



安装概述

Astra Control Center

NetApp
November 21, 2023

This PDF was generated from https://docs.netapp.com/zh-cn/astra-control-center-2208/get-started/acc_cluster_cr_options.html on November 21, 2023. Always check docs.netapp.com for the latest.

目录

安装概述	1
使用标准流程安装 Astra 控制中心	1
使用 OpenShift OperatorHub 安装 Astra 控制中心	27
使用 Cloud Volumes ONTAP 存储后端安装 Astra 控制中心	34

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 Control Center，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 Control Center Operator 和 Astra Control Center。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"](#)。
- 如果您已在环境中配置或希望配置 POD 安全策略，请熟悉 POD 安全策略及其对 Astra Control Center 安装的影响。请参见 ["了解 POD 安全策略限制"](#)。
- 确保所有集群操作员均处于运行状况良好且可用。

```
kubectl get clusteroperators
```

- 确保所有 API 服务均处于运行状况良好且可用：

```
kubectl get apiservices
```

- 确保您计划使用的 Astra FQDN 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。
- 如果集群中已存在证书管理器，则需要执行某些操作 ["前提条件步骤"](#) 这样，Astra 控制中心就不会安装自己的证书管理器。

关于此任务

Astra 控制中心安装过程将执行以下操作：

- 将 Astra 组件安装到中 `netapp-acc` (或自定义命名的)命名空间。
- 创建默认帐户。
- 建立默认管理用户电子邮件地址和默认一次性密码。系统会为该用户分配首次登录到 UI 所需的所有者角色。
- 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 workflow、请仅在上部署Astra控制中心 pcloud 命名空间、而不是 netapp-acc 此操作步骤 的步骤中介绍的命名空间或自定义命名空间。



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](#)
- [\[验证系统状态\]](#)
- [\[设置传入以进行负载均衡\]](#)
- [登录到 Astra 控制中心 UI](#)

下载并解包Astra Control Center软件包

1. 下载 Astra Control Center 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 AstraControlCenter-public.pub -signature
astra-control-center-[version].tar.gz.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 命令行插件可节省执行与部署和升级Astra控制中心相关的常见任务所需的时间。

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` 用于收集此信息的命令。

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架构所需文件的名称：

```
ls kubectl-astra/
```

2. 将文件复制到与标准文件相同的位置 kubectl 实用程序。在此示例中、将显示 kubectl 实用程序位于中 `/usr/local/bin` 目录。替换 `<binary-name>` 使用所需文件的名称：

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

将映像添加到本地注册表

1. 为容器引擎完成相应的步骤顺序：

Docker

1. 更改为Astra目录：

```
cd acc
```

2. [substep_image_local_registry_push]]将Astra控制中心映像目录中的软件包映像推送到本地注册表。
在运行命令之前、请执行以下替换操作：

- 将bundle_file替换为Astra Control捆绑包文件的名称(例如、acc.manifest.yaml)。
- 将my_regRegistry替换为Docker存储库的URL。
- 将my_registry_user替换为用户名。
- 将my_registry_token替换为注册表的授权令牌。

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

Podman

1. 登录到注册表：

```
podman login [your_registry_path]
```

2. 运行以下脚本、按照注释中的说明替换<your_registry>：

```
# 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
```

为具有身份验证要求的注册表设置命名空间和密钥

1. 导出Astra控制中心主机集群的KUBECONFIG:

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

2. 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. 为创建密钥 netapp-acc-operator 命名空间。添加 Docker 信息并运行以下命令：



占位符 `your_registry_path` 应与您先前上传的映像的位置匹配(例如、`[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]
```

响应示例：

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



如果在生成密钥后删除命名空间、则需要在重新创建命名空间后重新生成命名空间的密钥。

- c. 创建 netapp-acc (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. 为创建密钥 netapp-acc (或自定义命名)命名空间。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. 更改目录：

```
cd manifests
```

2. 编辑Astra控制中心操作员部署YAML (astra_control_center_operator_deploy.yaml)以引用您的本地注册表和密钥。

```
vim astra_control_center_operator_deploy.yaml
```



以下步骤将提供一个标注的YAML示例。

a. 如果您使用的注册表需要身份验证、请替换的默认行 `imagePullSecrets: []` 使用以下命令：

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

- b. 更改 `[your_registry_path]`。 `kube-rbac-proxy` 将映像推送到注册表路径中 [上一步](#)。
- c. 更改 `[your_registry_path]`。 `acc-operator-controller-manager` 将映像推送到注册表路径中 [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: []

```

3. 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
```

4. 验证Pod是否正在运行：

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

配置 Astra 控制中心

1. 编辑Astra Control Center自定义资源(CR)文件 (astra_control_center_min.yaml)进行帐户、AutoSupport、注册表和其他必要配置：



astra_control_center_min.yaml 是默认CR、适用于大多数安装。熟悉所有内容 ["CR 选项及其潜在值"](#) 以确保为您的环境正确部署Astra控制中心。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。

