# 入门

## Astra Control Center

NetApp
November 21, 2023

# 目录

# 入门

= :allow-uri-read:

## Astra 控制中心要求

首先验证操作环境，应用程序集群，应用程序，许可证和 Web 浏览器的就绪情况。

- [操作环境要求]
- [支持的存储后端]
- 访问 Internet
- [许可证]
- 内部 Kubernetes 集群的传入
- [网络要求]
- 支持的 Web 浏览器
- [应用程序集群的其他要求]
- Google Anthos集群要求
- VMware Tanzu Kubernetes Grid 集群要求

### 操作环境要求

Astra控制中心已在以下类型的操作环境中进行了验证：

- 采用Kubernetes 1.22的Cisco IKS
- Google Anthos 1.11或1.12 (请参见 Google Anthos集群要求）
- Rancher Kubernetes Engine (RKE）：
  - RKE1.3.12与Rancher 2.6.5和2.6.6
  - RKE1.3.13与Rancher 2.6.8
  - RKE2 (v1.23.6+rke2r1)与Rancher 2.6.5和2.6.6
  - RKE2 (v1.24.x)与Rancher 2.6.8
- Red Hat OpenShift容器平台4.8至4.11
- 上游Kubernetes 1.23到1.25 (Kubernetes 1.25需要Astra Trident 22.10或更高版本)
- VMware Tanzu Kubernetes网格：(请参见 VMware Tanzu Kubernetes Grid 集群要求）
  - VMware Tanzu Kubernetes网格1.5
  - VMware Tanzu Kubernetes Grid Integrated Edition 1.13和1.14

确保您选择托管 Astra 控制中心的操作环境满足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 |
| --- | --- |
| CPU扩展 | 托管环境中所有节点的CPU必须启用AVX扩展。 |
| 存储后端容量 | 至少500 GB可用 |
| 工作节点 | 总共至少 3 个辅助节点，每个节点有 4 个 CPU 核和 12 GB RAM |
| FQDN 地址 | Astra 控制中心的 FQDN 地址 |
| Astra Trident | 已为基于SnapMirror的应用程序复制安装Astra Trident 22.01或更高版本并进行了配置Astra Trident 22.07或更高版本已为Kubernetes 1.25集群安装Astra Trident 22.10或更高版本(必须先升级到Astra Trident 22.10、然后再升级到Kubernetes 1.25) |

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。

- * 映像注册表 * ：您必须具有可将 Astra 控制中心构建映像推送到的现有私有 Docker 映像注册表。您需要提供要将映像上传到的映像注册表的 URL 。
- * Astra Trident / ONTAP 配置*：
  - 您需要在集群上至少配置一个Astra Trident存储类。如果配置了默认存储类、请确保它是唯一具有默认指定的存储类。
  - 确保集群中的工作节点已配置适当的存储驱动程序、以便Pod可以与后端存储进行交互。Astra 控制中心支持由 Astra Trident 提供的以下 ONTAP 驱动程序：
    - ontap-NAS
    - ontap-san
    - ontap-san-economy.(不支持应用程序复制)

## 支持的存储后端

Astra 控制中心支持以下存储后端。

- NetApp ONTAP 9.5或更高版本的AFF 、FAS 和ASA 系统
- NetApp ONTAP 9.8或更高版本的AFF 、FAS 和ASA 系统、用于基于SnapMirror的应用程序复制
- NetApp ONTAP Select 9.5或更高版本
- 适用于基于SnapMirror的应用程序复制的NetApp ONTAP Select 9.8或更高版本
- NetApp Cloud Volumes ONTAP 9.5或更高版本

要使用Astra控制中心、请根据您需要完成的任务、验证您是否具有以下ONTAP 许可证：

- FlexClone
- SnapMirror：可选。只有在使用SnapMirror技术复制到远程系统时才需要。请参见 "SnapMirror许可证信息"。

- S3许可证：可选。只有ONTAP S3存储分段才需要

要检查ONTAP 系统是否具有所需的许可证、请参见 "管理ONTAP 许可证"。

## 访问 Internet

您应确定是否可以从外部访问 Internet 。否则，某些功能可能会受到限制，例如从 NetApp Cloud Insights 接收监控和指标数据或向发送支持包 "NetApp 支持站点"。

## 许可证

要实现全部功能， Astra 控制中心需要获得 Astra 控制中心许可证。从 NetApp 获取评估版许可证或完整许可证。您需要一个许可证来保护应用程序和数据。请参见 "Astra控制中心功能" 了解详细信息。

您可以使用评估版许可证试用Astra控制中心、这样、您可以在自下载此许可证之日起的90天内使用Astra控制中心。您可以通过注册注册注册免费试用版 "此处"。

要设置许可证、请参见 "使用 90 天评估许可证"。

要了解有关许可证工作原理的详细信息，请参见 "许可"。

有关ONTAP 存储后端所需许可证的详细信息、请参见 "支持的存储后端"。

## 内部 Kubernetes 集群的传入

您可以选择 Astra 控制中心使用的网络传入类型。默认情况下， Astra 控制中心会将 Astra 控制中心网关（ service/traefik ）部署为集群范围的资源。如果您的环境允许使用服务负载平衡器，则 Astra 控制中心也支持使用服务负载平衡器。如果您希望使用服务负载平衡器、但尚未配置此平衡器、则可以使用MetalLB负载平衡器自动为该服务分配外部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。

> 负载平衡器应使用与Astra控制中心工作节点IP地址位于同一子网中的IP地址。

> 如果要在Tanzu Kubernetes Grid集群上托管Astra控制中心、请使用 `kubectl get nsxlbmonitors -A` 命令以查看是否已将服务监控器配置为接受传入流量。如果存在一个，则不应安装 MetalLB ，因为现有服务监控器将覆盖任何新的负载平衡器配置。

有关详细信息，请参见 "设置传入以进行负载平衡"。

## 网络要求

托管 Astra 控制中心的操作环境使用以下 TCP 端口进行通信。您应确保允许这些端口通过任何防火墙，并将防火墙配置为允许来自 Astra 网络的任何 HTTPS 传出流量。某些端口需要在托管 Astra 控制中心的环境与每个受管集群之间进行双向连接（请在适用时注明）。

> 您可以在双堆栈 Kubernetes 集群中部署 Astra 控制中心，而 Astra 控制中心则可以管理为双堆栈操作配置的应用程序和存储后端。有关双堆栈集群要求的详细信息，请参见 "Kubernetes 文档"。

| 源 | 目标 | **Port** | 协议 | 目的 |
| --- | --- | --- | --- | --- |
| 客户端 PC | Astra 控制中心 | 443. | HTTPS | UI / API 访问 - 确保托管 Astra 控制中心的集群与每个受管集群之间的此端口是双向开放的 |
| 指标使用者 | Astra 控制中心工作节点 | 9090 | HTTPS | 指标数据通信—确保每个受管集群都可以访问托管 Astra 控制中心的集群上的此端口 （需要双向通信） |
| Astra 控制中心 | 托管 Cloud Insights 服务 | 443. | HTTPS | Cloud Insights 通信 |
| Astra 控制中心 | Amazon S3 存储分段提供商 | 443. | HTTPS | Amazon S3 存储通信 |
| Astra 控制中心 | NetApp AutoSupport | 443. | HTTPS | NetApp AutoSupport 通信 |

## 支持的 Web 浏览器

Astra 控制中心支持最新版本的 Firefox ， Safari 和 Chrome ，最小分辨率为 1280 x 720 。

## 应用程序集群的其他要求

如果您计划使用以下Astra控制中心功能、请记住这些要求：

- 应用程序集群要求： "集群管理要求"
  - 受管应用程序要求： "应用程序管理要求"
  - 应用程序复制的其他要求： "复制前提条件"

## Google Anthos集群要求

在Google Anthos集群上托管Astra控制中心时、请注意、Google Anthos默认包括MetalLB负载平衡器和Istio入口网关服务、使您可以在安装期间轻松使用Astra控制中心的通用入口功能。请参见 "配置 Astra 控制中心" 了解详细信息。

## VMware Tanzu Kubernetes Grid 集群要求

在 VMware Tanzu Kubernetes Grid （ TKG ）或 Tanzu Kubernetes Grid Integrated Edition （ TKGi ）集群上托管 Astra Control Center 时，请记住以下注意事项。

- 在任何要由 Astra Control 管理的应用程序集群上禁用 TKG 或 TKGi 默认存储类强制实施。您可以通过编辑来执行此操作 `TanzuKubernetesCluster` 命名空间集群上的资源。
- 在 TKG 或 TKGi 环境中部署 Astra 控制中心时，请注意 Astra Trident 的特定要求。有关详细信息，请参见 "Astra Trident 文档"。

> 默认的 VMware TKG 和 TKGi 配置文件令牌将在部署后 10 小时过期。如果您使用的是 Tanzu 产品组合，则必须使用未过期的令牌生成 Tanzu Kubernetes 集群配置文件，以防止 Astra 控制中心与受管应用程序集群之间出现连接问题。有关说明，请访问 "VMware NSX-T 数据中心产品文档。"

### 下一步行动

查看 "快速入门" 概述。

# Astra 控制中心快速入门

下面简要介绍了开始使用Astra控制中心所需的步骤。每个步骤中的链接将转到一个页面，其中提供了更多详细信息。

1 查看 **Kubernetes** 集群要求

确保您的环境满足这些要求。

- Kubernetes集群*
- "确保您的环境满足运营环境要求"
- "为内部Kubernetes集群的负载平衡配置传入"

存储集成

- "确保您的环境包含Astra Trident支持的版本"
- "准备工作节点"
- "配置Astra Trident存储后端"
- "配置Astra Trident存储类"
- "安装Astra Trident卷快照控制器"
- "创建卷快照类"
- ONTAP 凭据*
- "配置ONTAP 凭据"

2 下载并安装**Astra**控制中心

完成这些安装任务。

- "从NetApp 支持站点 评估下载页面下载Astra控制中心"
- 获取NetApp许可证文件：
  - "如果您正在评估Astra控制中心、请下载评估版许可证文件"
  - "如果您已购买Astra Control Center、请生成许可证文件"
- "安装 Astra 控制中心"

- "执行其他可选配置步骤"

3 完成一些初始设置任务

完成一些基本任务以开始使用。

- "添加许可证"
- "准备用于集群管理的环境"
- "添加集群"
- "添加存储后端"
- "添加存储分段"

4 使用 **Astra** 控制中心

设置完Astra控制中心后、您接下来可以执行以下操作。您可以使用Astra Control用户界面(UI)或 "Astra Control API"。

- "管理应用程序"
- "保护应用程序"：配置保护策略以及复制、克隆和迁移应用程序。
- "管理帐户"：用户、角色、LDAP、凭据等
- "(可选)连接到Cloud Insights"：查看有关系统运行状况的指标。

