 AWS

接下来、将AWS配置为创建虚拟网络、设置EC2计算实例、创建AWS S3存储分段、创建弹性容器注册表(ECR)以托管Astra控制中心映像、并将这些映像推送到此注册表。

按照 AWS 文档完成以下步骤。请参见 "AWS 安装文档"。

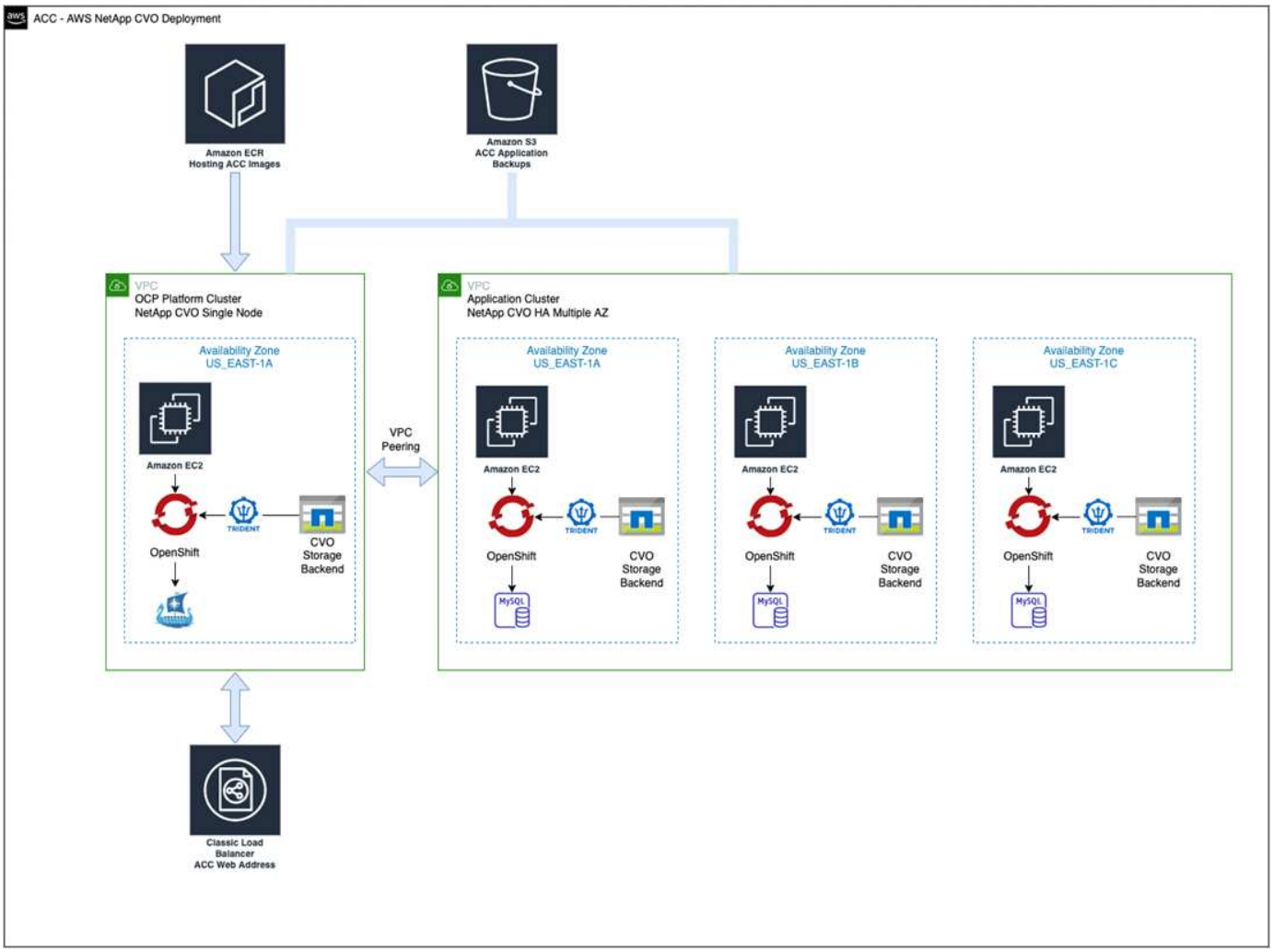
1. 创建AWS虚拟网络。
2. 查看 EC2 计算实例。这可以是 AWS 中的裸机服务器或 VM 。
3. 如果实例类型尚未与主节点和工作节点的 Astra 最低资源要求匹配，请更改 AWS 中的实例类型以满足 Astra 要求。 请参见 "Astra 控制中心要求"。
4. 至少创建一个 AWS S3 存储分段来存储备份。
5. 创建 AWS 弹性容器注册表（ ECR ）以托管所有 AccR 映像。