```
vim astra_control_center_min.yaml
```



如果您使用的注册表不需要授权、则必须删除 secret 行内 imageRegistry 否则安装将失败。

- a. 更改 [your_registry_path] 到上一步中将映像推送到的注册表路径。
- b. 更改 accountName 字符串、表示要与帐户关联的名称。
- c. 更改 astraAddress 指向要在浏览器中用于访问Astra的FQDN的字符串。请勿使用 http:// 或 https:// 地址中。复制此 FQDN 以在中使用 [后续步骤](#)。

- d. 更改 email 字符串到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. 更改 enrolled 用于将AutoSupport 连接到 false 对于不具有Internet连接或保留的站点 true 对于已连接站点。
- f. 如果使用外部证书管理器、请将以下行添加到 spec:

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

- g. (可选)添加名字 firstName 和姓氏 lastName 与帐户关联的用户的。您可以在用户界面中立即或稍后执行此步骤。
- h. (可选)更改 storageClass 如果您的安装需要、则为另一个Trident storageClass资源指定值。
- i. (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](#)下、通过在此YAML文件中添加一个名为的新字段来提供密钥名称
astraKubeConfigSecret: "acc-kubeconfig-cred or custom secret name"
- j. 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 Control Center安装用于设置其网关 (service/traefik)的类型 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、请设置 ingressType to AccTraefik。

这将部署Astra控制中心 traefik 网关作为Kubernetes loadbalancer类型的服务。

Astra控制中心使用类型为"loadbalancer"的服务 (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 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执行此操作、请创建 netapp-acc (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. 在中安装Astra控制中心 netapp-acc (或自定义)命名空间:

```
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-6b44d68d94-d8m55 13m	1/1	Running	0
activity-78f99ddf8-hltct 10m	1/1	Running	0
api-token-authentication-457nl 9m28s	1/1	Running	0
api-token-authentication-dgwsz 9m28s	1/1	Running	0
api-token-authentication-hmqqc 9m28s	1/1	Running	0
asup-75fd554dc6-m6qzh 9m38s	1/1	Running	0
authentication-6779b4c85d-92gds 8m11s	1/1	Running	0
bucket-service-7cc767f8f8-lqwr8 9m31s	1/1	Running	0
certificates-549fd5d6cb-5kmd6 9m56s	1/1	Running	0
certificates-549fd5d6cb-bkj9 9m56s	1/1	Running	0
cloud-extension-7bcb7948b-hn8h2 10m	1/1	Running	0
cloud-insights-service-56ccf86647-fgg69 9m46s	1/1	Running	0
composite-compute-677685b9bb-7vgsf 10m	1/1	Running	0
composite-volume-657d6c5585-dnq79 9m49s	1/1	Running	0
credentials-755fd867c8-vrlmt 11m	1/1	Running	0
entitlement-86495cdf5b-nwhh2 10m	1/1	Running	2
features-5684fb8b56-8d6s8 10m	1/1	Running	0
fluent-bit-ds-rhx7v 7m48s	1/1	Running	0
fluent-bit-ds-rjms4 7m48s	1/1	Running	0
fluent-bit-ds-zf5ph 7m48s	1/1	Running	0
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-vmppm	1/1	Running	0
10m			
traefik-7489dc59f9-6mnst	1/1	Running	0
3m19s			
traefik-7489dc59f9-xrkkg	1/1	Running	0
3m4s			
trident-svc-6c8dc458f5-jswcl	1/1	Running	0
10m			
vault-controller-6b954f9b76-gz9nm	1/1	Running	0
11m			

2. (可选)为确保安装完成、您可以观看 `acc-operator` 使用以下命令记录。

```
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. 在所有Pod运行时、验证安装是否成功 (READY 为 True)并获取登录到Astra控制中心时要使用的一次性密码：

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

响应：

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



复制UUID值。密码为 ACC- 后跟UUID值 (ACC-[UUID] 或者、在此示例中、 ACC-9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f) 。

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 (`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 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"](#)。

根据您的使用的入口控制器类型，步骤会有所不同：

- Istio入口
- 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 。

Istio入口的步骤

1. 配置Istio入口。



此操作步骤 假定使用"默认"配置文件部署Istio。

2. 为传入网关收集或创建所需的证书和专用密钥文件。

您可以使用CA签名或自签名证书。公用名必须为Astra地址(FQDN)。

命令示例：

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

3. 创建密钥 `tls secret name` 类型 `kubernetes.io/tls` 中的TLS专用密钥和证书 `istio-system namespace` 如TLS机密中所述。

命令示例：

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



密钥名称应与匹配 `spec.tls.secretName` 在中提供 `istio-ingress.yaml` 文件

4. 在中部署传入资源 `netapp-acc` (或自定义命名)命名空间使用v1beta1 (在Kubernetes版本低于或1.22的情况下已弃用)或v1资源类型作为已弃用或新模式：

输出：

```

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 address>
    secretName: [tls secret name]
  rules:
  - host: [ACC address]
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          serviceName: traefik
          servicePort: 80

```

对于v1新模式、请遵循以下示例：

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

输出：

```

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 address>
    secretName: [tls secret name]
  rules:
  - host: [ACC address]
    http:
      paths:
      - path: /
        pathType: Prefix
        backend:
          service:
            name: traefik
            port:
              number: 80

```

5. 照常部署Astra控制中心。

6. 检查入口状态：

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

响应：

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

nginx 入口控制器的步骤

1. 创建类型的密钥[kubernetes.io/tls]中的TLS专用密钥和证书 netapp-acc (或自定义命名的)命名空

间、如中所述 "TLS 密钥"。

2. 在中部署传入资源 `netapp-acc` (或自定义命名的)命名空间 `v1beta1` (在低于或1.22的Kubernetes版本中已弃用)或 `v1` 已弃用或新模式的资源类型:

- a. 对于 `v1beta1` 已弃用模式、请按照以下示例进行操作:

```
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. 。 `v1` 新架构、请按照以下示例进行操作:

```

apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: netapp-acc-ingress
  namespace: [netapp-acc or custom namespace]
spec:
  ingressClassName: [class name for nginx controller]
  tls:
  - hosts:
    - <ACC address>
    secretName: [tls secret name]
  rules:
  - host: <ACC address>
    http:
      paths:
      - path:
        backend:
          service:
            name: traefik
            port:
              number: 80
        pathType: ImplementationSpecific

```

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、输入在中使用的 FQDN astraAddress 在中 astra_control_center_min.yaml CR 时间您安装了 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、输入您用于的值 email 在中 astra_control_center_min.yaml CR时间 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