### 有关详细信息 ...

- "Astra Control API"
- "升级 Astra 控制中心"
- "获取有关Astra Control的帮助"

# 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"

根据您的环境、安装Astra控制中心后可能需要进行其他配置：

- "安装后配置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。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。

其他安装过程

- 使用**RedHat OpenShift OperatorHub**安装：使用此 "备用操作步骤" 使用OperatorHub在OpenShift上安装Astra控制中心。
- 使用**Cloud Volumes ONTAP** 后端在公有 云中安装：使用 "这些过程" 在带有Cloud Volumes ONTAP存储后端的Amazon Web Services (AWS)、Google云平台(GCP)或Microsoft Azure中安装Astra控制中心。

有关Astra控制中心安装过程的演示、请参见 "此视频"。

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 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控制中心就不会尝试安装自己的证书管理器。默认情况下、Astra控制中心会在安装期间安装自己的证书管理器。

关于此任务

Astra控制中心安装过程可帮助您执行以下操作：

- 将Astra组件安装到中 `netapp-acc` (或自定义命名的)命名空间。
- 创建默认的Astra Control所有者管理员帐户。
- 建立管理用户电子邮件地址和默认初始设置密码。系统会为此用户分配首次登录到UI所需的所有者角色。
- 确定所有Astra控制中心Pod均正在运行。
- 安装Astra控制中心UI。

> 请勿删除Astra Control Center运算符(例如、 `kubectl delete -f astra_control_center_operator_deploy.yaml`)、以避免删除Pod。

步骤

要安装 Astra 控制中心，请执行以下步骤：

- 下载并提取Astra控制中心
- 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

下载并提取**Astra**控制中心

1. 转至 "Astra Control Center评估下载页面" 页面。
2. 下载包含Astra Control Center的软件包 (`astra-control-center-[version].tar.gz`）。
3. (建议但可选)下载Astra控制中心的证书和签名包 (`astra-control-center-certs-[version].tar.gz`)以验证捆绑包的签名：

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

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

此时将显示输出 `Verified OK` 验证成功后。

4. 从Astra Control Center捆绑包中提取映像：

```
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可为不同的CPU架构和操作系统提供插件二进制文件。在执行此任务之前、您需要了解您的CPU和操作系统。

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：

> kubectl插件库是tar包的一部分、并会解压缩到文件夹中 `kubectl-astra`。

```
ls kubectl-astra/
```

2. 将正确的二进制文件移动到当前路径并重命名为 `kubectl-astra`：

```
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tarball的根目录。您应看到此文件和目录：

   ```
   acc.manifest.bundle.yaml
   acc/
   ```

2. 将Astra Control Center映像目录中的软件包映像推送到本地注册表。在运行之前、请进行以下替换 `push-images` 命令：

   - 将<BUNDLE_FILE> 替换为Astra Control捆绑包文件的名称 (`acc.manifest.bundle.yaml`）。
   - 将<MY_FULL_REGISTRY_PATH> 替换为Docker存储库的URL；例如 "<a href="https://<docker-registry>"" class="bare">https://<docker-registry>"</a>。
   - 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_FULL_REGISTRY_PATH> -u <MY_REGISTRY_USER> -p
     <MY_REGISTRY_TOKEN>
     ```

**Podman**

1. 更改为tarball的根目录。您应看到此文件和目录：

   ```
   acc.manifest.bundle.yaml
   acc/
   ```

2. 登录到注册表：

   ```
   podman login <YOUR_REGISTRY>
   ```

3. 准备并运行以下针对您使用的Podman版本自定义的脚本之一。将<MY_FULL_REGISTRY_PATH> 替换为包含任何子目录的存储库的URL。

   ```
   <strong>Podman 4</strong>
   ```

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=22.11.0-82
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/${
PACKAGEVERSION}/${astraImageNoPath}
done
```

```
<strong>Podman 3</strong>
```

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=22.11.0-82
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/${
PACKAGEVERSION}/${astraImageNoPath}
done
```

> 根据您的注册表配置、此脚本创建的映像路径应类似于以下内容：
> https://netappdownloads.jfrog.io/docker-astra-control-prod/netapp/astra/acc/22.11.0-82/image:version

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]
```

> 在完成安装之前、请确保您的KUBECONFIG指向要安装Astra控制中心的集群。KUBECONFIG只能包含一个上下文。

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/22.11.0-82`）。

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

   响应示例：

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

   > 如果在生成密钥后删除命名空间、请重新创建命名空间、然后重新生成命名空间的密钥。

   c. 创建 `netapp-acc` (或自定义命名的)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` (或自定义命名的)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
```

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。

```
<strong>astra_control_center_operator_deploy.yaml</strong>
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    control-plane: controller-manager
  name: acc-operator-controller-manager
  namespace: netapp-acc-operator
```

```
spec:
  replicas: 1
  selector:
    matchLabels:
      control-plane: controller-manager
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        control-plane: controller-manager
    spec:
      containers:
      - args:
        - --secure-listen-address=0.0.0.0:8443
        - --upstream=http://127.0.0.1:8080/
        - --logtostderr=true
        - --v=10
        image: [your_registry_path]/kube-rbac-proxy:v4.8.0
        name: kube-rbac-proxy
        ports:
        - containerPort: 8443
          name: https
      - args:
        - --health-probe-bind-address=:8081
        - --metrics-bind-address=127.0.0.1:8080
        - --leader-elect
        env:
        - name: ACCOP_LOG_LEVEL
          value: "2"
        - name: ACCOP_HELM_INSTALLTIMEOUT
          value: 5m
        image: [your_registry_path]/acc-operator:[version x.y.z]
        imagePullPolicy: IfNotPresent
        livenessProbe:
          httpGet:
            path: /healthz
            port: 8081
          initialDelaySeconds: 15
          periodSeconds: 20
        name: manager
        readinessProbe:
          httpGet:
            path: /readyz
            port: 8081
          initialDelaySeconds: 5
```

```
          periodSeconds: 10
        resources:
          limits:
            cpu: 300m
            memory: 750Mi
          requests:
            cpu: 100m
            memory: 75Mi
        securityContext:
          allowPrivilegeEscalation: false
      imagePullSecrets: []
      securityContext:
        runAsUser: 65532
      terminationGracePeriodSeconds: 10
```

3. 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.yaml`)进行帐户、支持、注册表和其他必要配置：

```
vim astra_control_center.yaml
```

> 以下步骤将提供一个标注的YAML示例。

2. 修改或确认以下设置：

**<code>accountName</code>**

| 正在设置 ... | 指导 | **Type** | 示例 |
|---|---|---|---|
| `accountName` | 更改 `accountName` 字符串、表示要与Astra Control Center帐户关联的名称。只能有一个accountName。 | string | `Example` |

**<code>astraVersion</code>**

| 正在设置 ... | 指导 | **Type** | 示例 |
|---|---|---|---|
| `astraVersion` | 要部署的Astra控制中心版本。无需对此设置执行任何操作、因为此值将预先填充。 | string | `22.11.0-82` |

**<code>astraAddress</code>**

| 正在设置 ... | 指导 | Type | 示例 |
|---|---|---|---|
| `astraAddress` | 更改 `astraAddress` 指向要在浏览器中访问Astra控制中心的FQDN (建议)或IP地址的字符串。此地址用于定义如何在数据中心中找到Astra控制中心、并且与您在完成后从负载平衡器配置的FQDN或IP地址相同 "Astra 控制中心要求"。注意：请勿使用 `http://` 或 `https://` 地址中。复制此 FQDN 以在中使用后续步骤。 | string | `astra.example.com` |

**<code>autoSupport</code>**

您在本节中的选择将决定您是否要参与NetApp主动支持应用程序NetApp Active IQ 以及数据的发送位置。需要互联网连接(端口442)、所有支持数据均会匿名化。

| 正在设置 ... | 使用 ... | 指导 | Type | 示例 |
|---|---|---|---|---|
| `autoSupport.enrolled` | 两者之一 `enrolled` 或 `url` 必须选择字段 | 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。的设置 `true` 允许将匿名数据发送给NetApp以供支持。默认选择为 `false` 和表示不会向NetApp发送任何支持数据。 | 布尔值 | `false` (此值为默认值) |
| `autoSupport.url` | 两者之一 `enrolled` 或 `url` 必须选择字段 | 此URL用于确定匿名数据的发送位置。 | string | `https://support.netapp.com/asupprod/post/1.0/postAsup` |

**<code>email</code>**

| 正在设置 ... | 指导 | Type | 示例 |
|---|---|---|---|
| `email` | 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。此电子邮件地址将用作初始帐户的用户名、用于登录到UI、并在Astra Control中收到事件通知。 | string | `admin@example.com` |

**<code>firstName</code>**

| 正在设置 ... | 指导 | Type | 示例 |
|---|---|---|---|
| `firstName` | 与Astra帐户关联的默认初始管理员的名字。首次登录后、此处使用的名称将显示在用户界面的标题中。 | string | `SRE` |

**<code>LastName</code>**

| 正在设置 ... | 指导 | Type | 示例 |
|---|---|---|---|
| `lastName` | 与Astra帐户关联的默认初始管理员的姓氏。首次登录后、此处使用的名称将显示在用户界面的标题中。 | string | `Admin` |

**<code>imageRegistry</code>**

您在本节中的选择定义了托管Astra应用程序映像、Astra控制中心操作员和Astra控制中心Helm存储库的容器映像注册表。

| 正在设置 ... | 使用 ... | 指导 | Type | 示例 |
|---|---|---|---|---|
| `imageRegistry.name` | Required | 在中推送映像的映像注册表的名称 上一步。请勿使用 `http://` 或 `https://` 注册表名称。 | string | `example.registry.com/astra` |
| `imageRegistry.secret` | 如果您为输入的字符串、则为必填项 `imageRegistry.name' requires a secret.`<br><br>`IMPORTANT: If you are using a registry that does not require authorization, you must delete this` `secret` 行内 `imageRegistry` 否则安装将失败。 | 用于通过映像注册表进行身份验证的Kubernetes密钥的名称。 | string | `astra-registry-cred` |

**<code>storageClass</code>**

| 正在设置 ... | 指导 | Type | 示例 |
| --- | --- | --- | --- |
| `storageClass` | 更改 `storageClass` 价值来自 `ontap-gold` 安装所需的其他Trident storageClass资源。运行命令 `kubectl get sc` 以确定已配置的现有存储类。必须在清单文件中输入一个基于Trident的存储类 (`astra-control-center-<version>.manifest`)、并将用于Astra PV。如果未设置、则会使用默认存储类。注意：如果配置了默认存储类、请确保它是唯一具有默认标注的存储类。 | string | `ontap-gold` |

