入门

Astra Control Center

NetApp
November 21, 2023

目录

入门

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

Astra 控制中心需要以下类型的操作环境之一：

- Kubernetes 1.20 到 1.23
- 使用 RKE1 的 Rancher 2.2.8 ， 2.0.9 或 2.6
- Red Hat OpenShift 容器平台 4.6-8 ， 4.7 ， 4.8 或 4.9
- VMware Tanzu Kubernetes 网格 1.4
- VMware Tanzu Kubernetes Grid Integrated Edition 1.12.2

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
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	• 已安装并配置 Astra Trident 21.04 或更高版本 • 如果使用Astra数据存储作为存储后端、则已安装并配置Astra Trident 21.10.1或更高版本

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 。
- * 天文学 Trident / ONTAP 配置 *： 天文学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 控制中心支持由 Astra Trident 提供的以下 ONTAP 驱动程序：
  - ontap-NAS
  - ontap-san
  - ontap-san-economy.

> 在 OpenShift 环境中克隆应用程序期间， Astra Control Center 需要允许 OpenShift 挂载卷并更改文件所有权。因此，您需要配置 ONTAP 卷导出策略以允许执行这些操作。您可以使用以下命令执行此操作：
>
> 1. `export-policy rule modify -vserver <storage virtual machine name> -policyname <policy name> -ruleindex 1 -superuser sys`
> 2. `export-policy rule modify -vserver <storage virtual machine name> -policyname <policy name> -ruleindex 1 -anon 65534`

> 如果您计划将第二个 OpenShift 操作环境添加为托管计算资源，则需要确保已启用 Astra Trident 卷快照功能。要使用 Astra Trident 启用和测试卷快照， "请参见官方的 Astra Trident 说明"。

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。为此，您可以编辑命名空间集群上的 `TanuKubernetes Cluster` 资源。
- 您必须创建一个允许 Astra 控制中心在集群中创建 Pod 的安全策略。您可以使用以下命令执行此操作：

```
kubectl config use-context <context-of-workload-cluster>
kubectl create clusterrolebinding default-tkg-admin-privileged-binding
--clusterrole=psp:vmware-system-privileged --group=system:authenticated
```

- 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。"

支持的存储后端

Astra 控制中心支持以下存储后端。

- Astra 数据存储
- NetApp ONTAP 9.5 或更高版本的 AFF 和 FAS 系统
- NetApp Cloud Volumes ONTAP

应用程序集群要求

对于计划从 Astra 控制中心管理的集群， Astra 控制中心具有以下要求。如果您计划管理的集群是托管 Astra 控制中心的运行环境集群，则这些要求也适用。

- Kubernetes 的最新版本 "Snapshot 控制器组件" 已安装
- Astra Trident "volumesnapshotclass 对象" 已由管理员定义
- 集群上存在默认 Kubernetes 存储类
- 至少将一个存储类配置为使用 Astra Trident

> 您的应用程序集群应具有一个 `kubeconfig.yaml` 文件，该文件仅定义一个 *context* 元素。请访问的 Kubernetes 文档 "有关创建 kubeconfig 文件的信息"。

> 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。

应用程序管理要求

Astra Control 具有以下应用程序管理要求：

- * 许可 *： 要使用 Astra 控制中心管理应用程序，您需要获得 Astra 控制中心许可证。
- * 命名空间 *： Astra Control 要求一个应用程序不能跨越多个命名空间，但一个命名空间可以包含多个应用程序。
- * 存储类 *： 如果您安装的应用程序明确设置了 StorageClass ，并且需要克隆该应用程序，则克隆操作的目标集群必须具有最初指定的 StorageClass 。将显式设置了 StorageClass 的应用程序克隆到不具有相同 StorageClass 的集群将失败。
- * Kubernetes Resources*： 使用非 Astra Control 收集的 Kubernetes 资源的应用程序可能没有完整的应用程序数据管理功能。Astra Control 收集以下 Kubernetes 资源：

ClusterRole	ClusterRoleBinding.	配置映射
cronjob	自定义资源定义	自定义资源
DemonSet	DeploymentConfig	HorizontalPodAutoscaler
传入	MutatingWebhook	网络策略
PersistentVolumeClaim	POD	PodDisruptionBuget
播客模板	ReplicaSet	Role
RoleBinding.	路由	机密
服务	ServiceAccount	状态集
验证 Webhook		

支持的应用程序安装方法

Astra Control 支持以下应用程序安装方法：

- * 清单文件 *： Astra Control 支持使用 kubectl 从清单文件安装的应用程序。例如：

```
kubectl apply -f myapp.yaml
```

- * Helm 3*：如果使用 Helm 安装应用程序，则 Astra Control 需要 Helm 版本 3 。完全支持管理和克隆随 Helm 3 安装的应用程序（或从 Helm 2 升级到 Helm 3 ）。不支持管理随 Helm 2 安装的应用程序。
- * 操作员部署的应用程序 *： Astra Control 支持使用命名空间范围的运算符安装的应用程序。以下是已针对此安装模式验证的一些应用程序：
  - "Apache K8ssandra"
  - "Jenkins CI"
  - "Percona XtraDB 集群"

> 操作员及其安装的应用程序必须使用相同的命名空间；您可能需要为操作员修改部署 .yaml 文件，以确保情况确实如此。

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。如果没有许可证，您将无法：

- 定义自定义应用程序
- 为现有应用程序创建快照或克隆
- 配置数据保护策略

如果您要尝试使用 Astra 控制中心，可以 "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"。

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。

> 如果要在 Tanzu Kubernetes 网格集群上托管 Astra 控制中心，请使用 `kubectl get nssxlbmonitors -a` 命令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 。

下一步行动

查看 "快速入门" 概述。

Astra 控制中心快速入门

此页面简要概述了开始使用 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。

试用！如果您要试用 Astra Control Center ，可以使用 90 天评估许可证。请参见 "许可信息" 了解详细信息。

1. 查看 **Kubernetes** 集群要求

- Astra 可与具有 Trident 配置的 ONTAP 存储后端或 Astra 数据存储存储后端的 Kubernetes 集群结合使用。
- 集群必须以运行状况良好的状态运行，并且至少有三个联机辅助节点。

- 集群必须运行 Kubernetes 。

"了解有关 Astra 控制中心要求的更多信息"。

2 下载并安装**Astra**控制中心

- 从下载 Astra 控制中心 "NetApp 支持站点 Astra 控制中心下载页面"。
- 在本地环境中安装 Astra Control Center 。

  或者，也可以使用 Red Hat OperatorHub 安装 Astra 控制中心。

"了解有关安装 Astra 控制中心的更多信息"。

3 完成一些初始设置任务

- 添加许可证
- 添加 Kubernetes 集群， Astra 控制中心将发现详细信息。
- 添加 ONTAP 或 "Astra 数据存储" 存储后端。
- 或者，添加用于存储应用程序备份的对象存储分段。