如果有任何服务位于中 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"。

=
:allow-uri-read:

了解POD安全策略限制

Astra控制中心通过POD安全策略(PSP)支持权限限制。通过POD安全策略、您可以限制哪些用户或组能够运行容器以及这些容器可以具有哪些权限。

某些Kubernetes分发版(例如RKE2)的默认POD安全策略限制性过强、在安装Astra Control Center时会出现问题。

您可以使用此处提供的信息和示例来了解Astra控制中心创建的POD安全策略、并配置POD安全策略、以便在不干扰Astra控制中心功能的情况下提供所需的保护。

由Astra控制中心安装的Psp

Astra控制中心会在安装期间创建多个POD安全策略。其中一些是永久性的、其中一些是在某些操作期间创建的、操作完成后会将其删除。

在安装期间创建的Psp

在安装Astra控制中心期间、Astra控制中心操作员会安装一个自定义POD安全策略、一个角色对象和一个RoleBinding.对象、以支持在Astra控制中心命名空间中部署Astra控制中心服务。

新策略和对象具有以下属性：

```
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	

备份操作期间创建的Psp

在备份操作期间、Astra控制中心会创建一个动态POD安全策略、一个ClusterRole对象和一个RoleBinding.它们支持备份过程、该过程会在单独的命名空间中进行。

新策略和对象具有以下属性：

```
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

在集群管理期间创建的Psp

管理集群时、Astra控制中心会在受管集群中安装NetApp监控操作员。此运算符将创建Pod安全策略、ClusterRole对象和RoleBinding对象、以便在Astra Control Center命名空间中部署遥测服务。

新策略和对象具有以下属性：

```
kubectl get psp
```

NAME		PRIV	CAPS		
SELINUX	RUNASUSER	FSGROUP	SUPGROUP	READONLYROOTFS	
VOLUMES					
netapp-monitoring-bsp-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

在命名空间之间启用网络通信

某些环境使用NetworkPolicy构造来限制命名空间之间的流量。Astra控制中心操作员、Astra控制中心和适用于VMware vSphere的Astra插件都位于不同的命名空间中。这些不同命名空间中的服务需要能够彼此通信。要启用此通信、请执行以下步骤。

步骤

1. 删除Astra控制中心命名空间中的任何NetworkPolicy资源：

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

2. 对于上述命令返回的每个NetworkPolicy对象、请使用以下命令将其删除。将<object_name>替换为返回对象的名称：

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

3. 应用以下资源文件以配置Acc-AVP-network-policy对象、以允许适用于VMware vSphere的Astra插件服务向Astra控制中心服务发出请求。将括号<>中的信息替换为您环境中的信息：

```
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. 应用以下资源文件来配置Acc-operator-network-policy对象、以使Astra控制中心操作员能够与Astra控制中心服务进行通信。将括号<>中的信息替换为您环境中的信息：

```

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

```

消除资源限制

某些环境使用ResourceQuotas和LimitRanges对象来防止命名空间中的资源占用集群上的所有可用CPU和内存。Astra控制中心未设置最大限制、因此不符合这些资源的要求。您需要将其从计划安装Astra控制中心的命名空间中删除。

您可以使用以下步骤检索和删除这些配额和限制。在这些示例中、命令输出会立即显示在命令后面。

步骤

1. 在NetApp-Accc命名空间中获取资源配额：

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

响应：

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. 按名称删除所有资源配额：

```
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. 在NetApp-Accc命名空间中获取限制范围：

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

响应：

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

4. 按名称删除限制范围：

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

=
:allow-uri-read:

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 为 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 为 true)：

```
oc get apiservices
```

- 在数据中心中为Astra控制中心创建FQDN地址。
- 获取对Red Hat OpenShift容器平台的必要权限和访问权限、以执行所述的安装步骤。
- 如果集群中已存在证书管理器、则需要执行某些操作 ["前提条件步骤"](#) 这样、Astra控制中心就不会安装自己的证书管理器。

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. 下载 Astra Control Center 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 AstraControlCenter-public.pub -signature  
astra-control-center-[version].tar.gz.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 命令行插件可节省执行与部署和升级Astra控制中心相关的常见任务所需的时间。

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` 用于收集此信息的命令。

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架构所需文件的名称：

```
ls kubectl-astra/
```

2. 将文件复制到与标准文件相同的位置 kubectl 实用程序。在此示例中、将显示 kubectl 实用程序位于中 /usr/local/bin 目录。替换 <binary-name> 使用所需文件的名称：

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

将映像添加到本地注册表

1. 为容器引擎完成相应的步骤顺序：

Docker

1. 更改为Astra目录：

```
cd acc
```

2. [substep_image_local_registry_push]]将Astra控制中心映像目录中的软件包映像推送到本地注册表。
在运行命令之前、请执行以下替换操作：

- 将bundle_file替换为Astra Control捆绑包文件的名称(例如、acc.manifest.yaml)。
- 将my_regRegistry替换为Docker存储库的URL。
- 将my_registry_user替换为用户名。
- 将my_registry_token替换为注册表的授权令牌。