**<code>volumeReclaimPolicy</code>**

| 正在设置 ... | 指导 | Type | 选项 |
| --- | --- | --- | --- |
| `volumeReclaimPolicy` | 这将为Astra的PV设置回收策略。将此策略设置为 `Retain` 删除Astra后保留永久性卷。将此策略设置为 `Delete` 删除Astra后删除永久性卷。如果未设置此值、则会保留PV。 | string | • `Retain` (这是默认值)<br>• `Delete` |

**<code>ingressType</code>**

| 正在设置 ... | 指导 | Type | 选项 |
| --- | --- | --- | --- |
| `ingressType` | 请使用以下入口类型之一：**`Generic`** (`ingressType: "Generic"`)(默认)如果您正在使用另一个入口控制器或希望使用自己的入口控制器、请使用此选项。部署Astra控制中心后、您需要配置 "入口控制器" 以使用URL公开Astra控制中心。**`AccTraefik`** (`ingressType: "AccTraefik"`)如果您不想配置入口控制器、请使用此选项。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，您应将为 Astra 控制中心选择的 DNS 名称指向负载平衡的 IP 地址。注意：有关"loadbalancer"服务类型和入口的详细信息、请参见 "要求"。 | string | • `Generic` (这是默认值)<br>• `AccTraefik` |

**<code>astraResourcesScaler</code>**

| 正在设置 ... | 指导 | **Type** | 选项 |
| --- | --- | --- | --- |
| `astraResourcesScaler` | AstraControlCenter资源限制的扩展选项。默认情况下、Astra控制中心会进行部署、并为Astra中的大多数组件设置了资源请求。通过这种配置、Astra控制中心软件堆栈可以在应用程序负载和扩展性增加的环境中更好地运行。但是、在使用较小的开发或测试集群的情况下、CR字段为 `astraResourcesScalar` 可设置为 `Off`。此操作将禁用资源请求、并允许在较小的集群上部署。 | string | • `Default` (这是默认值)<br>• `Off` |

**<code>crds</code>**

您在本节中的选择决定了Astra控制中心应如何处理CRD。

| 正在设置 ... | 指导 | **Type** | 示例 |
|---|---|---|---|
| `crds.externalCertManager` | 如果使用外部证书管理器、请进行更改 `externalCertManager` to `true`。默认值 `false` 使Astra控制中心在安装期间安装自己的证书管理器CRD。CRD是集群范围的对象、安装它们可能会影响集群的其他部分。您可以使用此标志向Astra控制中心发出信号、指示这些CRD将由Astra控制中心以外的集群管理员安装和管理。 | 布尔值 | `False` (此值为默认值) |
| `crds.externalTraefik` | 默认情况下、Astra控制中心将安装所需的Traefik CRD。CRD是集群范围的对象、安装它们可能会影响集群的其他部分。您可以使用此标志向Astra控制中心发出信号、指示这些CRD将由Astra控制中心以外的集群管理员安装和管理。 | 布尔值 | `False` (此值为默认值) |

```
<strong>astra_control_center.yaml</strong>
```

```
apiVersion: astra.netapp.io/v1
kind: AstraControlCenter
metadata:
  name: astra
spec:
  accountName: "Example"
  astraVersion: "ASTRA_VERSION"
  astraAddress: "astra.example.com"
  autoSupport:
    enrolled: true
  email: "[admin@example.com]"
  firstName: "SRE"
  lastName: "Admin"
  imageRegistry:
    name: "[your_registry_path]"
    secret: "astra-registry-cred"
  storageClass: "ontap-gold"
  volumeReclaimPolicy: "Retain"
  ingressType: "Generic"
  astraResourcesScaler: "Default"
  additionalValues: {}
  crds:
    externalTraefik: false
    externalCertManager: false
```

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.yaml -n [netapp-acc or custom
   namespace]
   ```

   响应示例：

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

验证系统状态

您可以使用kubectl命令验证系统状态。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。

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-76d8d845c9-ggds2                1/1     Running   0
14m
activity-6cc67ff9f4-z48mr                     1/1     Running   2
(8m32s ago)   9m
api-token-authentication-7s67v                1/1     Running   0
8m56s
api-token-authentication-bplb4                1/1     Running   0
8m56s
api-token-authentication-p2c9z                1/1     Running   0
8m56s
asup-6cdfbc6795-md8vn                         1/1     Running   0
9m14s
authentication-9477567db-8hnc9                1/1     Running   0
7m4s
bucketservice-f4dbdfcd6-wqzkw                 1/1     Running   0
8m48s
cert-manager-bb756c7c4-wm2cv                  1/1     Running   0
14m
cert-manager-cainjector-c9bb86786-8wrf5       1/1     Running   0
14m
cert-manager-webhook-dd465db99-j2w4x          1/1     Running   0
14m
certificates-68dff9cdd6-kcvml                 1/1     Running   2
(8m43s ago)   9m2s
certificates-68dff9cdd6-rsnsb                 1/1     Running   0
9m2s
cloud-extension-69d48c956c-2s8dt              1/1     Running   3
(8m43s ago)   9m24s
cloud-insights-service-7c4f48b978-7gvlh       1/1     Running   3
(8m50s ago)   9m28s
composite-compute-7d9ff5f68-nxbhl             1/1     Running   0
8m51s
composite-volume-57b4756d64-nl66d             1/1     Running   0
9m13s
credentials-6dbc55f89f-qpzff                  1/1     Running   0
11m
entitlement-67bfb6d7-gl6kp                    1/1     Running   4
(8m33s ago)   9m38s
features-856cc4dccc-mxbdb                     1/1     Running   0
9m20s
fluent-bit-ds-4rtsp                           1/1     Running   0
```

```
6m54s
fluent-bit-ds-9rqll                          1/1     Running   0
6m54s
fluent-bit-ds-w5mp7                          1/1     Running   0
6m54s
graphql-server-7c7cc49776-jz2kn              1/1     Running   0
2m29s
identity-87c59c975-9jpnf                     1/1     Running   0
9m6s
influxdb2-0                                  1/1     Running   0
13m
keycloak-operator-84ff6d59d4-qcnmc           1/1     Running   0
7m1s
krakend-cbf6c7df9-mdtzv                      1/1     Running   0
2m30s
license-5b888b78bf-plj6j                     1/1     Running   0
9m32s
login-ui-846b4664dd-fz8hv                    1/1     Running   0
2m24s
loki-0                                       1/1     Running   0
13m
metrics-facade-779cc9774-n26rw               1/1     Running   0
9m18s
monitoring-operator-974db78f-pkspq           2/2     Running   0
6m58s
nats-0                                       1/1     Running   0
13m
nats-1                                       1/1     Running   0
13m
nats-2                                       1/1     Running   0
13m
nautilus-7bdc7ddc54-49tfn                    1/1     Running   0
7m50s
nautilus-7bdc7ddc54-cwc79                    1/1     Running   0
9m36s
openapi-5584ff9f46-gbrdj                     1/1     Running   0
9m17s
openapi-5584ff9f46-z9mzk                     1/1     Running   0
9m17s
packages-bfc58cc98-lpxq9                     1/1     Running   0
8m58s
polaris-consul-consul-server-0               1/1     Running   0
13m
polaris-consul-consul-server-1               1/1     Running   0
13m
polaris-consul-consul-server-2               1/1     Running   0
```

```
13m
polaris-keycloak-0                          1/1     Running   3
(6m15s ago)   6m56s
polaris-keycloak-1                          1/1     Running   0
4m22s
polaris-keycloak-2                          1/1     Running   0
3m41s
polaris-keycloak-db-0                       1/1     Running   0
6m56s
polaris-keycloak-db-1                       1/1     Running   0
4m23s
polaris-keycloak-db-2                       1/1     Running   0
3m36s
polaris-mongodb-0                           2/2     Running   0
13m
polaris-mongodb-1                           2/2     Running   0
13m
polaris-mongodb-2                           2/2     Running   0
12m
polaris-ui-5ccff47897-8rzgh                 1/1     Running   0
2m33s
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-6cb7bfc49b-p54xm             1/1     Running   1
(8m29s ago)   9m31s
storage-backend-metrics-5c77994586-kjn48    1/1     Running   0
8m52s
storage-provider-769fdc858c-62w54           1/1     Running   0
8m54s
task-service-9ffc484c5-kx9f4                1/1     Running   3
(8m44s ago)   9m34s
telegraf-ds-bphb9                           1/1     Running   0
6m54s
telegraf-ds-rtsm2                           1/1     Running   0
6m54s
telegraf-ds-s9h5h                           1/1     Running   0
6m54s
telegraf-rs-lbpv7                           1/1     Running   0
6m54s
telemetry-service-57cfb998db-zjx78          1/1     Running   1
(8m40s ago)   9m26s
tenancy-5d5dfbcf9f-vmbxh                    1/1     Running   0
```

```
9m5s
traefik-7b87c4c474-jmgp2                    1/1     Running   0
2m24s
traefik-7b87c4c474-t9k8x                    1/1     Running   0
2m24s
trident-svc-c78f5b6bd-nwdsq                 1/1     Running   0
9m22s
vault-controller-55bbc96668-c6425           1/1     Running   0
11m
vault-controller-55bbc96668-lq9n9           1/1     Running   0
11m
vault-controller-55bbc96668-rfkgg           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。如果日志中指示的集群注册失败、您可以尝试通过重新注册 "在UI中添加集群工作流" 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初始设置密码：

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

响应：

```
NAME     UUID                                   VERSION      ADDRESS
READY
astra    9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f   22.11.0-82   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

您可以设置一个Kubernetes入口控制器、用于管理对服务的外部访问。如果您使用的是默认值、则以下过程提供了入口控制器的设置示例 `ingressType: "Generic"` 在Astra Control Center自定义资源中 (`astra_control_center.yaml`）。如果指定、则不需要使用此操作步骤 `ingressType: "AccTraefik"`

在Astra Control Center自定义资源中 (`astra_control_center.yaml`）。

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。

设置步骤因所使用的入口控制器类型而异。Astra控制中心支持多种传入控制器类型。这些设置过程提供了以下传入控制器类型的示例步骤：

- 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。

**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. 在中部署入站资源 `netapp-acc` (或自定义命名的)命名空间 (`istio-Ingress.yaml` 在此示例中使用)：

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

5. 应用更改：

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

6. 检查入口状态：

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

响应：

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

7. 完成Astra控制中心安装。

**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 (`nginx-Ingress.yaml` 在此示例中使用)：

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

3. 应用更改：

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

NetApp建议将nginx控制器安装为部署、而不是安装 `daemonSet`。

**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FQDN (包括 `https://` 前缀) `astraAddress` 在中 `astra_control_center.yaml` CR时间 您安装了 Astra 控制中心。
2. 如果出现提示、请接受自签名证书。

   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.yaml` CR时间 您安装了 Astra 控制中心、后跟初始设置密码 (`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。