"了解有关初始设置过程的更多信息"。

4 使用 **Astra** 控制中心

设置完 Astra 控制中心后，您接下来可能会执行以下操作：

- 管理应用程序。 "详细了解如何管理应用程序"。
- 或者，也可以连接到 NetApp Cloud Insights ，以便在 Astra 控制中心 UI 中显示有关系统运行状况，容量和吞吐量的指标。 "了解有关连接到 Cloud Insights 的更多信息"。

5 从此快速入门继续

"安装 Astra 控制中心"。

了解更多信息

- "使用 Astra Control API"

安装概述

选择并完成以下 Astra 控制中心安装过程之一：

- "使用标准流程安装 Astra 控制中心"
- "（如果使用 Red Hat OpenShift ）使用 OpenShift OperatorHub 安装 Astra 控制中心"

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

使用标准流程安装 Astra 控制中心

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

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

您需要的内容

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

  OpenShift 示例：

  ```
  oc get clusteroperators
  ```

- 确保所有 API 服务均处于运行状况良好且可用：

  OpenShift 示例：

  ```
  oc get apiservices
  ```

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

关于此任务

Astra 控制中心安装过程将执行以下操作：

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

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

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

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

步骤

要安装 Astra 控制中心，请执行以下步骤：

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

下载并解包**Astra Control Center**软件包

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

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

4. 提取映像：

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

安装**NetApp Astra kubectl**插件

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

您需要的内容

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

步骤

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

```
ls kubectl-astra/
```

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

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

将映像添加到本地注册表

1. 更改为Astra目录：

```
cd acc
```

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

> 有关自动加载映像的信息，请参见下面的示例脚本。

a. 登录到注册表：

Docker：

```
docker login [your_registry_path]
```

播客：

```
podman login [your_registry_path]
```

b. 使用适当的脚本加载映像，标记映像，并将这些映像推送到本地注册表：

Docker：

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

播客：

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

为具有身份验证要求的注册表设置命名空间和密钥

1. 如果您使用的注册表需要身份验证，则需要执行以下操作：

   a. 创建 `NetApp-Acc-operator` 命名空间：

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

   响应：

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

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

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

响应示例：

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

c. 创建 `NetApp-Accc` （或自定义命名）命名空间。

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

响应示例：

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

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

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

响应

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

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

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

安装 **Astra** 控制中心操作员

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

```
vim astra_control_center_operator_deploy.yaml
```

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

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

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

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

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

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

2. 安装 Astra 控制中心操作员：

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

响应示例：

```
namespace/netapp-acc-operator created
customresourcedefinition.apiextensions.k8s.io/astracontrolcenters.astra.
netapp.io created
role.rbac.authorization.k8s.io/acc-operator-leader-election-role created
clusterrole.rbac.authorization.k8s.io/acc-operator-manager-role created
clusterrole.rbac.authorization.k8s.io/acc-operator-metrics-reader
created
clusterrole.rbac.authorization.k8s.io/acc-operator-proxy-role created
rolebinding.rbac.authorization.k8s.io/acc-operator-leader-election-
rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/acc-operator-manager-
rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/acc-operator-proxy-
rolebinding created
configmap/acc-operator-manager-config created
service/acc-operator-controller-manager-metrics-service created
deployment.apps/acc-operator-controller-manager created
```

配置 **Astra** 控制中心

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

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

```
vim astra_control_center_min.yaml
```

> 首次部署 Astra 控制中心后，无法更改 CR 配置的属性。

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

a. 将 `[yor_registry_path]` 更改为上一步中用于推送映像的注册表路径。
b. 将 `accountName` 字符串更改为要与帐户关联的名称。
c. 将 `astraAddress` 字符串更改为要在浏览器中使用的 FQDN 以访问 Astra。请勿在此地址中使用 `http：//` 或 `https：//`。复制此 FQDN 以在中使用 后续步骤。
d. 将 `email` 字符串更改为默认的初始管理员地址。复制此电子邮件地址以在中使用 后续步骤。
e. 将 AutoSupport 的 **已注册** 更改为 `false` 对于无 Internet 连接的站点，或者将已连接站点的 `true` 保留。

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

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

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

i. 完成以下步骤之一：

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

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

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

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

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

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

```
apiVersion: astra.netapp.io/v1
kind: AstraControlCenter
metadata:
  name: astra
spec:
  accountName: "Example"
  astraVersion: "ASTRA_VERSION"
  astraAddress: "astra.example.com"
  astraKubeConfigSecret: "acc-kubeconfig-cred or custom secret name"
  ingressType: "Generic"
  autoSupport:
    enrolled: true
  email: "[admin@example.com]"
  firstName: "SRE"
  lastName: "Admin"
  imageRegistry:
    name: "[your_registry_path]"
    secret: "astra-registry-cred"
  storageClass: "ontap-gold"
```

完成 **Astra** 控制中心和操作员安装

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

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

响应示例：

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

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

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

响应示例：

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

验证系统状态

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

1. 验证是否已成功安装所有系统组件。

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

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

响应示例：

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

```
api-token-authentication-sqnb7             1/1     Running   0
8m50s
asup-5578bbdd57-dxkbp                      1/1     Running   0
9m3s
authentication-56bff4f95d-mspmq            1/1     Running   0
7m31s
bucketservice-6f7968b95d-9rrrl             1/1     Running   0
8m36s
cert-manager-5f6cf4bc4b-82khn              1/1     Running   0
6m19s
cert-manager-cainjector-76cf976458-sdrbc   1/1     Running   0
6m19s
cert-manager-webhook-5b7896bfd8-2n45j      1/1     Running   0
6m19s
cloud-extension-749d9f684c-8bdhq           1/1     Running   0
9m6s
cloud-insights-service-7d58687d9-h5tzw     1/1     Running   2
8m56s
composite-compute-968c79cb5-nv7l4          1/1     Running   0
9m11s
composite-volume-7687569985-jg9gg          1/1     Running   0
8m33s
credentials-5c9b75f4d6-nx9cz               1/1     Running   0
8m42s
entitlement-6c96fd8b78-zt7f8               1/1     Running   0
8m28s
features-5f7bfc9f68-gsjnl                  1/1     Running   0
8m57s
fluent-bit-ds-h88p7                        1/1     Running   0
7m22s
fluent-bit-ds-krhnj                        1/1     Running   0
7m23s
fluent-bit-ds-l5bjj                        1/1     Running   0
7m22s
fluent-bit-ds-lrclb                        1/1     Running   0
7m23s
fluent-bit-ds-s5t4n                        1/1     Running   0
7m23s
fluent-bit-ds-zpr6v                        1/1     Running   0
7m22s
graphql-server-5f5976f4bd-vbb4z            1/1     Running   0
7m13s
identity-56f78b8f9f-8h9p9                  1/1     Running   0
8m29s
influxdb2-0                                1/1     Running   0
11m
```