> 如果不创建ECR、则Astra控制中心无法从包含Cloud Volumes ONTAP 且具有AWS后端的集群访问监控数据。如果您尝试使用 Astra 控制中心发现和管理的集群没有 AWS ECR 访问权限，则会导致出现问题描述 。

6. 将这些 Accc 映像推送到您定义的注册表。

> AWS 弹性容器注册表（ ECR ）令牌将在 12 小时后过期，并导致跨集群克隆操作失败。从为AWS配置的Cloud Volumes ONTAP 管理存储后端时会发生此问题描述。要更正此问题描述，请再次向 ECR 进行身份验证，并生成一个新密钥，以便成功恢复克隆操作。

以下是 AWS 部署示例：



配置 **NetApp Cloud Manager**

使用 Cloud Manager 创建工作空间，向 AWS 添加连接器，创建工作环境并导入集群。

按照 Cloud Manager 文档完成以下步骤。请参见以下内容：

- "AWS 中的 Cloud Volumes ONTAP 入门"。
- "使用 Cloud Manager 在 AWS 中创建连接器"

步骤

1. 将凭据添加到 Cloud Manager 。
2. 创建工作空间。
3. 为 AWS 添加连接器。选择 AWS 作为提供程序。
4. 为您的云环境创建一个工作环境。
   a. 位置： "Amazon Web Services （AWS）"
   b. 类型： Cloud Volumes ONTAP HA
5. 导入 OpenShift 集群。集群将连接到您刚刚创建的工作环境。
   a. 选择 * K8s* > * 集群列表 * > * 集群详细信息 * ，查看 NetApp 集群详细信息。

b. 在右上角，记下 Trident 版本。

c. 记下显示 NetApp 作为配置程序的 Cloud Volumes ONTAP 集群存储类。

此操作将导入 Red Hat OpenShift 集群并为其分配默认存储类。您可以选择存储类。Trident 会在导入和发现过程中自动安装。

6. 记下此Cloud Volumes ONTAP 部署中的所有永久性卷和卷。

> Cloud Volumes ONTAP 可以作为单个节点运行，也可以在高可用性环境下运行。如果已启用 HA ，请记下在 AWS 中运行的 HA 状态和节点部署状态。

安装 **Astra** 控制中心

请遵循标准 "Astra 控制中心安装说明"。

在 **Microsoft Azure** 中部署 **Astra** 控制中心

您可以在 Microsoft Azure 公有 云上托管的自管理 Kubernetes 集群上部署 Astra 控制中心。

Azure所需的功能

在 Azure 中部署 Astra 控制中心之前，您需要满足以下条件：

- Astra Control Center 许可证。请参见 "Astra 控制中心许可要求"。
- "满足 Astra 控制中心的要求"。
- NetApp Cloud Central account
- Red Hat OpenShift 容器平台（ OCP ） 4.8
- Red Hat OpenShift Container Platform （ OCP ）权限（在命名空间级别用于创建 Pod ）
- 具有用于创建存储分段和连接器的权限的 Azure 凭据