   如果这是您第一次登录、但您忘记了密码、并且尚未创建任何其他管理用户帐户、请联系 "NetApp 支持" 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 deploy/acc-operator-controller-manager -n netapp-acc-
operator -c manager -f
```

下一步行动

- (可选)根据您的环境、完成安装后操作 "配置步骤"。
- 执行以完成部署 "设置任务"。

=
: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

- 满足环境前提条件： "开始安装之前，请为 Astra Control Center 部署准备您的环境"。
- 运行状况良好的集群操作员和**API**服务：
  - 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：

    ```
    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：

    ```
    oc get apiservices
    ```

- * FQDN地址*：获取数据中心中Astra控制中心的FQDN地址。
- * OpenShift权限*：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。
- 已配置证书管理器：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。默认情况下、Astra控制中心会在安装期间安装自己的证书管理器。
- * Kubernetes入口控制器*：如果您的Kubernetes入口控制器负责管理对服务的外部访问、例如集群中的负载平衡、则需要将其设置为与Astra控制中心配合使用：
  a. 创建操作员命名空间：

    ```
    oc create namespace netapp-acc-operator
    ```

  b. "完成设置" 适用于您的入口控制器类型。

步骤

- 下载并提取Astra控制中心
- 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

下载并提取**Astra**控制中心

1. 转至 "Astra Control Center评估下载页面" 页面。
2. 下载包含Astra Control Center的软件包 (`astra-control-center-[version].tar.gz`）。
3. (建议但可选)下载Astra控制中心的证书和签名包 (`astra-control-center-certs-[version].tar.gz`)以验证捆绑包的签名：

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

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

此时将显示输出 `Verified OK` 验证成功后。

4. 从Astra Control Center捆绑包中提取映像：

```
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可为不同的CPU架构和操作系统提供插件二进制文件。在执行此任务之前、您需要了解您的CPU和操作系统。

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：

> kubectl插件库是tar包的一部分、并会解压缩到文件夹中 `kubectl-astra`。

```
ls kubectl-astra/
```

2. 将正确的二进制文件移动到当前路径并重命名为 `kubectl-astra`：

```
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tarball的根目录。您应看到此文件和目录：

   ```
   acc.manifest.bundle.yaml
   acc/
   ```

2. 将Astra Control Center映像目录中的软件包映像推送到本地注册表。在运行之前、请进行以下替换 `push-images` 命令：

   - 将<BUNDLE_FILE> 替换为Astra Control捆绑包文件的名称 (`acc.manifest.bundle.yaml`）。
   - 将<MY_FULL_REGISTRY_PATH> 替换为Docker存储库的URL；例如 "<a href="https://<docker-registry>"" class="bare">https://<docker-registry>"</a>。
   - 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_FULL_REGISTRY_PATH> -u <MY_REGISTRY_USER> -p
     <MY_REGISTRY_TOKEN>
     ```

**Podman**

1. 更改为tarball的根目录。您应看到此文件和目录：

   ```
   acc.manifest.bundle.yaml
   acc/
   ```

2. 登录到注册表：

   ```
   podman login <YOUR_REGISTRY>
   ```

3. 准备并运行以下针对您使用的Podman版本自定义的脚本之一。将<MY_FULL_REGISTRY_PATH> 替换为包含任何子目录的存储库的URL。

   ```
   <strong>Podman 4</strong>
   ```

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=22.11.0-82
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/${
PACKAGEVERSION}/${astraImageNoPath}
done
```

```
<strong>Podman 3</strong>
```

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=22.11.0-82
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed 's/Loaded
image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/${
PACKAGEVERSION}/${astraImageNoPath}
done
```

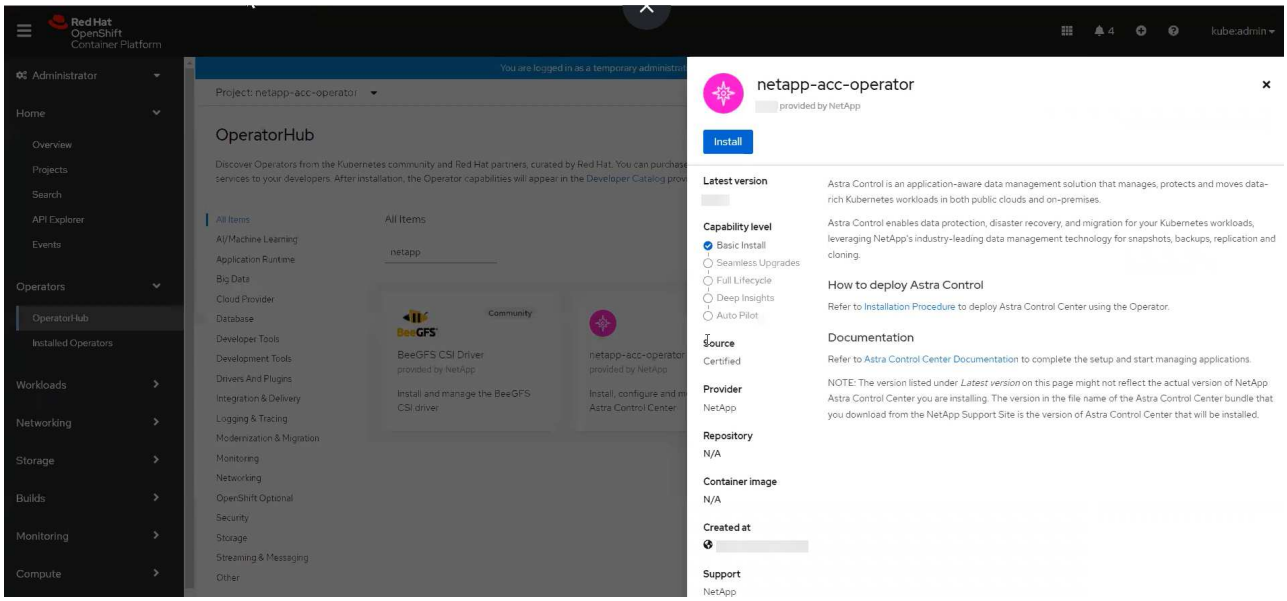
> 根据您的注册表配置、此脚本创建的映像路径应类似于以下内容：
> https://netappdownloads.jfrog.io/docker-astra-control-prod/netapp/astra/acc/22.11.0-82/image:version

找到操作员安装页面

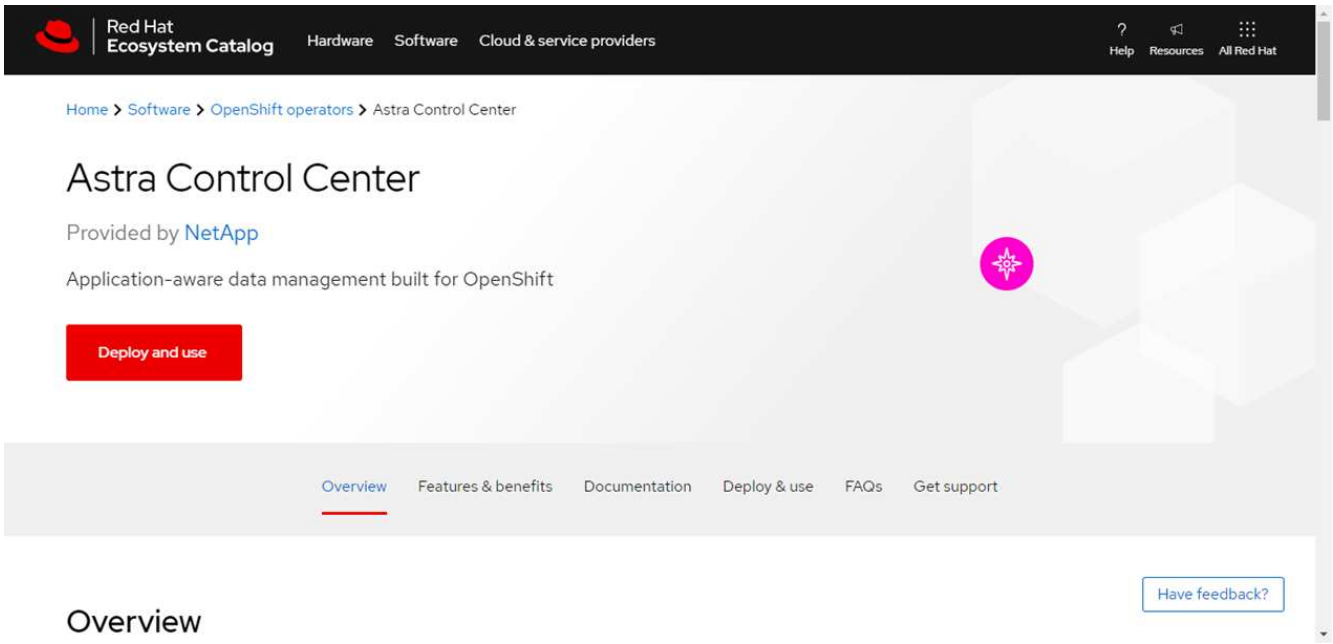
1. 要访问操作员安装页面，请完成以下过程之一：
   - 从 Red Hat OpenShift Web 控制台：
     i. 登录到 OpenShift 容器平台 UI 。

ii. 从侧面菜单中，选择 * 运算符 > OperatorHub * 。

iii. 搜索并选择NetApp Astra Control Center运算符。




- 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。

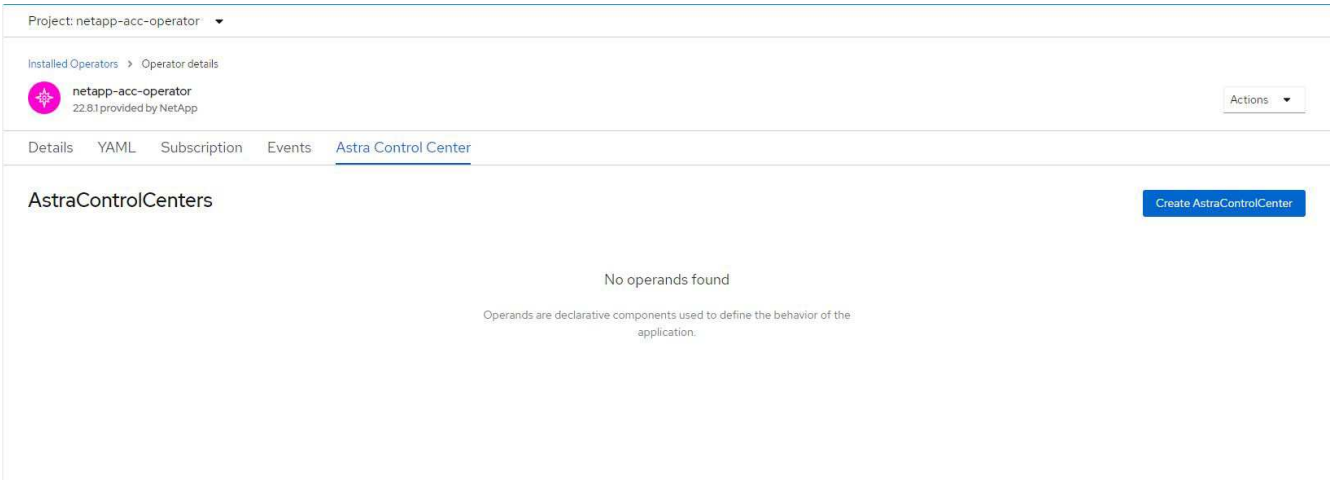
c. 选择 * 安装 * 。

> 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. 从Astra Control Center操作员的* Astra Control Center*选项卡中的控制台中、选择*创建AstraControlCenter*。




2. 完成 `Create AstraControlCenter` 表单字段：

   a. 保留或调整 Astra 控制中心名称。

   b. 为Astra控制中心添加标签。

   c. 启用或禁用自动支持。建议保留自动支持功能。

   d. 输入Astra控制中心FQDN或IP地址。请止步 `http://` 或 `https://` 在地址字段中。

   e. 输入Astra控制中心版本；例如22.04.1。

   f. 输入帐户名称，电子邮件地址和管理员姓氏。

   g. 选择的卷回收策略 `Retain`， `Recycle`或 `Delete`。默认值为 `Retain`。

   h. 选择入口类型：

      - **`Generic`** (`ingressType: "Generic"`)(默认)

        如果您正在使用另一个入口控制器或希望使用您自己的入口控制器、请使用此选项。部署Astra控制中心后、您需要配置 "入口控制器" 以使用URL公开Astra控制中心。

      - **`AccTraefik`** (`ingressType: "AccTraefik"`)

        如果您不希望配置入口控制器、请使用此选项。这将部署Astra控制中心 `traefik` 网关作

为Kubernetes的"loadbalancer"类型服务。

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"。

a. 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://` 在地址字段中。
b. 如果您使用的映像注册表需要身份验证、请输入映像密钥。

> 如果您使用的注册表需要身份验证、 在集群上创建密钥。

c. 输入管理员的名字。
d. 配置资源扩展。
e. 提供默认存储类。

> 如果配置了默认存储类、请确保它是唯一具有默认标注的存储类。

f. 定义 CRD 处理首选项。

3. 选择YAML视图以查看您选择的设置。
4. 选择 … `Create`。

创建注册表密钥

如果您使用的注册表需要身份验证、请在OpenShift集群上创建一个密钥、然后在中输入该密钥名称 `Create AstraControlCenter` 表单字段。

1. 为Astra控制中心操作员创建命名空间：