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

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

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

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

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

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

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

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

YAML 详细信息示例

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

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

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

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

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

设置传入以进行负载平衡

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

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

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

根据您使用的入口控制器类型，步骤会有所不同：

- nginx 入口控制器
- OpenShift 入口控制器

您需要的内容

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

nginx 入口控制器的步骤

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

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

b. 对于 `v1` 新架构，请遵循以下示例：

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

OpenShift 入口控制器的步骤

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

```
oc create route edge --service=traefik
--port=web -n [netapp-acc or custom namespace]
--insecure-policy=Redirect --hostname=<ACC address>
--cert=cert.pem --key=key.pem
```

登录到 **Astra** 控制中心 **UI**

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

步骤

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

> 您可以在登录后创建自定义证书。

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

> 如果您输入的密码三次不正确，管理员帐户将锁定 15 分钟。

4. 选择 * 登录 * 。
5. 根据提示更改密码。

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

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

对安装进行故障排除

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

步骤

1. 要检查 Astra 控制中心操作员日志，请输入以下内容：

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

下一步行动

执行以完成部署 "设置任务"。

使用 OpenShift OperatorHub 安装 Astra 控制中心

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

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

您需要的内容

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

```
oc get clusteroperators
```

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

```
oc get apiservices
```

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

步骤

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

下载并解包**Astra Control Center**软件包

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