Azure 的操作环境要求

确保您选择托管 Astra 控制中心的操作环境符合环境官方文档中概述的基本资源要求。

除了环境的资源要求之外， Astra 控制中心还需要以下资源：

请参见 "Astra 控制中心运营环境要求"。

组件	要求
后端 **NetApp Cloud Volumes ONTAP** 存储容量	至少 300 GB 可用
员工节点（ **Azure** 计算要求）	总共至少 3 个辅助节点，每个节点有 4 个 vCPU 核心和 12 GB RAM
负载平衡器	服务类型 "loadbalancer" 可用于将传入流量发送到操作环境集群中的服务

组件	要求
FQDN （ **Azure DNS** 区域）	一种将 Astra 控制中心的 FQDN 指向负载平衡 IP 地址的方法
Astra Trident （在 **NetApp Cloud Manager** 中发现 **Kubernetes** 集群时安装）	安装和配置的 Astra Trident 21.04 或更高版本以及 NetApp ONTAP 9.5 或更高版本将用作存储后端
映像注册表	您必须具有一个现有的专用注册表，例如 Azure 容器注册表（ACR），您可以将 Astra Control Center 构建映像推送到该注册表。您需要提供要将映像上传到的映像注册表的 URL 。 您需要启用匿名访问以提取要备份的 Restic 映像。
Astra Trident / ONTAP 配置	Astra 控制中心要求创建一个存储类并将其设置为默认存储类。Astra 控制中心支持以下 ONTAP Kubernetes 存储类，这些存储类是在将 Kubernetes 集群导入到 NetApp Cloud Manager 中时创建的。这些功能由 Astra Trident 提供： • `vsaworkingenvironment-<>-ha-nas csi.trident.netapp.io` • `vsaworkingenvironment-<>-ha-san csi.trident.netapp.io` • `vsaworkingenvironment-<>-Singal-NAS csi.trident.netapp.io` • `vsaworkingenvironment-<>-Singon-san csi.trident.netapp.io`

这些要求假定 Astra 控制中心是运行环境中唯一运行的应用程序。如果环境运行的是其他应用程序，请相应地调整这些最低要求。

Azure 部署概述

下面简要介绍了适用于 Azure 的 Astra 控制中心的安装过程。

下面详细介绍了其中每个步骤。

1. 在 Azure 上安装 RedHat OpenShift 集群。
2. 创建 Azure 资源组。
3. 确保您具有足够的 IAM 权限。
4. 配置 Azure。
5. 配置 NetApp Cloud Manager。
6. 安装和配置 Astra 控制中心。

在 **Azure** 上安装 **RedHat OpenShift** 集群

第一步是在 Azure 上安装 RedHat OpenShift 集群。

有关安装说明、请参见上的RedHat文档 "在Azure上安装OpenShift集群" 和 "安装Azure帐户"。

创建 **Azure** 资源组

至少创建一个 Azure 资源组。

> OpenShift 可能会创建自己的资源组。除了这些之外，您还应定义 Azure 资源组。请参见 OpenShift 文档。

您可能需要创建平台集群资源组和目标应用程序 OpenShift 集群资源组。

确保您具有足够的 **IAM** 权限

确保您具有足够的IAM角色和权限、可以安装RedHat OpenShift集群和NetApp Cloud Manager Connector。

请参见 "Azure 凭据和权限"。

配置 **Azure**

接下来、将Azure配置为创建虚拟网络、设置计算实例、创建Azure Blob容器、创建Azure容器注册表(ACR)以托管Astra控制中心映像、并将这些映像推送到此注册表。

按照 Azure 文档完成以下步骤。请参见 "在 Azure 上安装 OpenShift 集群"。

1. 创建Azure虚拟网络。
2. 查看计算实例。这可以是 Azure 中的裸机服务器或 VM 。
3. 如果实例类型尚未与主节点和工作节点的 Astra 最低资源要求匹配，请在 Azure 中更改实例类型以满足 Astra 要求。 请参见 "Astra 控制中心要求"。
4. 至少创建一个Azure Blob容器以存储备份。
5. 创建存储帐户。您需要一个存储帐户来创建要用作 Astra 控制中心分段的容器。
6. 创建存储分段访问所需的密钥。
7. 创建 Azure 容器注册表（ACR ）以托管所有 Astra 控制中心映像。
8. 为 Docker 推送 / 拉所有 Astra 控制中心映像设置 ACR 访问。
9. 输入以下脚本，将 Accc 映像推送到此注册表：