```
oc create ns [netapp-acc-operator or custom namespace]
```

2. 在此命名空间中创建密钥：

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

> Astra Control仅支持Docker注册表机密。

3. 完成中的其余字段 创建AstraControlCenter表单字段。

下一步行动

完成 "剩余步骤" 要验证是否已成功安装Astra控制中心、请设置一个入口控制器(可选)并登录到UI。此外、您还需要执行 "设置任务" 完成安装后。

## 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满足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 BlueXP** 中作为**Kubernetes**集群发现的一部分安装、以前称为**Cloud Manager)** | 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 。<br>ⓘ 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。 |
| **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 BlueXP (以前称为Cloud Manager)时创建的。这些功能由 Astra Trident 提供：<br>• `vsaworkingenvironment-<>-ha-nas csi.trident.netapp.io`<br>• `vsaworkingenvironment-<>-ha-san csi.trident.netapp.io`<br>• `vsaworkingenvironment-<>-single-nas csi.trident.netapp.io`<br>• `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。

ⓘ 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. 配置适用于AWS的NetApp BlueXP。
5. 安装适用于AWS的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 BlueXP (以前称为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"。

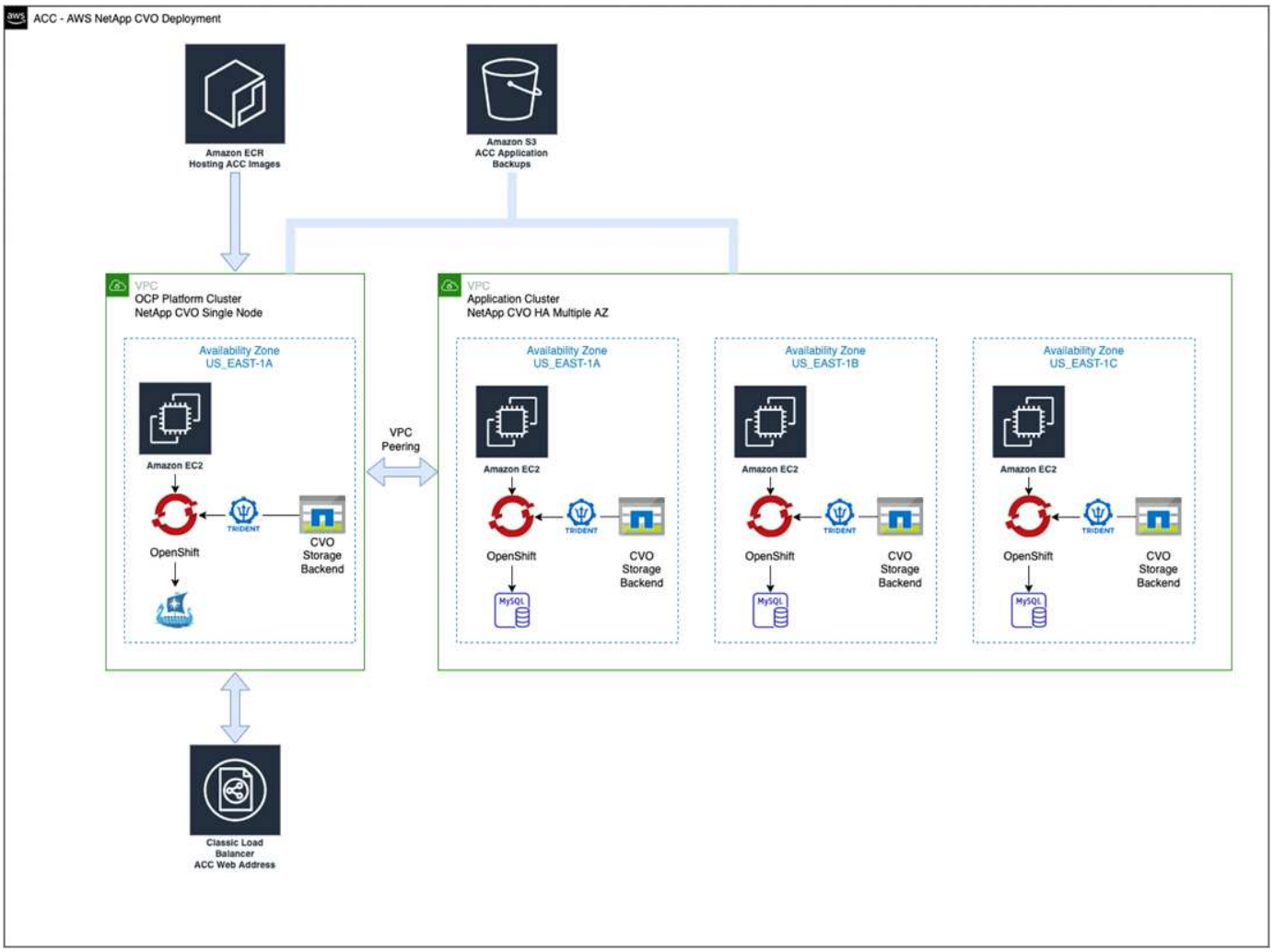
1. 创建AWS虚拟网络。
2. 查看 EC2 计算实例。这可以是 AWS 中的裸机服务器或 VM 。
3. 如果实例类型尚未与主节点和工作节点的 Astra 最低资源要求匹配，请更改 AWS 中的实例类型以满足 Astra 要求。请参见 "Astra 控制中心要求"。
4. 至少创建一个 AWS S3 存储分段来存储备份。
5. 创建 AWS 弹性容器注册表（ ECR ）以托管所有 AccR 映像。

> 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：




配置适用于**AWS**的**NetApp BlueXP**

使用NetApp BlueXP (以前称为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BlueXP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BlueXP在AWS中创建连接器"

步骤

1. 将凭据添加到BlueXP。
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。

#### 安装适用于AWS的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满足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 (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 BlueXP中作为Kubernetes集群发现的一部分安装、以前称为Cloud Manager)** | 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 。<br>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控制中心支持在将ONTAP Kubernetes集群导入到NetApp BlueXP中时创建的以下Kubernetes存储类。这些功能由 Astra Trident 提供：<br>• `vsaworkingenvironment-<>-ha-nas csi.trident.netapp.io`<br>• `vsaworkingenvironment-<>-ha-san csi.trident.netapp.io`<br>• `vsaworkingenvironment-<>-single-nas csi.trident.netapp.io`<br>• `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. 为GCP配置NetApp BlueXP。
6. 安装适用于GCP的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 BlueXP (以前称为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。

#### 为GCP配置NetApp BlueXP

使用NetApp BlueXP (以前称为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BlueXP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BlueXP。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 "从BlueXP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。