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

4. 提取映像：

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

安装**NetApp Astra kubectl**插件

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

您需要的内容

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

步骤

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

```
ls kubectl-astra/
```

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

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

将映像添加到本地注册表

1. 更改为Astra目录：

```
cd acc
```

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

> 有关自动加载映像的信息，请参见下面的示例脚本。

a. 登录到注册表：

Docker：

```
docker login [your_registry_path]
```

播客：

```
podman login [your_registry_path]
```

b. 使用适当的脚本加载映像，标记映像，并将这些映像推送到本地注册表：

Docker：

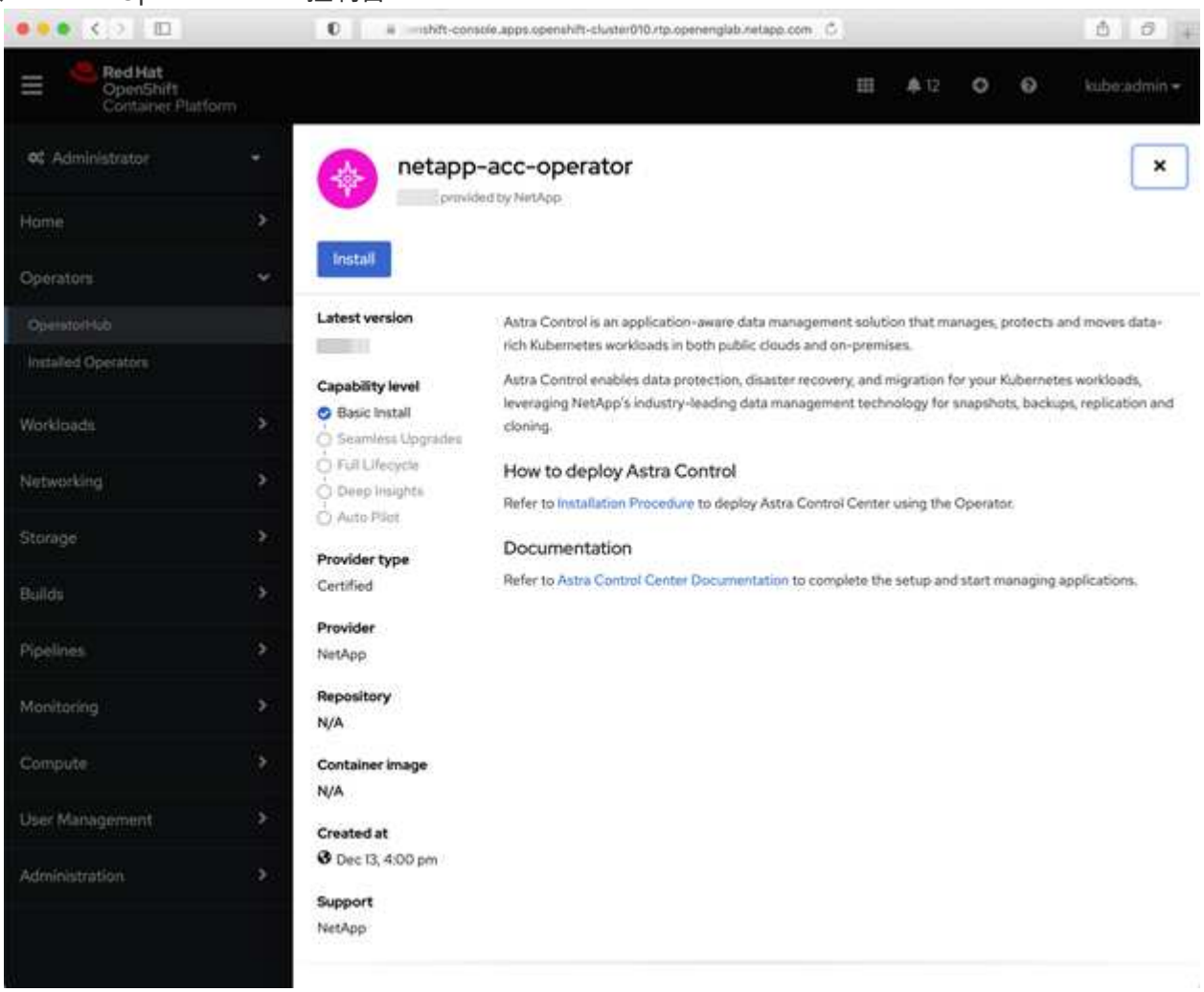
```
export REGISTRY=[Docker_registry_path]
for astraImageFile in $(ls images/*.tar) ; do
  # Load to local cache. And store the name of the loaded image
trimming the 'Loaded images: '
  astraImage=$(docker load --input ${astraImageFile} | sed 's/Loaded
image: //')
  astraImage=$(echo ${astraImage} |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

播客：

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

找到操作员安装页面

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



：

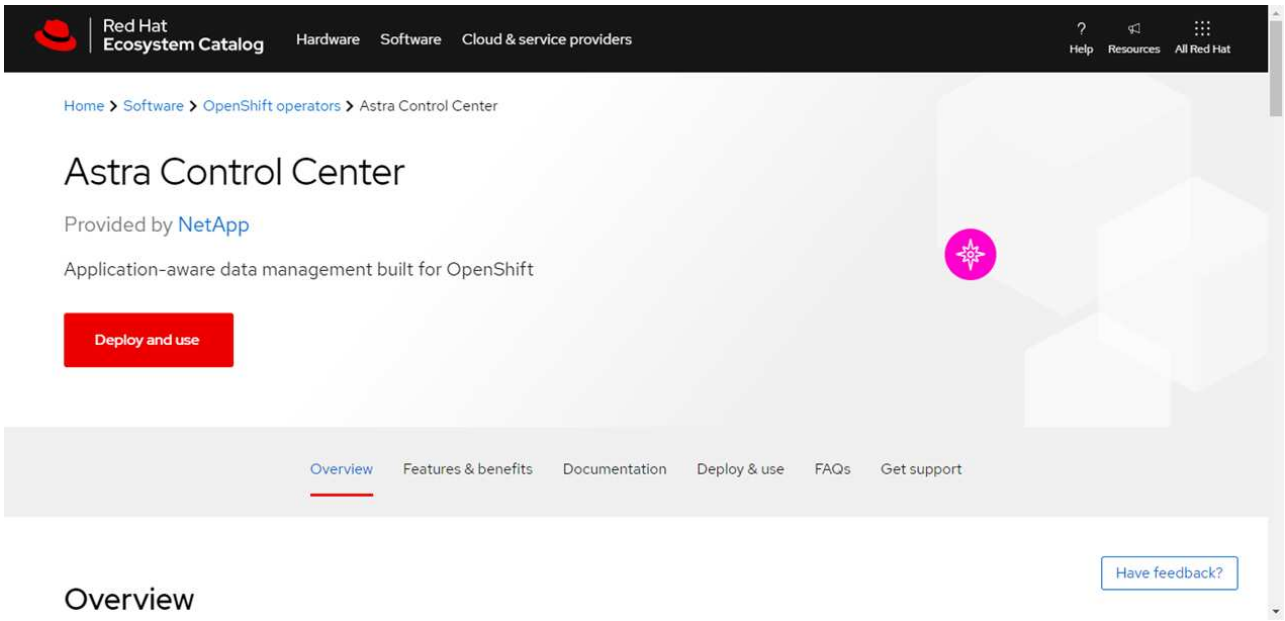
i. 登录到 OpenShift 容器平台 UI 。

ii. 从侧面菜单中，选择 * 运算符 > OperatorHub * 。

iii. 选择 NetApp Astra Control Center 操作员。

iv. 选择 * 安装 * 。

- 从 Red Hat 生态系统目录
：



i. 选择 NetApp Astra 控制中心 "运算符"。

ii. 选择 * 部署并使用 * 。

安装操作员

1. 完成 * 安装操作员 * 页面并安装操作员：

   操作员将在所有集群命名空间中可用。

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

   b. 选择手动或自动批准策略。

      建议手动批准。每个集群只能运行一个操作员实例。

   c. 选择 * 安装 * 。

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

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

安装 **Astra** 控制中心

1. 在 Astra 控制中心操作员的详细信息视图的控制台中，在提供的 API 部分中选择 `Create instance` 。
2. 填写 `Create AstraControlCenter` Form 字段：
   a. 保留或调整 Astra 控制中心名称。
   b. （可选）启用或禁用自动支持。建议保留自动支持功能。
   c. 输入 Astra 控制中心地址。请勿在此地址中输入 `http ： //` 或 `https ： //` 。
   d. 输入 Astra 控制中心版本；例如 21.12.60 。
   e. 输入帐户名称，电子邮件地址和管理员姓氏。
   f. 保留默认卷回收策略。
   g. 在 * 映像注册表 * 中，输入本地容器映像注册表路径。请勿在此地址中输入 `http ： //` 或 `https ： //` 。
   h. 如果您使用的注册表需要身份验证，请输入密钥。
   i. 输入管理员的名字。
   j. 配置资源扩展。
   k. 保留默认存储类。
   l. 定义 CRD 处理首选项。
3. 选择 `Create` 。

下一步行动

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

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

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

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

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