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

Podman

1. 登录到注册表：

```
podman login [your_registry_path]
```

2. 运行以下脚本、按照注释中的说明替换<your_registry>：


```

# 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:.*/::')

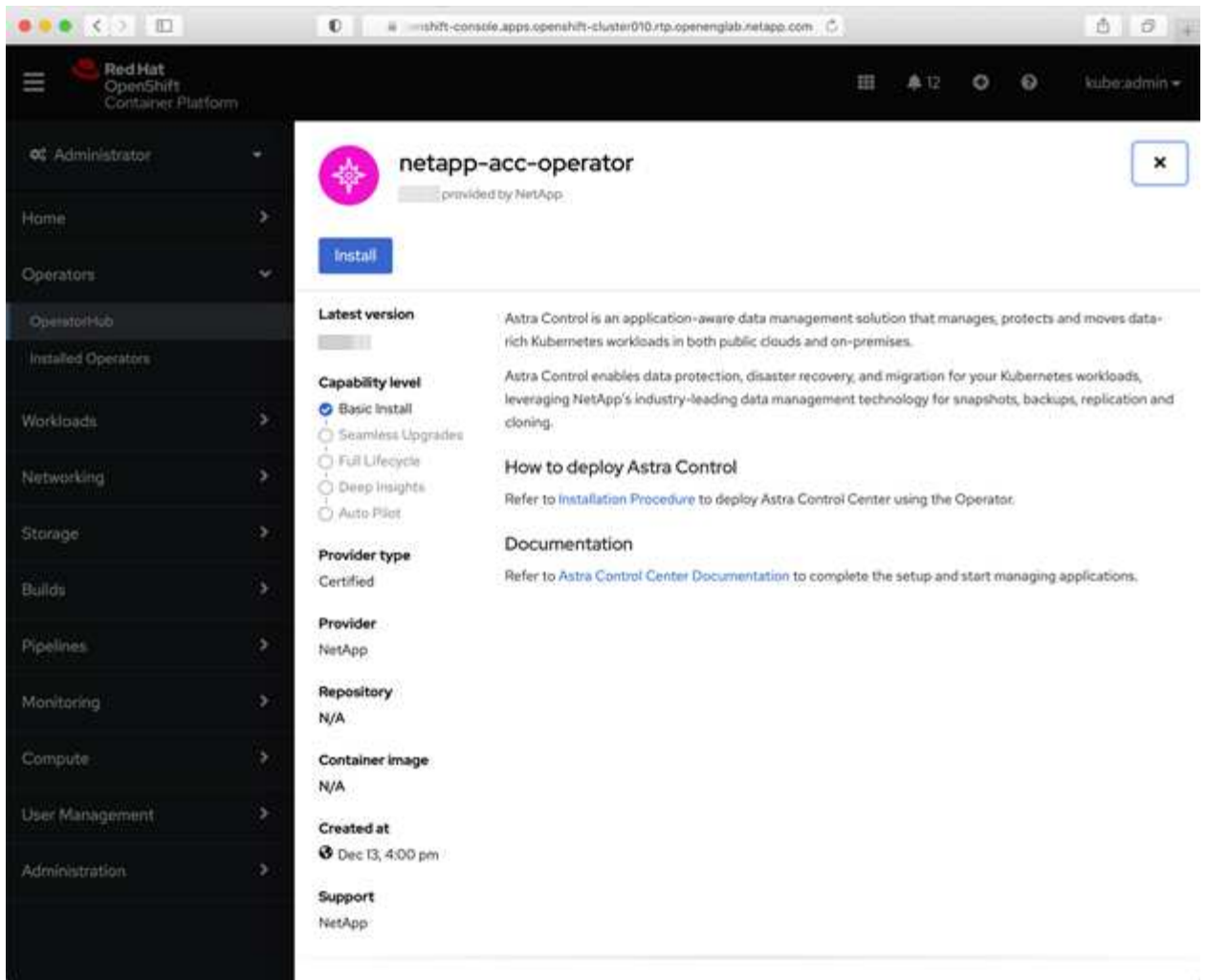
    # Tag with local image repo.
    podman tag ${astraImage} ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}

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

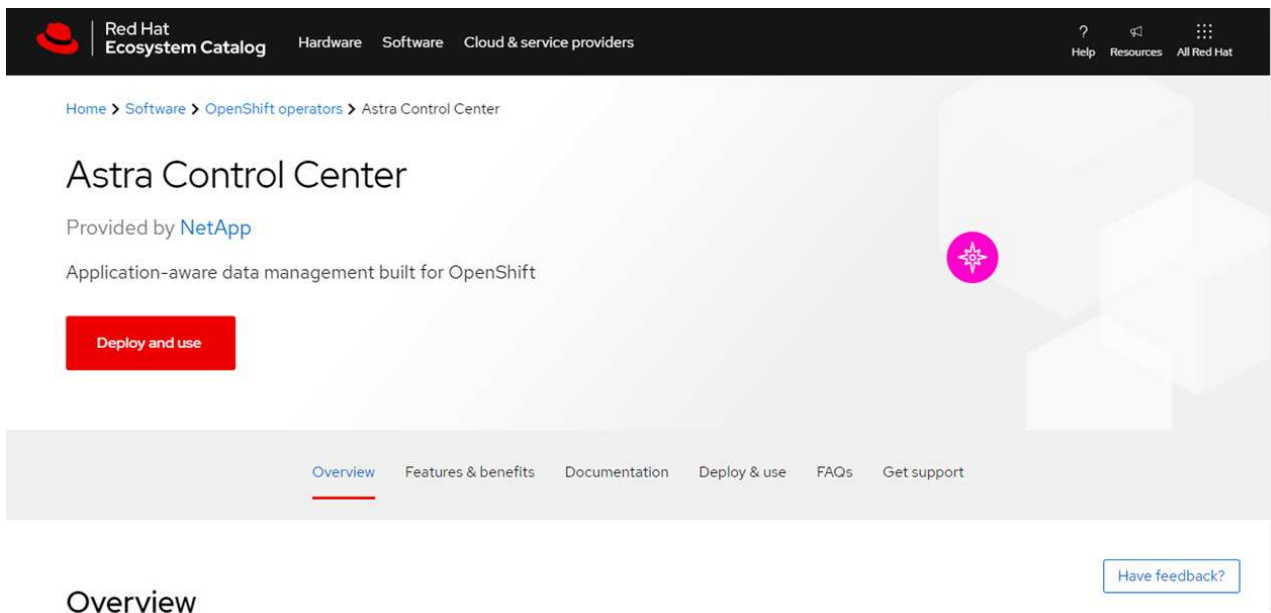
```