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

   - 示例 * ：

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

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

10. 设置 DNS 区域。

配置 **NetApp Cloud Manager**

使用 Cloud Manager 创建工作空间，向 Azure 添加连接器，创建工作环境并导入集群。

按照 Cloud Manager 文档完成以下步骤。请参见 "Azure 中的 Cloud Manager 入门"。

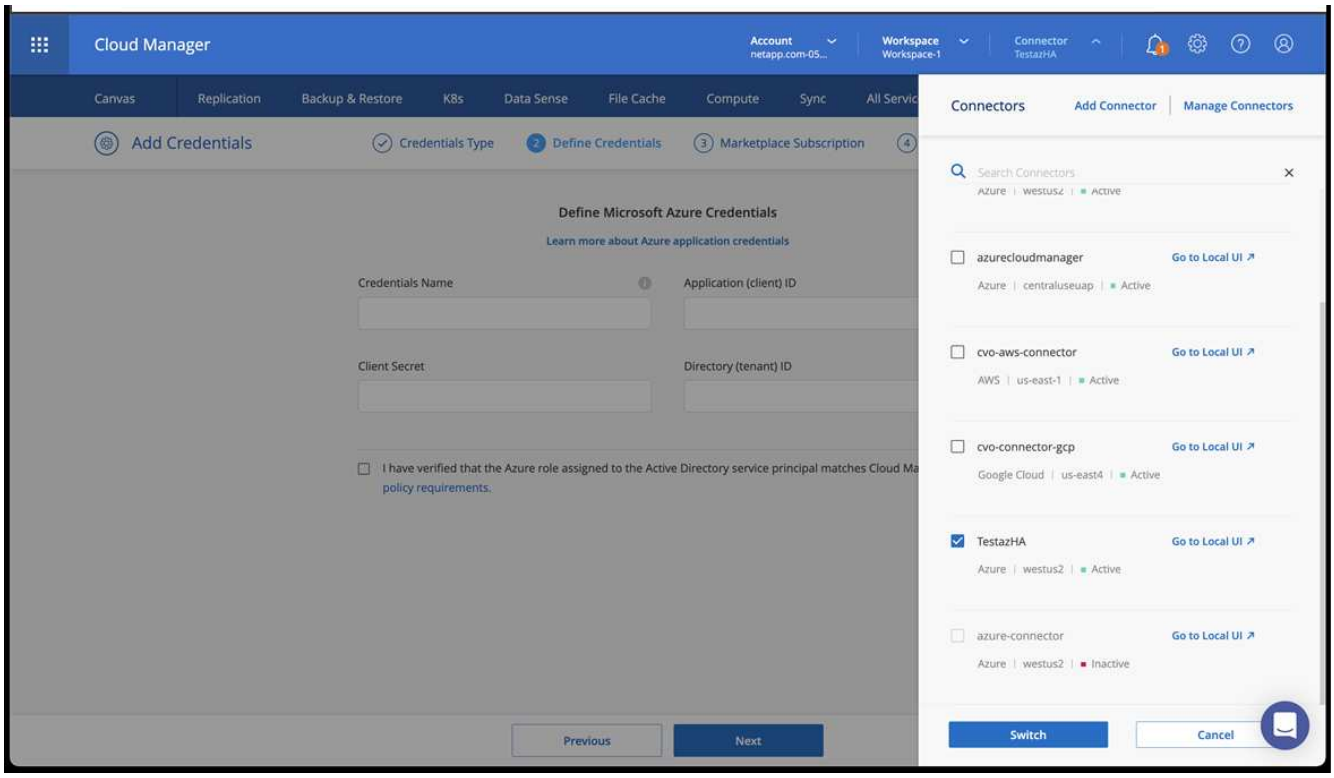
您需要的内容

使用所需的 IAM 权限和角色访问 Azure 帐户

步骤

1. 将凭据添加到 Cloud Manager 。
2. 添加适用于 Azure 的连接器。请参见 "Cloud Manager 策略"。
   a. 选择 * Azure * 作为提供程序。
   b. 输入 Azure 凭据，包括应用程序 ID ，客户端密钥和目录（租户） ID 。