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

在 **Amazon Web Services** 中部署 **Astra** 控制中心

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

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

AWS所需的功能

在 AWS 中部署 Astra 控制中心之前，您需要满足以下条件：

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

AWS 的操作环境要求

Astra 控制中心需要以下 AWS 操作环境：

- Red Hat OpenShift 容器平台 4.8

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

除了环境的资源要求之外， Astra 控制中心还需要以下资源：

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

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

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

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

AWS 部署概述

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

下面详细介绍了其中每个步骤。

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

确保您具有足够的 IAM 权限

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

请参见 "初始 AWS 凭据"。

在 AWS 上安装 RedHat OpenShift 集群

在 AWS 上安装 RedHat OpenShift 容器平台集群。

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

配置 **AWS**

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

按照 AWS 文档完成以下步骤。请参见 "AWS 安装文档"。

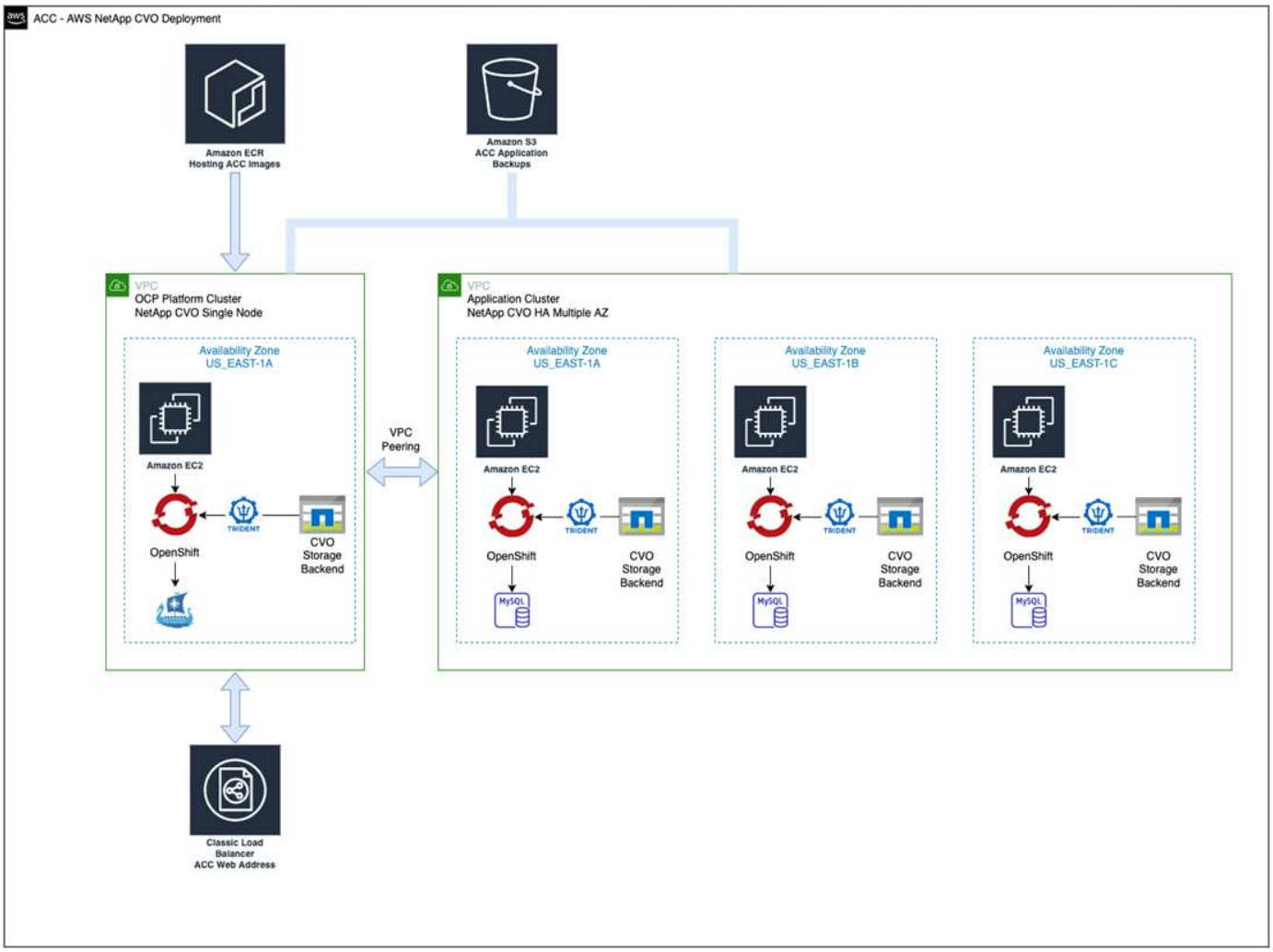
1. 创建AWS虚拟网络。
2. 查看 EC2 计算实例。这可以是 AWS 中的裸机服务器或 VM 。
3. 如果实例类型尚未与主节点和工作节点的 Astra 最低资源要求匹配，请更改 AWS 中的实例类型以满足 Astra 要求。 请参见 "Astra 控制中心要求"。
4. 至少创建一个 AWS S3 存储分段来存储备份。
5. 创建 AWS 弹性容器注册表（ ECR ）以托管所有 AccR 映像。

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

6. 将这些 Accc 映像推送到您定义的注册表。

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

以下是 AWS 部署示例：



配置 **NetApp Cloud Manager**

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

按照 Cloud Manager 文档完成以下步骤。请参见以下内容：

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

步骤

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

b. 在右上角，记下 Trident 版本。

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

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

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

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

安装 **Astra** 控制中心

请遵循标准 "Astra 控制中心安装说明"。

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

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

Azure所需的功能

在 Azure 中部署 Astra 控制中心之前，您需要满足以下条件：

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

Azure 的操作环境要求

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

除了环境的资源要求之外， Astra 控制中心还需要以下资源：

请参见 "Astra 控制中心运营环境要求"。

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

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

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

Azure 部署概述

下面简要介绍了适用于 Azure 的 Astra 控制中心的安装过程。