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. 选择操作符命名空间或 `netapp-acc-operator` 命名空间将在操作员安装过程中自动创建。
- 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中的控制台中、选择 `Create instance` 在提供的API部分中。
2. 完成 `Create AstraControlCenter` 表单字段：
 - 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。请止步 `http://` 或 `https://` 地址中。
 - 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。请止步 <http://> 或 <https://> 地址中。
- 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)、Google 云平台 (GCP) 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 Google Cloud Platform 中部署 Astra 控制中心](#)
- [在 Microsoft Azure 中部署 Astra 控制中心](#)

您可以使用自管理 Kubernetes 集群 (例如 OpenShift 容器平台 (OCP)) 在分发版中管理应用程序。只有自管理的 OCP 集群才会通过验证来部署 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。

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
- 如果使用OCP、则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
负载均衡器	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 指向负载均衡 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	<p>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。</p> <div> <p>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。</p> </div>

组件	要求
Astra Trident / ONTAP 配置	<p>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：</p> <ul style="list-style-type: none"> • <code>vsaworkingenvironment-<>-ha-nas</code> <code>csi.trident.netapp.io</code> • <code>vsaworkingenvironment-<>-ha-san</code> <code>csi.trident.netapp.io</code> • <code>vsaworkingenvironment-<>-single-nas</code> <code>csi.trident.netapp.io</code> • <code>vsaworkingenvironment-<>-single-san</code> <code>csi.trident.netapp.io</code>



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。



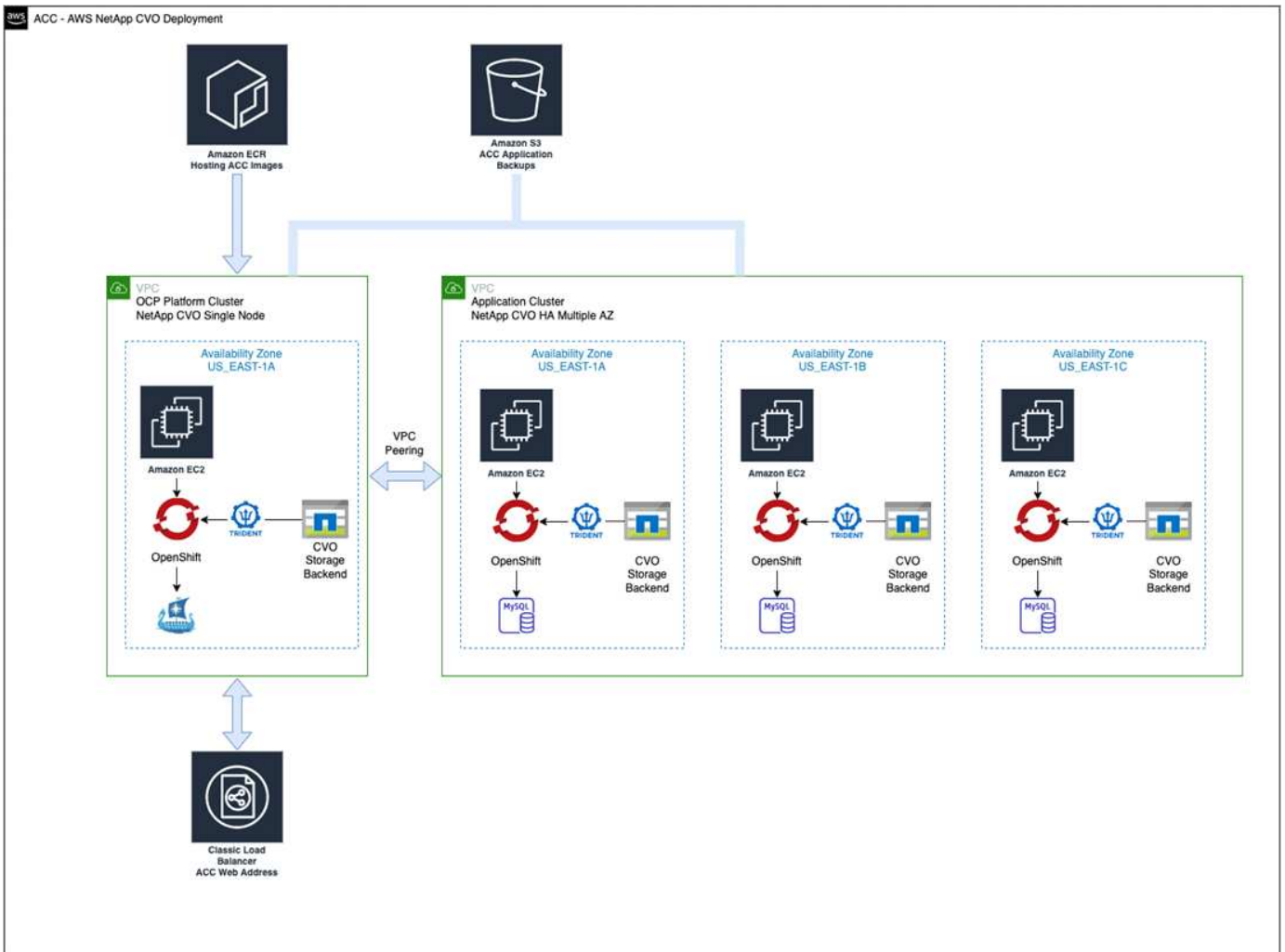
如果不创建ECR、则Astra控制中心无法从包含Cloud Volumes ONTAP 且具有AWS后端的集群访问监控数据。如果您尝试使用 Astra 控制中心发现和管理的集群没有 AWS ECR 访问权限，则会导致出现问题描述 。