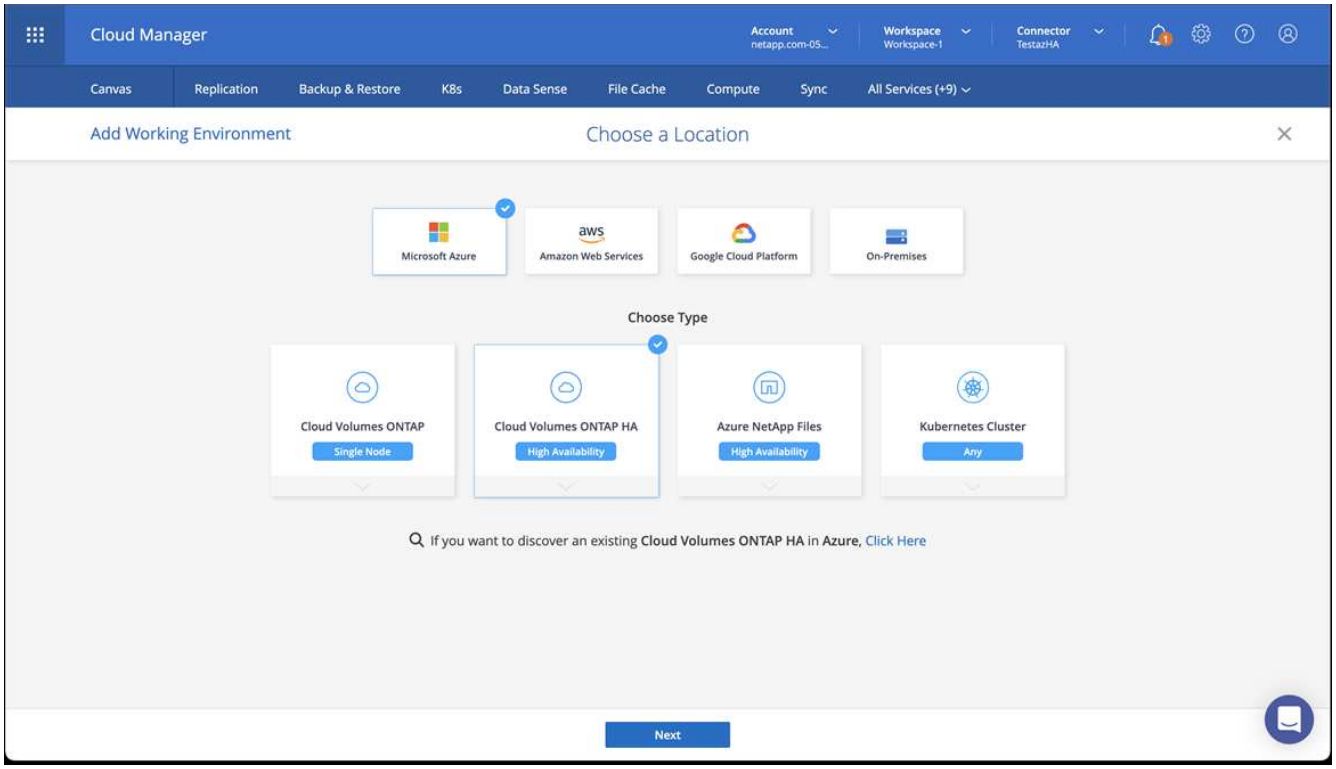
      请参见 "从 Cloud Manager 在 Azure 中创建连接器"。
3. 确保连接器正在运行，然后切换到该连接器。