下面详细介绍了其中每个步骤。

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

在 Azure 上安装 RedHat OpenShift 集群

第一步是在 Azure 上安装 RedHat OpenShift 集群。

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

创建 **Azure** 资源组

至少创建一个 Azure 资源组。

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

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

确保您具有足够的 **IAM** 权限

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

请参见 "Azure 凭据和权限"。

配置 **Azure**

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

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

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

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

   - 示例 * ：

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

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

10. 设置 DNS 区域。

配置 NetApp Cloud Manager

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

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

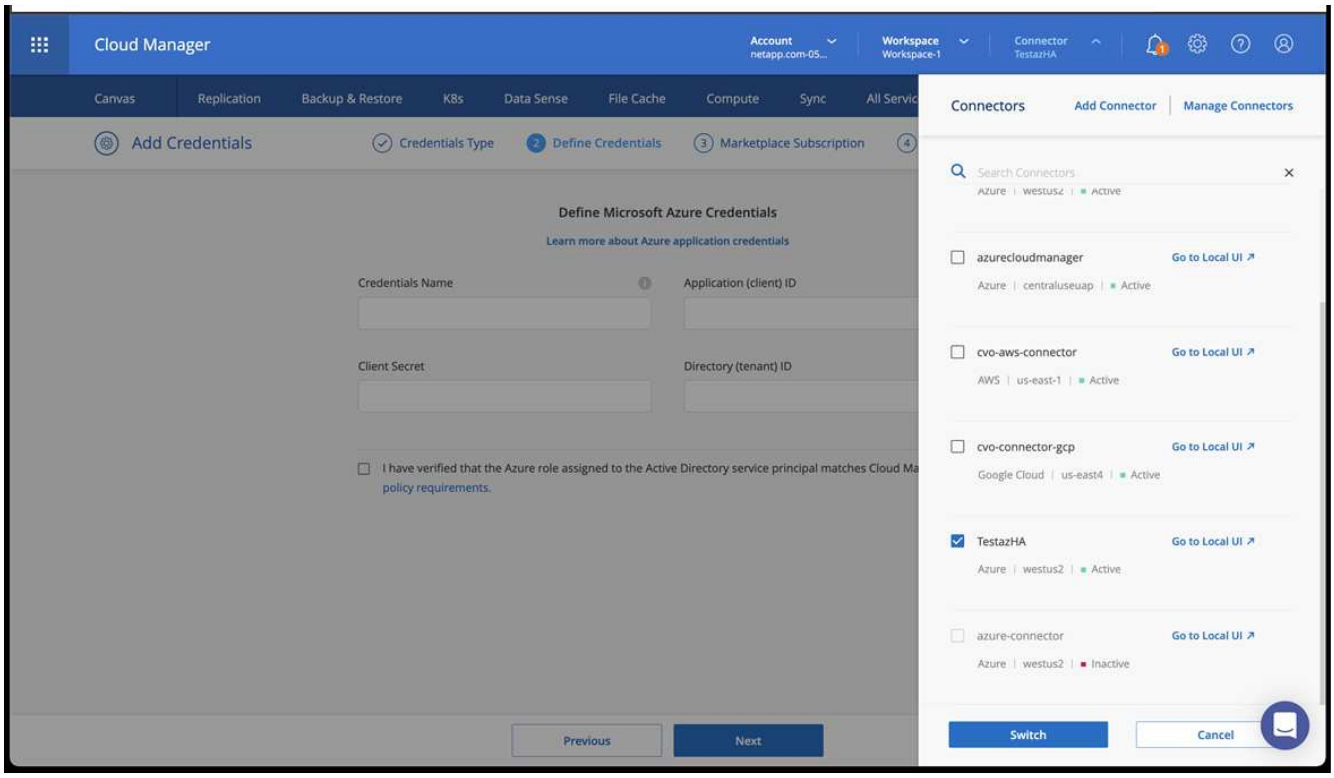
您需要的内容

使用所需的 IAM 权限和角色访问 Azure 帐户

步骤

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

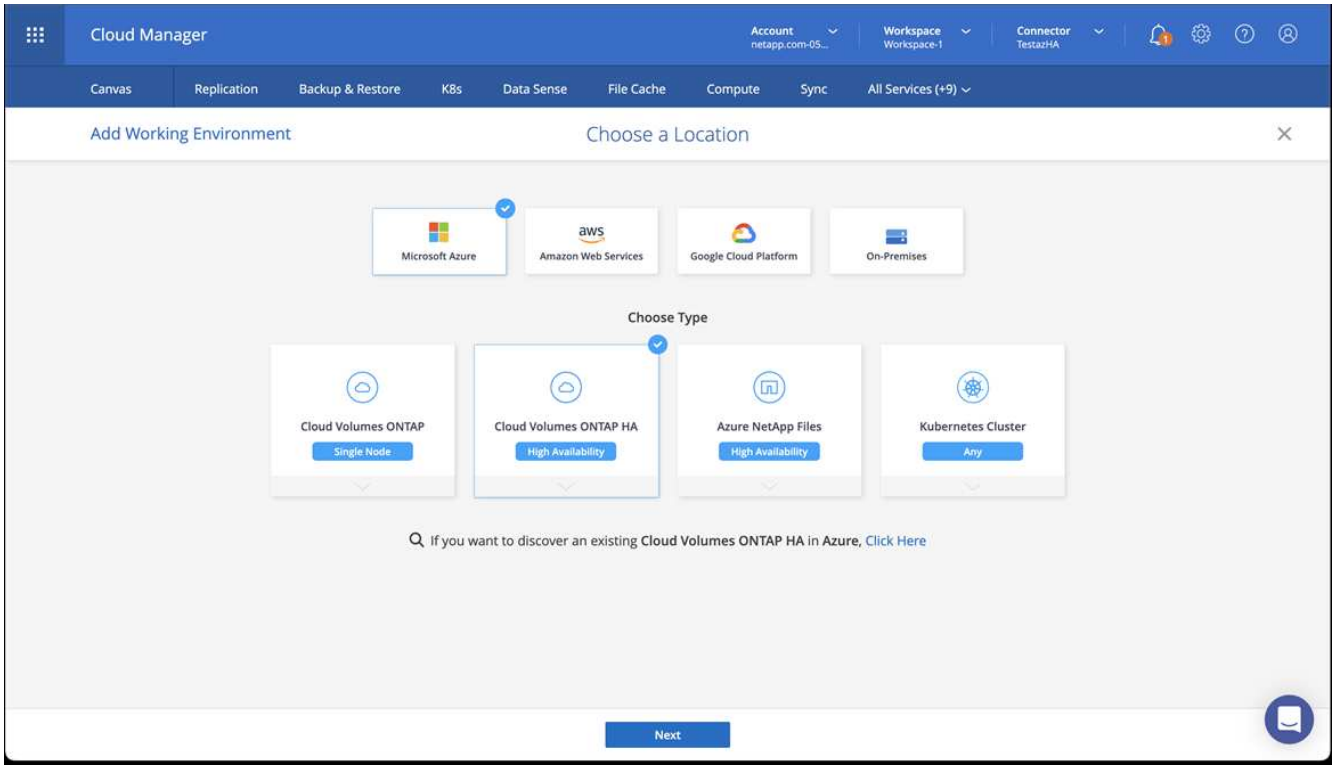
      请参见 "从 Cloud Manager 在 Azure 中创建连接器"。
3. 确保连接器正在运行，然后切换到该连接器。



4. 为您的云环境创建一个工作环境。

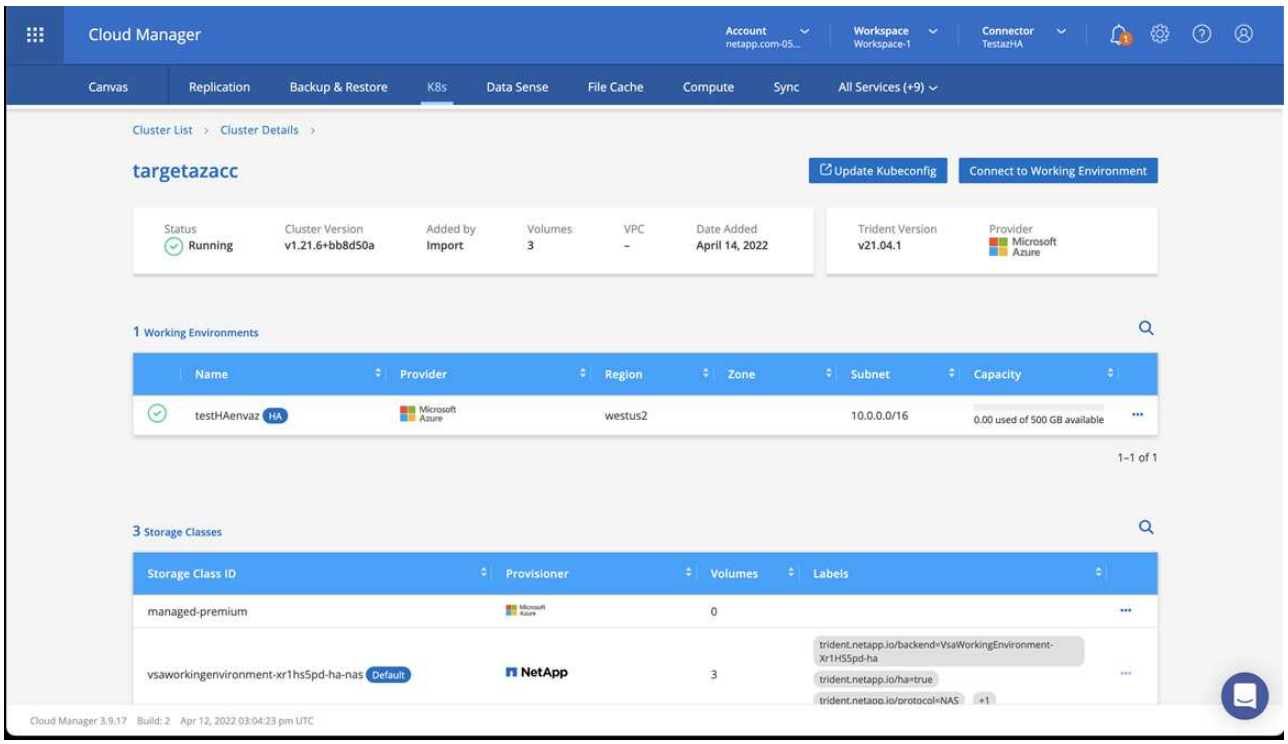
   a. 位置： "Microsoft Azure" 。

   b. 键入： Cloud Volumes ONTAP HA 。



5. 导入 OpenShift 集群。集群将连接到您刚刚创建的工作环境。

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



b. 在右上角，记下 Trident 版本。

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

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

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

安装和配置 **Astra** 控制中心

按照标准安装 Astra 控制中心 "安装说明"。

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

设置 Astra 控制中心

Astra 控制中心支持并监控 ONTAP 和 Astra 数据存储作为存储后端。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
- [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 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。

有关如何计算许可证的详细信息，请参见 "许可"。

> 要更新现有评估版或完整许可证，请参见 "更新现有许可证"。

Astra 控制中心许可证使用 Kubernetes CPU 单元测量 CPU 资源。此许可证需要考虑分配给所有受管 Kubernetes 集群的工作节点的 CPU 资源。在添加许可证之前，您需要从获取许可证文件（NLF） "NetApp 支持站点"。

您还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"。

> 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。

您需要的内容

从下载 Astra 控制中心时 "NetApp 支持站点"，您还下载了 NetApp 许可证文件（NLF）。确保您有权访问此许可证文件。

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。

> 如果您拥有评估许可证，请务必存储帐户 ID ，以避免在未发送 ASUP 的情况下 Astra 控制中心出现故障时丢失数据。

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。对于 Astra 数据存储，您希望添加 Kubernetes 应用程序集群，其中包含使用由 Astra 数据存储配置的卷的应用程序。

> 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。您可以使用 * 添加集群 * 功能通过 Astra 控制中心管理集群。