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"](#)。



AWS使用通用S3存储分段类型。

在Google Cloud Platform中部署Astra控制中心

您可以在Google云平台(GCP)公有云上托管的自管理Kubernetes集群上部署Astra控制中心。

GCP所需的功能

在GCP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
- 如果使用的是OCP、则为Red Hat OpenShift Container Platform (OCP) 4.10
- 如果使用OCP、则Red Hat OpenShift Container Platform (OCP)权限(在命名空间级别用于创建Pod)
- GCP服务帐户、具有创建存储分段和连接器的权限


GCP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：

组件	要求
后端 NetApp Cloud Volumes ONTAP 存储容量	至少 300 GB 可用
工作节点(GCP 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
负载均衡器	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 (GCP 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	<p>您必须具有现有的专用注册表、例如Google Container Registry、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 。</p> <div>  <p>您需要启用匿名访问以提取要备份的 Restic 映像。</p> </div>
Astra Trident / ONTAP 配置	<p>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：</p> <ul style="list-style-type: none"> • vsaworkingenvironment-<>-ha-nas csi.trident.netapp.io • vsaworkingenvironment-<>-ha-san csi.trident.netapp.io • vsaworkingenvironment-<>-single-nas csi.trident.netapp.io • vsaworkingenvironment-<>-single-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。

GCP部署概述

下面概述了在GCP中将Cloud Volumes ONTAP 作为存储后端的自管理OCP集群上安装Astra控制中心的过程。

下面详细介绍了其中每个步骤。

1. [在GCP上安装RedHat OpenShift集群。](#)
2. [创建GCP项目和虚拟私有云。](#)
3. [确保您具有足够的 IAM 权限。](#)
4. [配置GCP。](#)
5. [配置 NetApp Cloud Manager。](#)
6. [安装和配置 Astra 控制中心。](#)

在GCP上安装RedHat OpenShift集群

第一步是在GCP上安装RedHat OpenShift集群。

有关安装说明，请参见以下内容：

- ["在GCP中安装OpenShift集群"](#)
- ["创建GCP服务帐户"](#)

创建GCP项目和虚拟私有云

至少创建一个GCP项目和虚拟私有云(Virtual Private Cloud、VPC)。



OpenShift 可能会创建自己的资源组。此外、您还应定义GCP VPC。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 ["初始GCP凭据和权限"](#)。

配置GCP

接下来、将GCP配置为创建VPC、设置计算实例、创建Google Cloud Object Storage、创建用于托管Astra控制中心映像的Google Container Register并将这些映像推送到此注册表。

按照GCP文档完成以下步骤。请参见在GCP中安装OpenShift集群。

1. 在GCP中创建一个GCP项目和VPC、该项目和VPC计划用于具有CVO后端的OCP集群。
2. 查看计算实例。此服务器可以是GCP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GCP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GCP Cloud Storage Bucket以存储备份。
5. 创建存储分段访问所需的密钥。
6. 创建Google容器注册表以托管所有Astra控制中心映像。
7. 为所有Astra控制中心映像设置用于Docker推/拉的Google容器注册表访问权限。

示例：输入以下脚本可将Accc映像推送到此注册表：

```
gcloud auth activate-service-account <service account email address>
--key-file=<GCP Service Account JSON file>
```

此脚本需要一个Astra控制中心清单文件以及您的Google映像注册表位置。

示例

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

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

8. 设置 DNS 区域。

配置 NetApp Cloud Manager

使用Cloud Manager创建工作空间、向GCP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 ["GCP中的Cloud Volumes ONTAP 入门"](#)。

您需要的内容

- 使用所需的IAM权限和角色访问GCP服务帐户

步骤

1. 将凭据添加到 Cloud Manager 。请参见 ["正在添加GCP帐户"](#)。
2. 为GCP添加一个连接器。
 - a. 选择"GCP"作为提供程序。
 - b. 输入GCP凭据。请参见 ["通过Cloud Manager在GCP中创建连接器"](#)。
 - c. 确保连接器正在运行，然后切换到该连接器。
3. 为您的云环境创建一个工作环境。
 - a. 位置: "GCP"
 - b. 类型: Cloud Volumes ONTAP HA
4. 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. 记下显示为"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。

5. 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(HA)中运行。如果已启用 HA、请记下在GCP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](#)"。



GCP使用通用S3存储分段类型。

1. 生成Docker密钥以提取用于Astra控制中心安装的映像：

```
kubectl create secret docker-registry <secret name>
--docker-server=<Registry location>
--docker-username=_json_key
--docker-password="$(cat <GCP Service Account JSON file>)"
--namespace=pcloud
```

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
- 如果使用的是OCP、则为Red Hat OpenShift Container Platform (OCP) 4.8
- 如果使用OCP、则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
负载均衡器	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 （ Azure DNS 区域）	一种将 Astra 控制中心的 FQDN 指向负载均衡 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	<p>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。</p> <div>  <p>您需要启用匿名访问以提取要备份的 Restic 映像。</p> </div>
Astra Trident / ONTAP 配置	<p>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：</p> <ul style="list-style-type: none"> • <code>vsaworkingenvironment-<>-ha-nas</code> <code>csi.trident.netapp.io</code> • <code>vsaworkingenvironment-<>-ha-san</code> <code>csi.trident.netapp.io</code> • <code>vsaworkingenvironment-<>-single-nas</code> <code>csi.trident.netapp.io</code> • <code>vsaworkingenvironment-<>-single-san</code> <code>csi.trident.netapp.io</code>