#### 安装适用于GCP的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满足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 BlueXP** 中作为**Kubernetes**集群发现的一部分安装**)** | 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 。<br>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控制中心支持在将ONTAP Kubernetes集群导入到NetApp BlueXP中时创建的以下Kubernetes存储类。这些功能由 Astra Trident 提供：<br>• `vsaworkingenvironment-<>-ha-nas csi.trident.netapp.io`<br>• `vsaworkingenvironment-<>-ha-san csi.trident.netapp.io`<br>• `vsaworkingenvironment-<>-single-nas csi.trident.netapp.io`<br>• `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。

#### 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. 为Azure配置NetApp BlueXP (以前称为Cloud Manager)。
6. 安装和配置适用于Azure的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，请参见以下内容：

- "在 Azure 上安装 OpenShift 集群"。
- "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 BlueXP 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。

#### 为Azure配置NetApp BlueXP (以前称为Cloud Manager)

使用BlueXP (以前称为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BlueXP文档完成以下步骤。请参见 "Azure中的BlueXP入门"。

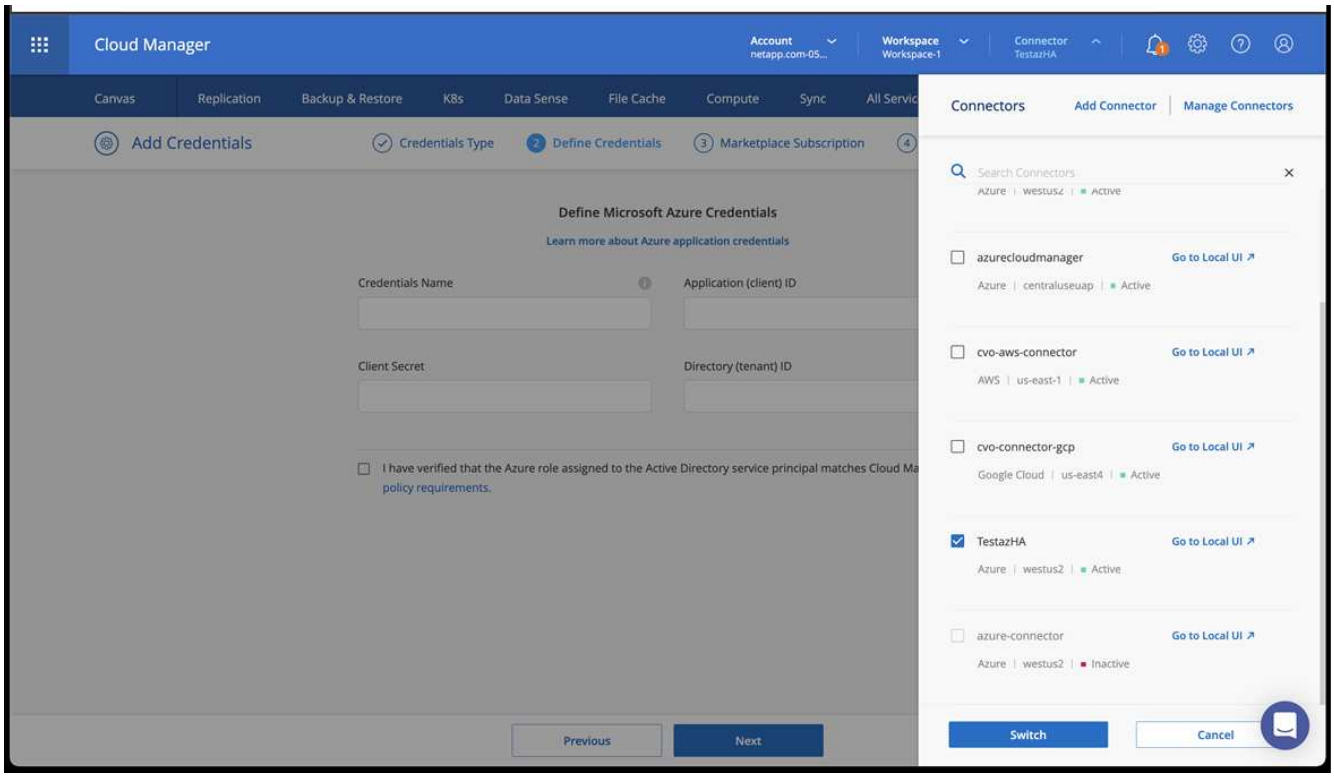
您需要的内容

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BlueXP。
2. 添加适用于 Azure 的连接器。请参见 "BlueXP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 。

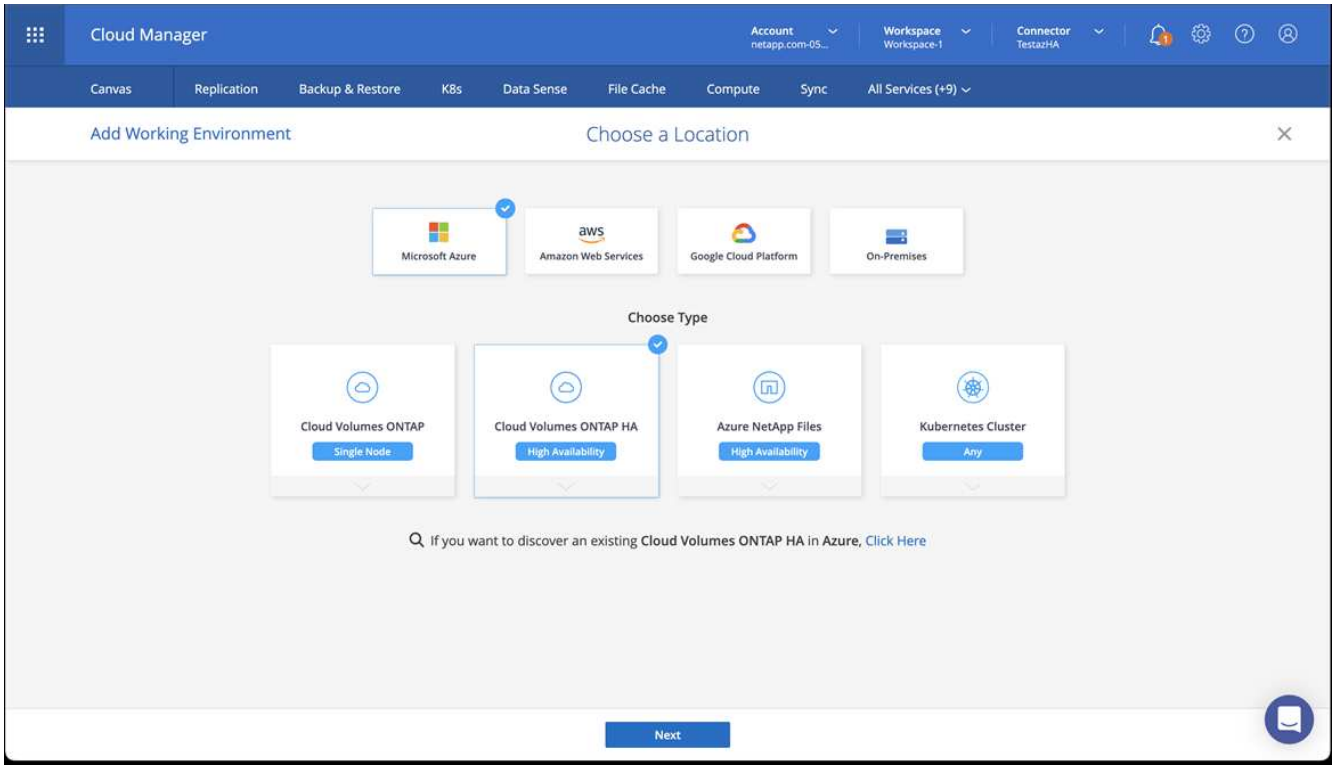
      请参见 "从BlueXPr.在Azure中创建连接器"。
3. 确保连接器正在运行，然后切换到该连接器。




4. 为您的云环境创建一个工作环境。

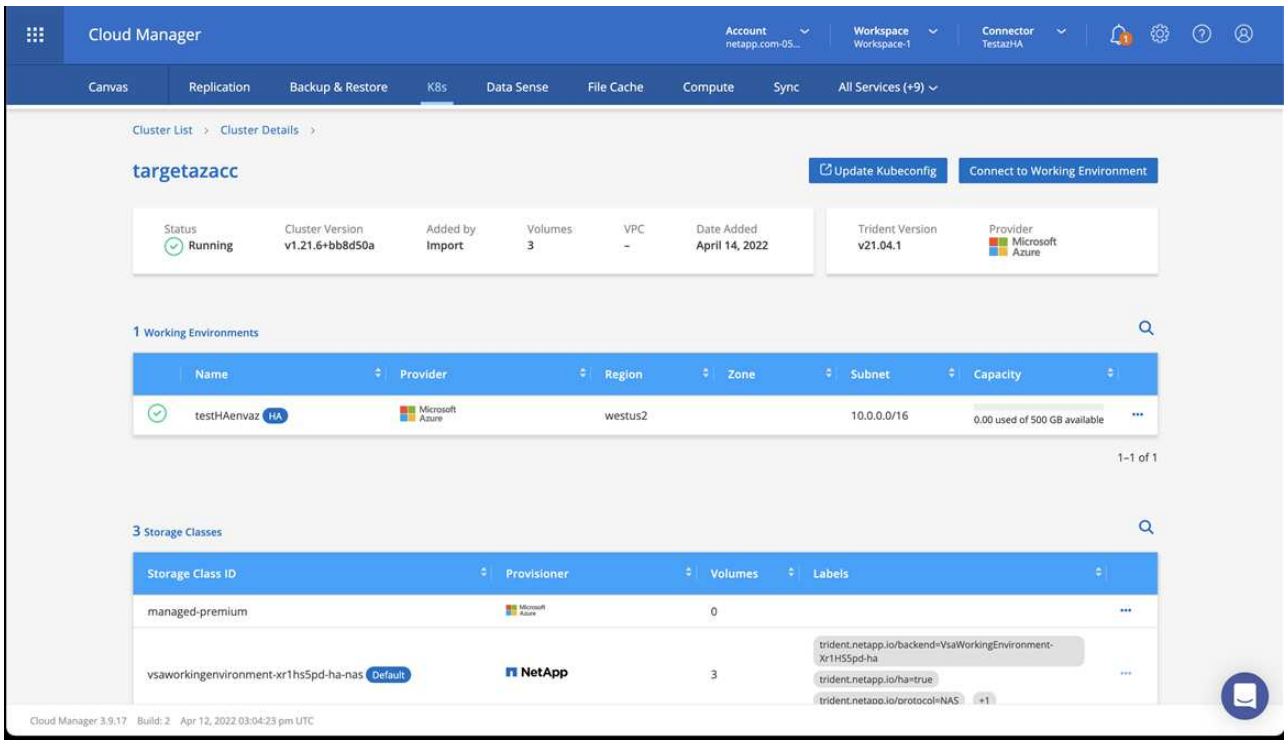
   a. 位置： "Microsoft Azure" 。

   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。

安装和配置适用于**Azure**的**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"。

=
:allow-uri-read:

# 设置 Astra 控制中心

安装 Astra 控制中心，登录到 UI 并更改密码后，您将需要设置许可证，添加集群，管理存储以及添加存储分段。

任务

- 添加 Astra 控制中心的许可证

- 使用Astra Control准备用于集群管理的环境
- [添加集群]
- [添加存储后端]
- [添加存储分段]

## 添加 Astra 控制中心的许可证

您可以使用Astra Control UI或添加新许可证 "API" 获得完整的 Astra 控制中心功能。如果没有许可证，则只能使用 Astra 控制中心来管理用户和添加新集群。

Astra控制中心许可证使用Kubernetes CPU单元测量CPU资源、并计算分配给所有受管Kubernetes集群的工作节点的CPU资源。许可证基于vCPU使用量。有关如何计算许可证的详细信息、请参见 "许可"。

> 如果您的安装增长到超过许可的 CPU 单元数，则 Astra 控制中心将阻止您管理新应用程序。超过容量时，将显示警报。

> 要更新现有评估版或完整许可证、请参见 "更新现有许可证"。

您需要的内容

- 访问新安装的Astra Control Center实例。
- 管理员角色权限。
- 答 "NetApp 许可证文件" (nlf)。

步骤

1. 登录到 Astra 控制中心 UI 。
2. 选择 * 帐户 * > * 许可证 * 。
3. 选择 * 添加许可证 * 。
4. 浏览到您下载的许可证文件 （ NLF ） 。
5. 选择 * 添加许可证 * 。
    - 帐户 * > * 许可证 * 页面显示许可证信息，到期日期，许可证序列号，帐户 ID 和使用的 CPU 单元。

> 如果您拥有评估版许可证、并且不向AutoSupport 发送数据、请确保存储您的帐户ID、以避免在Astra控制中心发生故障时丢失数据。

## 使用Astra Control准备用于集群管理的环境

在添加集群之前、应确保满足以下前提条件。您还应运行资格检查、以确保集群已准备好添加到Astra控制中心并创建集群管理角色。

您需要的内容

- 确保集群中的工作节点已配置适当的存储驱动程序、以便Pod可以与后端存储进行交互。
- 您的环境符合 "操作环境要求" 适用于Astra Trident和Astra控制中心。
- 一个版本的Astra Trident "受Astra控制中心支持" 已安装：

> 您可以 "部署Astra Trident" 使用Trident运算符(手动或使用Helm图表)或 `tridentctl`。在安装或升级Astra Trident之前、请查看 "支持的前端、后端和主机配置"。

  - 已配置**Trident**存储后端：至少必须有一个Astra Trident存储后端 "已配置" 在集群上。
  - 已配置**Trident**存储类：至少必须有一个Astra Trident存储类 "已配置" 在集群上。如果配置了默认存储类、请确保它是唯一具有默认标注的存储类。
  - 已安装并配置* Astra Trident卷快照控制器和卷快照类*：卷快照控制器必须为 "已安装" 以便可以在Astra Control中创建快照。至少一个Astra Trident `VolumeSnapshotClass` 已经 "设置" 由管理员执行。
- * Kubeconfig accessible*：您可以访问 "cluster kubeconfig" 这仅包括一个上下文元素。
- * ONTAP 凭据*：您需要在备用ONTAP 系统上设置ONTAP 凭据以及超级用户和用户ID、以便使用Astra控制中心备份和还原应用程序。

  在ONTAP 命令行中运行以下命令：