> 当Astra Control管理集群时、它会跟踪集群的默认存储类。如果使用`kubectl`命令更改存储类、则Astra Control将还原此更改。要更改由Astra Control管理的集群中的默认存储类、请使用以下方法之一：
>
> - 使用Astra Control API `PUT /managedClusters` Endpoint、并使用`DefaultStorageClass`参数分配其他默认存储类。
> - 使用Astra Control Web UI分配其他默认存储类。请参见 [更改默认存储类]。

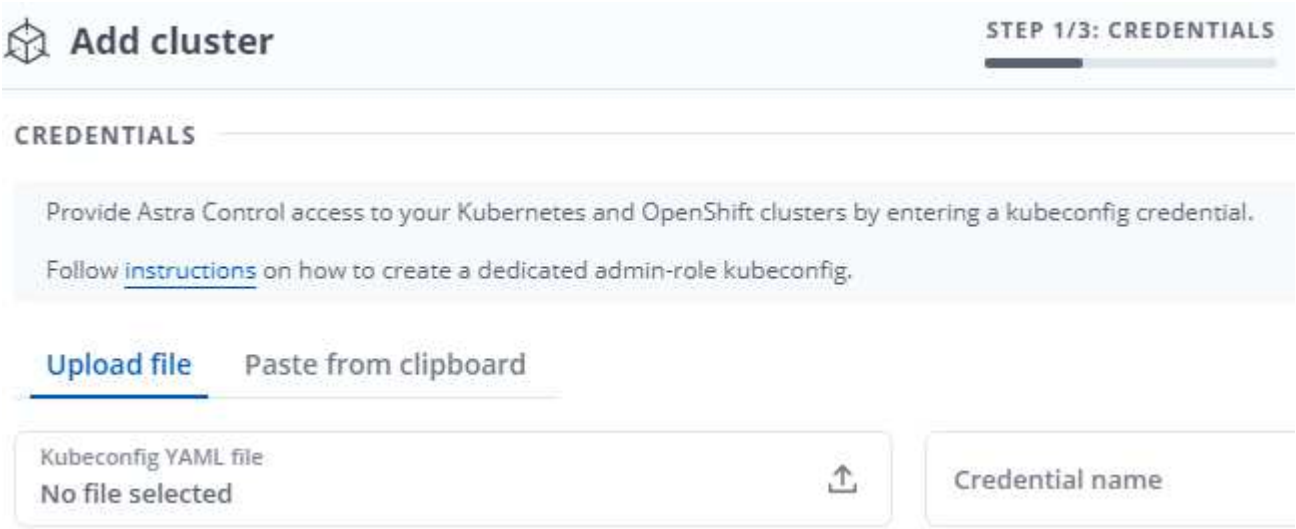
您需要的内容

- 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 Astra 控制中心用户界面的 * 信息板 * 中，选择集群部分中的 * 添加 * 。
2. 在打开的 * 添加集群 * 窗口中，上传 `kubeconfig.yaml` 文件或粘贴 `kubeconfig.yaml` 文件的内容。

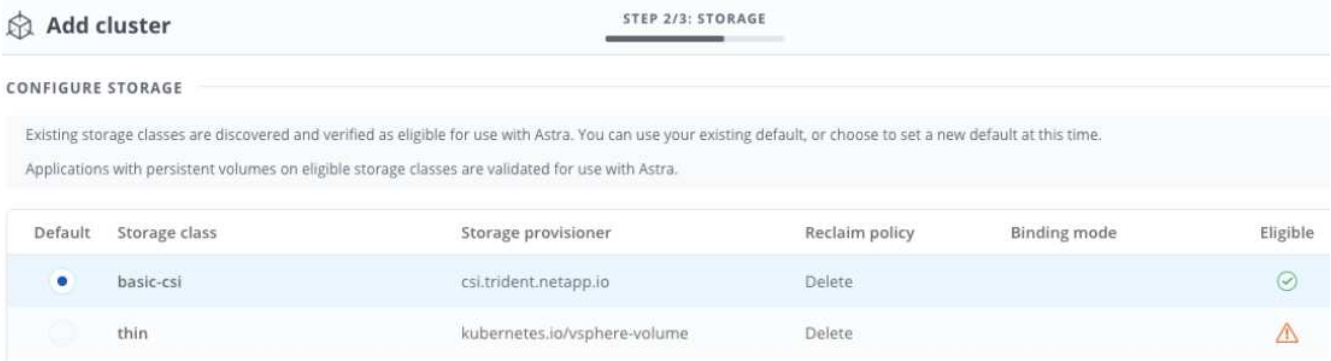
   > `kubeconfig.yaml` 文件应仅包含一个集群的集群凭据 * 。



   > 如果您创建自己的 `kubeconfig` 文件，则应仅在其中定义 * 一 * 上下文元素。请参见 "Kubernetes 文档" 有关创建 `kubeconfig` 文件的信息。

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 * 配置存储 * 。
5. 选择要用于此 Kubernetes 集群的存储类，然后选择 * 审核 * 。

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



6. 查看相关信息，如果一切正常，请选择 * 添加集群 * 。

结果

集群将进入 * 正在发现 * 状态，然后更改为 * 正在运行 * 。您已成功添加 Kubernetes 集群，现在正在 Astra 控制中心中对其进行管理。

> 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，然后运行 `oc get Pod -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

您可以添加存储后端，以使 Astra Control 能够管理其资源。您可以在受管集群上部署存储后端、也可以使用现有存储后端。

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。

现有**Astra Data Store**部署所需的资源

- 您已添加Kubernetes应用程序集群和底层计算集群。

> 添加适用于Astra Data Store的Kubernetes应用程序集群并由Astra Control管理后、该集群在已发现的后端列表中显示为`非受管`。接下来，您必须添加包含 Astra 数据存储的计算集群并将 Kubernetes 应用程序集群置于底层。您可以从用户界面中的 * 后端 * 执行此操作。选择集群的 "Actions" 菜单，选择 `Manage` ，然后 "添加集群"。在集群状态 非受管 更改为 Kubernetes 集群的名称后，您可以继续添加后端。

新的**Astra Data Store**部署所需的资源

- 您已拥有 "已上传要部署的安装包版本" 到Astra Control可访问的位置。
- 您已添加要用于部署的Kubernetes集群。
- 您已上传 Astra Data Store许可证 部署到可供Astra Control访问的位置。

选项

- [部署存储资源]
- [使用现有存储后端]

部署存储资源

您可以部署新的Astra数据存储并管理关联的存储后端。

步骤

1. 从信息板或后端菜单导航：
   - 从*信息板*：从资源摘要中、从存储后端窗格中选择一个链接、然后从后端部分中选择*添加*。
   - 从 * 后端 * ：
     i. 在左侧导航区域中，选择 * 后端 * 。
     ii. 选择 * 添加 * 。
2. 在*部署*选项卡中选择* Astra Data Store*部署选项。
3. 选择要部署的Astra Data Store软件包：
   a. 输入Astra Data Store应用程序的名称。
   b. 选择要部署的Astra数据存储的版本。

   > 如果您尚未上传要部署的版本、可以使用*添加软件包*选项或退出向导并使用 "软件包管理" 上传安装包。

4. 选择先前上传的Astra Data Store许可证、或者使用*添加许可证*选项上传要用于应用程序的许可证。

   > 具有完全权限的Astra Data Store许可证将与您的Kubernetes集群关联、并且这些关联的集群应自动显示。如果没有受管集群、您可以选择*添加集群*选项将其添加到Astra Control管理中。对于Astra Data Store许可证、如果许可证和集群之间未建立关联、您可以在向导的下一页定义此关联。

5. 如果尚未将Kubernetes集群添加到Astra Control管理中、则需要从* Kubernetes cluster*页面中执行此操作。从列表中选择一个现有集群或选择*添加底层集群*将集群添加到Astra Control管理中。
6. 选择要为Astra数据存储提供资源的Kubernetes集群的部署模板大小。

   > 选择模板时、请为大型工作负载选择具有更多内存和核心的大型节点、为小型工作负载选择更多节点。您应根据许可证允许的内容选择模板。每个模板选项都会建议符合条件且满足每个节点的内存、核心和容量模板模式的节点数。

7. 配置节点：