4. 为您的云环境创建一个工作环境。

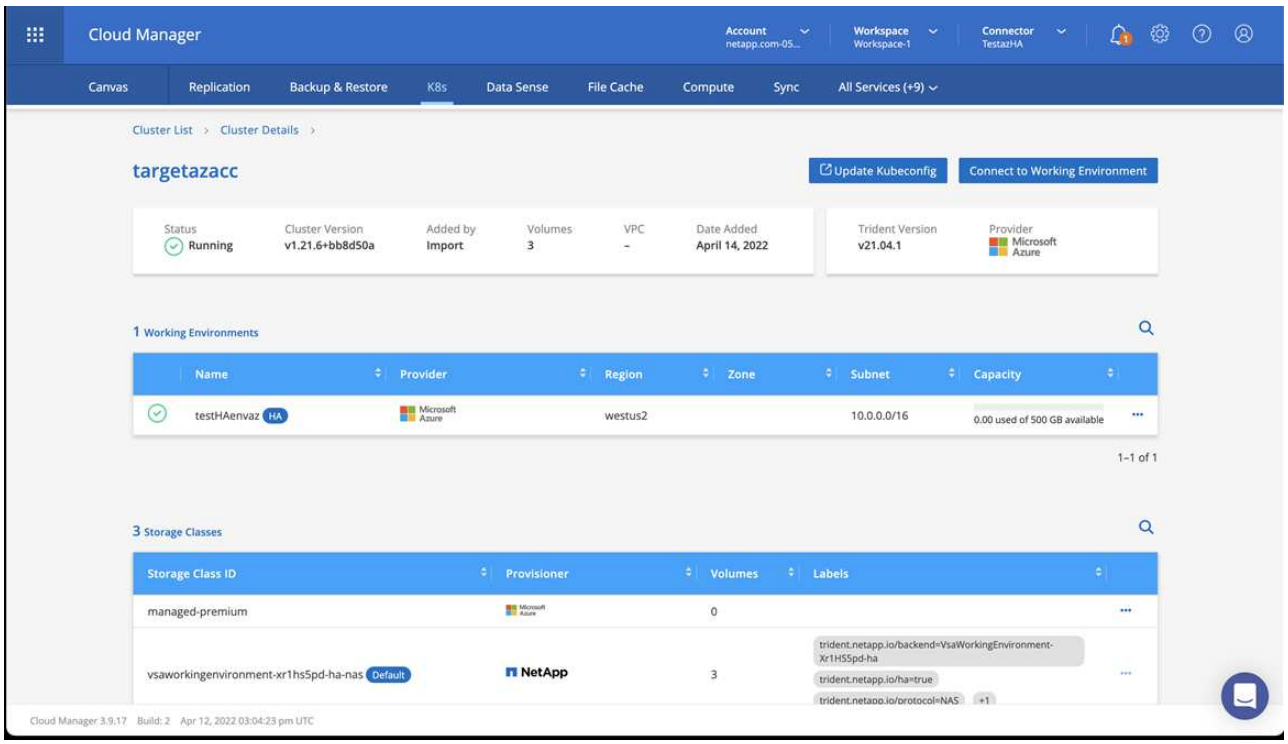
   a. 位置： "Microsoft Azure" 。

   b. 键入： Cloud Volumes ONTAP HA 。



5. 导入 OpenShift 集群。集群将连接到您刚刚创建的工作环境。

a. 选择 * K8s* > * 集群列表 * > * 集群详细信息 * ，查看 NetApp 集群详细信息。



b. 在右上角，记下 Trident 版本。

c. 记下显示 NetApp 作为配置程序的 Cloud Volumes ONTAP 集群存储类。

此操作将导入 Red Hat OpenShift 集群并分配默认存储类。您可以选择存储类。Trident 会在导入和发现过程中自动安装。

6. 记下此Cloud Volumes ONTAP 部署中的所有永久性卷和卷。
7. Cloud Volumes ONTAP 可以作为单个节点运行，也可以在高可用性环境下运行。如果已启用 HA ，请记下在 Azure 中运行的 HA 状态和节点部署状态。

安装和配置 **Astra** 控制中心

按照标准安装 Astra 控制中心 "安装说明"。

使用 Astra 控制中心添加 Azure 存储分段。请参见 "设置 Astra 控制中心并添加存储分段"。