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 $AZ_ACR_REGISTRY/$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

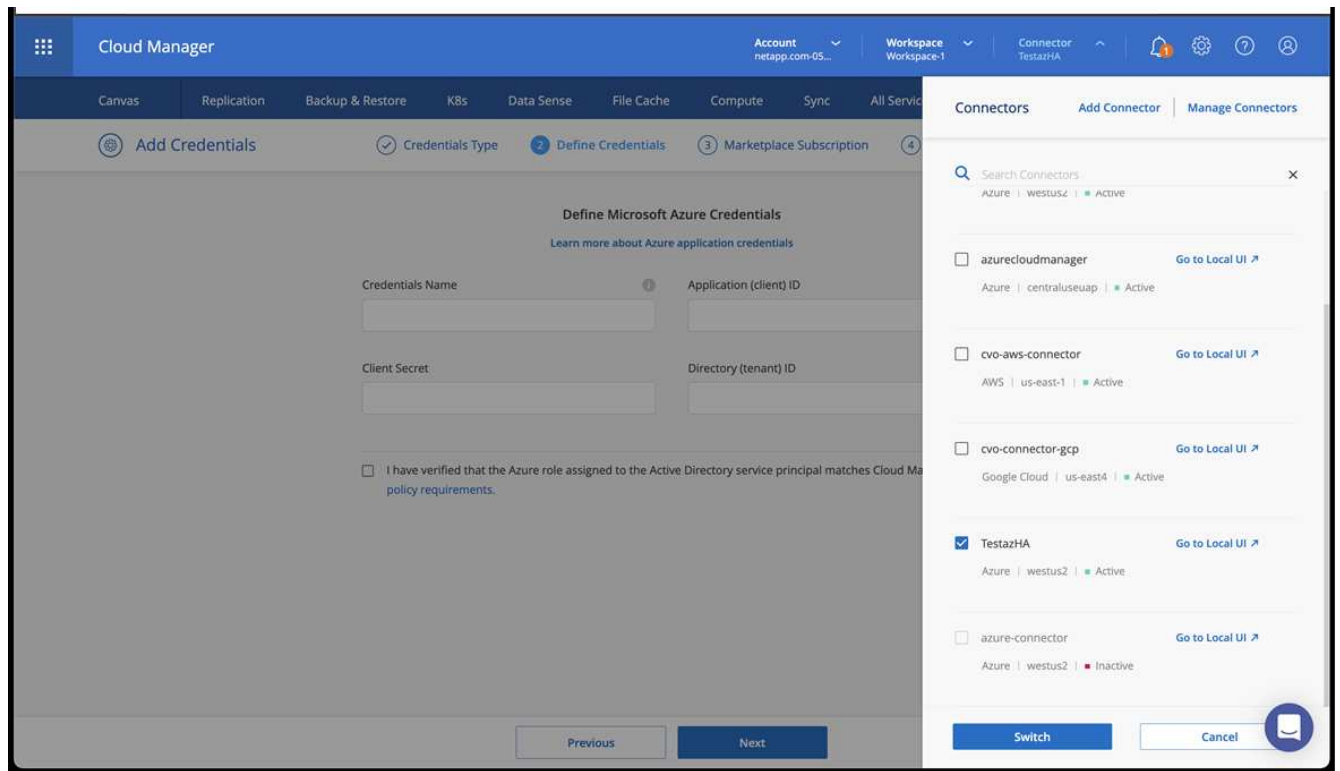
使用所需的 IAM 权限和角色访问 Azure 帐户

步骤

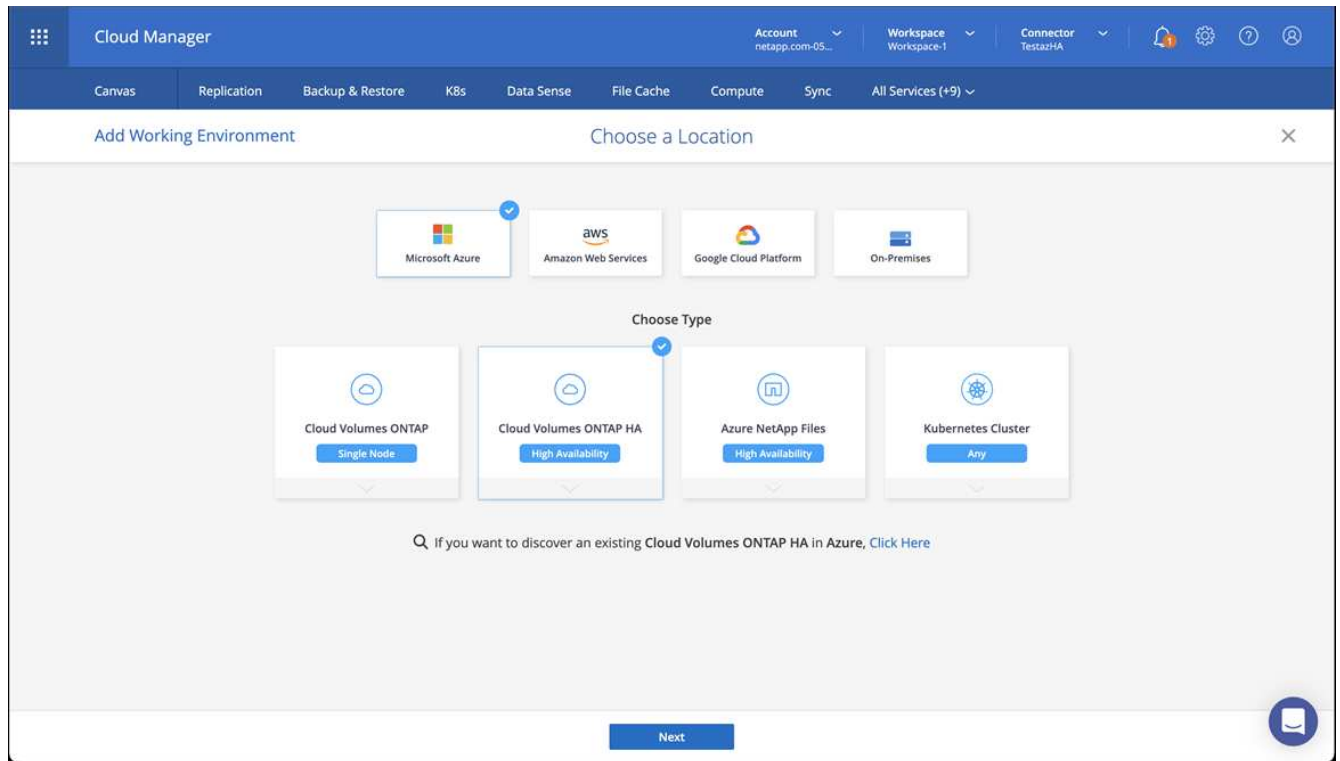
1. 将凭据添加到 Cloud Manager 。
2. 添加适用于 Azure 的连接器。请参见 ["Cloud Manager 策略"](#)。
 - a. 选择 * Azure * 作为提供程序。
 - b. 输入 Azure 凭据，包括应用程序 ID ， 客户端密钥和目录（租户） ID 。