  ```
  export-policy rule modify -vserver <storage virtual machine name>
  -policyname <policy name> -ruleindex 1 -superuser sys
  export-policy rule modify -vserver <storage virtual machine name>
  -policyname <policy name> -ruleindex 1 -anon 65534
  ```

- *仅Rancher *：在Rancher环境中管理应用程序集群时、请修改Rancher提供的kubeconfig文件中的应用程序集群默认上下文、以使用控制平面上下文、而不是Rancher API服务器上下文。这样可以减少 Rancher API 服务器上的负载并提高性能。

运行资格检查

运行以下资格检查，以确保您的集群已准备好添加到 Astra 控制中心。

步骤

1. 检查 Trident 版本。

   ```
   kubectl get tridentversions -n trident
   ```

   如果存在 Trident ，您将看到类似于以下内容的输出：

   ```
   NAME       VERSION
   trident    22.10.0
   ```

   如果 Trident 不存在，您将看到类似于以下内容的输出：

   ```
   error: the server doesn't have a resource type "tridentversions"
   ```

> 如果未安装 Trident 或安装的版本不是最新的，则需要先安装最新版本的 Trident ，然后再继续操作。请参见 "Trident 文档" 有关说明，请参见。

2. 确保Pod正在运行：

```
kubectl get pods -n trident
```

3. 确定存储类是否正在使用受支持的Trident驱动程序。配置程序名称应为 `csi.trident.netapp.io`。请参见以下示例：

```
kubectl get sc
```

响应示例：

```
NAME                    PROVISIONER             RECLAIMPOLICY
VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
ontap-gold (default)   csi.trident.netapp.io   Delete          Immediate
true                    5d23h
```

### 创建一个有限的集群角色kubeconfig

您可以选择为Astra控制中心创建有限的管理员角色。这不是Astra控制中心设置所需的操作步骤。此操作步骤 有助于创建一个单独的kubeconfig、以限制Astra Control对其管理的集群的权限。

#### 您需要的内容

在完成操作步骤 步骤之前、请确保您对要管理的集群具有以下信息：

- 已安装kubectl v1.23或更高版本
- kubectl访问要使用Astra控制中心添加和管理的集群

> 对于此操作步骤 、您不需要对运行Astra控制中心的集群进行kubectl访问。

- 要使用活动环境的集群管理员权限管理的集群的活动kubeconfig

步骤

1. 创建服务帐户：

   a. 创建名为的服务帐户文件 `astracontrol-service-account.yaml`。

      根据需要调整名称和命名空间。如果在此处进行了更改，则应在以下步骤中应用相同的更改。

```
<strong>astracontrol-service-account.yaml</strong>
```

+

```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: astracontrol-service-account
  namespace: default
```

   a. 应用服务帐户：

```
kubectl apply -f astracontrol-service-account.yaml
```

2. 使用Astra Control管理集群所需的最低权限创建一个有限的集群角色：

   a. 创建 `ClusterRole` 文件已调用 `astra-admin-account.yaml`。

      根据需要调整名称和命名空间。如果在此处进行了更改，则应在以下步骤中应用相同的更改。

```
<strong>astra-admin-account.yaml</strong>
```

+

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: astra-admin-account
rules:

# Get, List, Create, and Update all resources
# Necessary to backup and restore all resources in an app
- apiGroups:
  - '*'
  resources:
```

```
  - '*'
  verbs:
  - get
  - list
  - create
  - patch

# Delete Resources
# Necessary for in-place restore and AppMirror failover
- apiGroups:
  - ""
  - apps
  - autoscaling
  - batch
  - crd.projectcalico.org
  - extensions
  - networking.k8s.io
  - policy
  - rbac.authorization.k8s.io
  - snapshot.storage.k8s.io
  - trident.netapp.io
  resources:
  - configmaps
  - cronjobs
  - daemonsets
  - deployments
  - horizontalpodautoscalers
  - ingresses
  - jobs
  - namespaces
  - networkpolicies
  - persistentvolumeclaims
  - poddisruptionbudgets
  - pods
  - podtemplates
  - podsecuritypolicies
  - replicasets
  - replicationcontrollers
  - replicationcontrollers/scale
  - rolebindings
  - roles
  - secrets
  - serviceaccounts
  - services
  - statefulsets
  - tridentmirrorrelationships
```

```
  - tridentsnapshotinfos
  - volumesnapshots
  - volumesnapshotcontents
  verbs:
  - delete

# Watch resources
# Necessary to monitor progress
- apiGroups:
  - ""
  resources:
  - pods
  - replicationcontrollers
  - replicationcontrollers/scale
  verbs:
  - watch

# Update resources
- apiGroups:
  - ""
  - build.openshift.io
  - image.openshift.io
  resources:
  - builds/details
  - replicationcontrollers
  - replicationcontrollers/scale
  - imagestreams/layers
  - imagestreamtags
  - imagetags
  verbs:
  - update

# Use PodSecurityPolicies
- apiGroups:
  - extensions
  - policy
  resources:
  - podsecuritypolicies
  verbs:
  - use
```

a. 应用集群角色：

```
kubectl apply -f astra-admin-account.yaml
```

3. 为集群角色创建与服务帐户的集群角色绑定：

   a. 创建 `ClusterRoleBinding` 文件已调用 `astracontrol-clusterrolebinding.yaml`。

      根据需要调整创建服务帐户时修改的任何名称和命名空间。

```
<strong>astracontrol-clusterrolebinding.yaml</strong>
```

+

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: astracontrol-admin
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: astra-admin-account
subjects:
- kind: ServiceAccount
  name: astracontrol-service-account
  namespace: default
```

   a. 应用集群角色绑定：

```
kubectl apply -f astracontrol-clusterrolebinding.yaml
```

4. 列出服务帐户密码、替换 `<context>` 使用适用于您的安装的正确环境：

```
kubectl get serviceaccount astracontrol-service-account --context
<context> --namespace default -o json
```

输出的结尾应类似于以下内容：

```
"secrets": [
{ "name": "astracontrol-service-account-dockercfg-vhz87"},
{ "name": "astracontrol-service-account-token-r59kr"}
]
```

中每个元素的索引 `secrets` 阵列以0开头。在上面的示例中、是的索引 `astracontrol-service-account-dockercfg-vhz87` 将为0、并为创建索引 `astracontrol-service-account-token-r59kr` 将为1。在输出中，记下包含 "token" 一词的服务帐户名称的索引。

5. 按如下所示生成 kubeconfig：
   a. 创建 `create-kubeconfig.sh` 文件替换 `TOKEN_INDEX` 在以下脚本的开头、使用正确的值。

```
<strong>create-kubeconfig.sh</strong>
```

```
# Update these to match your environment.
# Replace TOKEN_INDEX with the correct value
# from the output in the previous step. If you
# didn't change anything else above, don't change
# anything else here.

SERVICE_ACCOUNT_NAME=astracontrol-service-account
NAMESPACE=default
NEW_CONTEXT=astracontrol
KUBECONFIG_FILE='kubeconfig-sa'

CONTEXT=$(kubectl config current-context)

SECRET_NAME=$(kubectl get serviceaccount ${SERVICE_ACCOUNT_NAME}
\
  --context ${CONTEXT} \
  --namespace ${NAMESPACE} \
  -o jsonpath='{.secrets[TOKEN_INDEX].name}')
TOKEN_DATA=$(kubectl get secret ${SECRET_NAME} \
  --context ${CONTEXT} \
  --namespace ${NAMESPACE} \
  -o jsonpath='{.data.token}')

TOKEN=$(echo ${TOKEN_DATA} | base64 -d)

# Create dedicated kubeconfig
# Create a full copy
kubectl config view --raw > ${KUBECONFIG_FILE}.full.tmp

# Switch working context to correct context
kubectl --kubeconfig ${KUBECONFIG_FILE}.full.tmp config use-
context ${CONTEXT}

# Minify
kubectl --kubeconfig ${KUBECONFIG_FILE}.full.tmp \
  config view --flatten --minify > ${KUBECONFIG_FILE}.tmp

# Rename context
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
```

```
  rename-context ${CONTEXT} ${NEW_CONTEXT}

# Create token user
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  set-credentials ${CONTEXT}-${NAMESPACE}-token-user \
  --token ${TOKEN}

# Set context to use token user
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  set-context ${NEW_CONTEXT} --user ${CONTEXT}-${NAMESPACE}-token
-user

# Set context to correct namespace
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  set-context ${NEW_CONTEXT} --namespace ${NAMESPACE}

# Flatten/minify kubeconfig
kubectl config --kubeconfig ${KUBECONFIG_FILE}.tmp \
  view --flatten --minify > ${KUBECONFIG_FILE}

# Remove tmp
rm ${KUBECONFIG_FILE}.full.tmp
rm ${KUBECONFIG_FILE}.tmp
```

b. 获取用于将其应用于 Kubernetes 集群的命令。

```
source create-kubeconfig.sh
```

6. (可选)将kubeconfig重命名为集群的有意义名称。

```
mv kubeconfig-sa YOUR_CLUSTER_NAME_kubeconfig
```

下一步是什么?

现在、您已确认满足了这些前提条件、您已做好准备 添加集群。

## 添加集群

要开始管理应用程序，请添加 Kubernetes 集群并将其作为计算资源进行管理。您必须为 Astra 控制中心添加一个集群，才能发现您的 Kubernetes 应用程序。

> 我们建议，在将其他集群添加到 Astra 控制中心进行管理之前，先由 Astra 控制中心管理其部署所在的集群。要发送 Kubemmetrics 数据和集群关联数据以获取指标和故障排除信息，必须对初始集群进行管理。

您需要的内容

- 在添加集群之前，请查看并执行必要的操作 前提条件任务。

步骤

1. 从信息板或集群菜单导航：
   - 从"Resource Summary"的"信息板"中、从"Clusters"窗格中选择"添加"。
   - 在左侧导航区域中、选择*集群*、然后从集群页面中选择*添加集群*。
2. 在打开的*添加集群*窗口中、上传 `kubeconfig.yaml` 归档或粘贴的内容 `kubeconfig.yaml` 文件

   > 。 `kubeconfig.yaml` 文件应仅包含一个集群的集群凭据*。

   > 创建自己的 `kubeconfig` file中、您只能定义*一*上下文元素。请参见 "Kubernetes 文档" 有关创建的信息 `kubeconfig` 文件。如果您使用为有限集群角色创建了kubeconfig 上述过程、请务必在此步骤中上传或粘贴kubeconfig。

3. 请提供凭据名称。默认情况下，凭据名称会自动填充为集群的名称。
4. 选择 * 下一步 * 。
5. 选择要用于此Kubernetes集群的默认存储类、然后选择*下一步*。

   > 您应选择一个由 ONTAP 存储提供支持的 Trident 存储类。

6. 查看相关信息、如果一切正常、请选择*添加*。

结果

集群将进入*正在发现*状态、然后更改为*运行状况良好*。现在、您正在使用Astra控制中心管理集群。

> 添加要在 Astra 控制中心中管理的集群后，部署监控操作员可能需要几分钟的时间。在此之前，通知图标将变为红色并记录一个 * 监控代理状态检查失败 * 事件。您可以忽略此问题，因为当 Astra 控制中心获得正确状态时，问题描述将解析。如果问题描述 在几分钟内未解析、请转至集群并运行 `oc get pods -n netapp-monitoring` 作为起点。您需要查看监控操作员日志以调试此问题。

## 添加存储后端

您可以将现有ONTAP 存储后端添加到Astra控制中心以管理其资源。

通过将 Astra Control 中的存储集群作为存储后端进行管理，您可以在永久性卷（ PV ）和存储后端之间建立链接，并获得其他存储指标。

步骤

1. 从左侧导航区域的信息板中、选择*后端*。

2. 执行以下操作之一：
   - 新建后端：选择*添加*以管理现有后端、选择* ONTAP 、然后选择*下一步。
   - 已发现后端：从操作菜单中、从受管集群中的已发现后端选择*管理*。
3. 输入ONTAP 集群管理IP地址和管理员凭据。凭据必须是集群范围的凭据。

> 您在此处输入凭据的用户必须具有 `ontapi` 在ONTAP 集群上的ONTAP 系统管理器中启用用户登录访问方法。如果您计划使用SnapMirror复制、请应用具有"admin"角色的用户凭据、该角色具有访问方法 `ontapi` 和 `http`、在源和目标ONTAP 集群上。请参见 "管理ONTAP 文档中的用户帐户" 有关详细信息 ...

4. 选择 * 下一步 * 。
5. 确认后端详细信息并选择 * 管理 * 。

结果

后端将显示在中 `Healthy` 包含摘要信息的列表中的状态。

> 您可能需要刷新页面才能显示后端。

## 添加存储分段

您可以使用Astra Control UI或添加存储分段 "API"。如果要备份应用程序和永久性存储，或者要跨集群克隆应用程序，则必须添加对象存储分段提供程序。Astra Control 会将这些备份或克隆存储在您定义的对象存储分段中。

如果您要将应用程序配置和永久性存储克隆到同一集群、则无需在Astra Control中使用存储分段。应用程序快照功能不需要存储分段。

您需要的内容

- 可从由Astra控制中心管理的集群访问的存储分段。
- 存储分段的凭据。
- 包含以下类型的存储分段：
  - NetApp ONTAP S3
  - NetApp StorageGRID S3
  - Microsoft Azure
  - 通用 S3

> Amazon Web Services (AWS)和Google Cloud Platform (GCP)使用通用S3存储分段类型。

> 虽然Astra控制中心支持将Amazon S3作为通用S3存储分段提供商、但Astra控制中心可能不支持声称支持Amazon S3的所有对象存储供应商。

步骤

1. 在左侧导航区域中，选择 * 桶 * 。

2. 选择 * 添加 * 。
3. 选择存储分段类型。

> 添加存储分段时，请选择正确的存储分段提供程序，并为该提供程序提供正确的凭据。例如， UI 接受 NetApp ONTAP S3 作为类型并接受 StorageGRID 凭据；但是，这将发生原因使使用此存储分段执行所有未来应用程序备份和还原失败。

4. 输入现有存储分段名称和可选的问题描述。

> 存储分段名称和问题描述 显示为备份位置、您可以稍后在创建备份时选择该位置。此名称也会在配置保护策略期间显示。

5. 输入 S3 端点的名称或 IP 地址。
6. 在*选择凭据*下、选择*添加*或*使用现有*选项卡。
   - 如果选择*添加*：
     i. 在 Astra Control 中输入凭据名称，以便与其他凭据区分开。
     ii. 通过粘贴剪贴板中的内容来输入访问 ID 和机密密钥。
   - 如果选择*使用现有*：
     i. 选择要用于存储分段的现有凭据。
7. 选择 … `Add`。

> 添加存储分段时、Astra Control会使用默认存储分段指示符标记一个存储分段。您创建的第一个存储分段将成为默认存储分段。添加分段时、您可以稍后决定添加 "设置另一个默认存储分段"。

## 下一步是什么?

现在、您已登录并将集群添加到Astra控制中心、即可开始使用Astra控制中心的应用程序数据管理功能。

- "管理本地用户和角色"
- "开始管理应用程序"
- "保护应用程序"
- "管理通知"
- "连接到 Cloud Insights"
- "添加自定义 TLS 证书"
- "更改默认存储类"

## 了解更多信息

- "使用 Astra Control API"
- "已知问题"

# 有关 Astra 控制中心的常见问题

如果您只是想快速了解问题解答，此常见问题解答会很有帮助。

## 概述

以下各节将为您在使用 Astra 控制中心时可能遇到的其他一些问题提供解答。如需更多说明，请联系 astra.feedback@netapp.com

## 访问 Astra 控制中心

- 什么是 Astra Control URL？ *

Astra 控制中心使用本地身份验证以及每个环境专用的 URL 。

对于URL、在浏览器中、在安装Astra控制中心时、输入在Astra_control_center.YAML自定义资源(CR)文件的spec.astraAddress字段中设置的完全限定域名(FQDN)。电子邮件是您在Astra_control_center.YAML CR的spec.email字段中设置的值。

## 许可

- 我正在使用评估版许可证。如何更改为完整许可证？ *

您可以通过获取 NetApp 许可证文件（ NLF ）轻松更改为完整许可证。

- 步骤 *
  1. 从左侧导航栏中，选择 * 帐户 * > * 许可证 * 。
  2. 选择 * 添加许可证 * 。
  3. 浏览到下载的许可证文件并选择 * 添加 * 。
- 我正在使用评估版许可证。我是否仍能管理应用程序？ *

可以，您可以使用评估版许可证测试管理应用程序功能。

## 注册 Kubernetes 集群

- 在添加到 Astra Control 后，我需要向 Kubernetes 集群添加工作节点。我该怎么办？ *

可以将新的工作节点添加到现有池中。这些信息将由 Astra Control 自动发现。如果新节点在 Astra Control 中不可见，请检查新工作节点是否正在运行受支持的映像类型。您还可以使用验证新工作节点的运行状况 `kubectl get nodes` 命令：

- 如何正确取消管理集群？ *
  1. "从 Astra Control 取消管理应用程序"。
  2. "从 Astra Control 取消管理集群"。
- 从 Astra Control 中删除 Kubernetes 集群后，应用程序和数据会发生什么情况？ *

从 Astra Control 中删除集群不会对集群的配置（应用程序和永久性存储）进行任何更改。对该集群上的应用程

序执行的任何 Astra Control 快照或备份都将无法还原。由 Astra Control 创建的永久性存储备份仍保留在 Astra Control 中，但无法还原。

> 在通过任何其他方法删除集群之前，请始终从 Astra Control 中删除集群。如果在集群仍由 Astra Control 管理时使用其他工具删除集群，则可能会对您的 Astra Control 帐户出现发生原因问题。

- 取消管理集群时是否自动从集群中卸载 NetApp Trident？ * 从 Astra 控制中心取消管理集群时，不会自动从集群中卸载 Trident 。要卸载 Trident ，您需要 "请按照 Trident 文档中的以下步骤进行操作"。

## 管理应用程序

- Astra Control 是否可以部署应用程序？ *

Astra Control 不会部署应用程序。应用程序必须部署在 Astra Control 之外。

- 停止从 Astra Control 管理应用程序后，应用程序会发生什么情况？ *

任何现有备份或快照都将被删除。应用程序和数据始终可用。数据管理操作不适用于非受管应用程序或属于该应用程序的任何备份或快照。

- Astra Control 是否可以管理非 NetApp 存储上的应用程序？ *

否虽然 Astra Control 可以发现使用非 NetApp 存储的应用程序，但它无法管理使用非 NetApp 存储的应用程序。

- 我是否应该管理 Astra Control 本身？ * 不，您不应该管理 Astra Control 本身，因为它是一个 " 系统应用程序 " 。
- 运行状况不正常的 Pod 是否影响应用程序管理？ * 如果受管应用程序中的 Pod 处于运行状况不正常的状态，则 Astra Control 无法创建新的备份和克隆。

## 数据管理操作

- 我的应用程序使用多个 PV 。Astra Control是否会为这些PV创建快照和备份？*

是的。Astra Control对应用程序执行的快照操作包括绑定到应用程序PVC的所有PV的快照。

- 是否可以直接通过其他接口或对象存储管理 Astra Control 创建的快照？ *

否Astra Control 创建的快照和备份只能使用 Astra Control 进行管理。