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。

The screenshot shows the 'Cloud Manager' interface. The top navigation bar includes 'Canvas', 'Replication', 'Backup & Restore', 'K8s', 'Data Sense', 'File Cache', 'Compute', 'Sync', and 'All Services (+9)'. The 'K8s' tab is selected. Below the navigation bar, the 'Cluster List' and 'Cluster Details' tabs are visible. The main content area displays the details for a cluster named 'targetazacc'. It includes a status bar with 'Running' and a green checkmark, and buttons for 'Update Kubeconfig' and 'Connect to Working Environment'. Below this, a table lists 'Working Environments' with columns: Name, Provider, Region, Zone, Subnet, and Capacity. The table shows one entry: 'testHAenvaz HA' with Provider 'Microsoft Azure', Region 'westus2', Subnet '10.0.0.0/16', and Capacity '0.00 used of 500 GB available'. Below the table, a section titled 'Storage Classes' shows a table with columns: Storage Class ID, Provisioner, Volumes, and Labels. The table lists two storage classes: 'managed-premium' with Provisioner 'Microsoft Azure' and Volumes '0'; and 'vsaworkingenvironment-xr1hs5pd-ha-nas' with Provisioner 'NetApp' and Volumes '3'. The 'vsaworkingenvironment-xr1hs5pd-ha-nas' storage class is highlighted with a blue 'Default' label. The 'Labels' column for this storage class shows three labels: 'trident.netapp.io/backend=Vsaworkingenvironment-xr1hs5pd-ha', 'trident.netapp.io/ha=true', and 'trident.netapp.io/protocol=NAS'.

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"。

版权信息

版权所有 © 2023 NetApp, Inc.。保留所有权利。中国印刷。未经版权所有者事先书面许可，本文档中受版权保护的任何部分不得以任何形式或通过任何手段（图片、电子或机械方式，包括影印、录音、录像或存储在电子检索系统中）进行复制。

从受版权保护的 NetApp 资料派生的软件受以下许可和免责声明的约束：

本软件由 NetApp 按“原样”提供，不含任何明示或暗示担保，包括但不限于适销性以及针对特定用途的适用性的隐含担保，特此声明不承担任何责任。在任何情况下，对于因使用本软件而以任何方式造成的任何直接性、间接性、偶然性、特殊性、惩罚性或后果性损失（包括但不限于购买替代商品或服务；使用、数据或利润方面的损失；或者业务中断），无论原因如何以及基于何种责任理论，无论出于合同、严格责任或侵权行为（包括疏忽或其他行为），NetApp 均不承担责任，即使已被告知存在上述损失的可能性。

NetApp 保留在不另行通知的情况下随时对本文档所述的任何产品进行更改的权利。除非 NetApp 以书面形式明确同意，否则 NetApp 不承担因使用本文档所述产品而产生的任何责任或义务。使用或购买本产品不表示获得 NetApp 的任何专利权、商标权或任何其他知识产权许可。

本手册中描述的产品可能受一项或多项美国专利、外国专利或正在申请的专利的保护。

有限权利说明：政府使用、复制或公开本文档受 DFARS 252.227-7013（2014 年 2 月）和 FAR 52.227-19（2007 年 12 月）中“技术数据权利 — 非商用”条款第 (b)(3) 条规定的限制条件的约束。

本文档中所含数据与商业产品和/或商业服务（定义见 FAR 2.101）相关，属于 NetApp, Inc. 的专有信息。根据本协议提供的所有 NetApp 技术数据和计算机软件具有商业性质，并完全由私人出资开发。美国政府对这些数据的使用权具有非排他性、全球性、受限且不可撤销的许可，该许可既不可转让，也不可再许可，但仅限在与交付数据所依据的美国政府合同有关且受合同支持的情况下使用。除本文档规定的情形外，未经 NetApp, Inc. 事先书面批准，不得使用、披露、复制、修改、操作或显示这些数据。美国政府对国防部的授权仅限于 DFARS 的第 252.227-7015(b)（2014 年 2 月）条款中明确的权利。

商标信息

NetApp、NetApp 标识和 <http://www.netapp.com/TM> 上所列的商标是 NetApp, Inc. 的商标。其他公司和产品名称可能是其各自所有者的商标。