   a. 添加节点标签以标识支持此Astra数据存储集群的工作节点池。

   > 在开始部署或部署失败之前、必须将此标签添加到集群中要用于部署Astra Data Store的每个节点上。

   b. 手动配置每个节点的容量(GiB)或选择允许的最大节点容量。
   c. 配置集群中允许的最大节点数或允许集群中的最大节点数。
8. (仅限Astra Data Store完整许可证)输入要用于保护域的标签的密钥。

> 为每个节点的密钥至少创建三个唯一标签。例如、如果您的密钥为`astra.datastore.protection.domain`、则可以创建以下标签：`astra.datastore.protection.domain=domain1,astra.datastore.protection.domain=domain2`和`astra.datastore.protection.domain=domain3`。

9. 配置管理网络：
   a. 输入Astra Data Store内部管理的管理IP地址、该地址与工作节点IP地址位于同一子网上。
   b. 选择对管理网络和数据网络使用相同的NIC、或者单独进行配置。
   c. 输入用于存储访问的数据网络IP地址池、子网掩码和网关。
10. 查看配置并选择*部署*以开始安装。

结果

成功安装后、后端会在后端列表中显示为`Available`状态、并显示活动性能信息。

> 您可能需要刷新页面才能显示后端。

使用现有存储后端

您可以将已发现的ONTAP 或Astra数据存储存储后端引入Astra控制中心管理。

步骤

1. 从信息板或后端菜单导航：
   - 从*信息板*：从资源摘要中、从存储后端窗格中选择一个链接、然后从后端部分中选择*添加*。
   - 从 * 后端 * ：
     i. 在左侧导航区域中，选择 * 后端 * 。
     ii. 在受管集群中发现的后端上选择*管理*、或者选择*添加*来管理其他现有后端。
2. 选择 * 使用现有 * 选项卡。
3. 根据后端类型执行以下操作之一：
   - * Astra 数据存储库 * ：
     i. 选择* Astra Data Store*。
     ii. 选择受管计算集群并选择 * 下一步 * 。
     iii. 确认后端详细信息并选择*添加存储后端*。
   - * ONTAP * ：
     i. 选择* ONTAP *。
     ii. 输入 ONTAP 管理员凭据并选择 * 审核 * 。
     iii. 确认后端详细信息并选择*添加存储后端*。

结果

后端会在列表中显示为 `available` 状态，并显示摘要信息。

您可能需要刷新页面才能显示后端。

添加存储分段

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。

添加存储分段时， Astra Control 会将一个存储分段标记为默认存储分段指示符。您创建的第一个存储分段将成为默认存储分段。

如果要将应用程序配置和永久性存储克隆到同一集群，则不需要存储分段。

使用以下任一存储分段类型：

- NetApp ONTAP S3
- NetApp StorageGRID S3
- 通用 S3

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。

有关如何使用 Astra Control API 添加存储分段的说明，请参见 "Astra Automation 和 API 信息"。

步骤

1. 在左侧导航区域中，选择 * 桶 * 。
   a. 选择 * 添加 * 。
   b. 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。

   c. 创建新的存储分段名称或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。

   d. 输入 S3 端点的名称或 IP 地址。
   e. 如果您希望此存储分段成为所有备份的默认存储分段，请选中 `MMake this bucket the default bucket for this private cloud` 选项。

      创建的第一个存储分段不会显示此选项。

   f. 通过添加继续 凭据信息。

添加 **S3** 访问凭据

随时添加 S3 访问凭据。

步骤

1. 从 " 分段 " 对话框中，选择 * 添加 * 或 * 使用现有 * 选项卡。
    a. 在 Astra Control 中输入凭据名称，以便与其他凭据区分开。
    b. 通过粘贴剪贴板中的内容来输入访问 ID 和机密密钥。

更改默认存储类

您可以更改集群的默认存储类。

步骤

1. 在Astra控制中心Web UI中、选择*集群*。
2. 在*集群*页面上、选择要更改的集群。
3. 选择 * 存储 * 选项卡。
4. 选择*存储类*类别。
5. 选择要设置为默认值的存储类的*操作*菜单。
6. 选择*设置为默认值*。

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。

- "管理用户"
- "开始管理应用程序"
- "保护应用程序"
- "克隆应用程序"
- "管理通知"
- "连接到 Cloud Insights"
- "添加自定义 TLS 证书"

了解更多信息

- "使用 Astra Control API"
- "已知问题"

添加集群的前提条件

在添加集群之前，应确保满足前提条件。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。

添加集群之前需要满足的要求

- 以下类型的集群之一：
  - 运行OpenShift 4.6-8、4.7、4.8或4.9的集群
  - 使用RKE1运行Rancher 2.2.8、2.0.9或2.6的集群
  - 运行Kubernetes 1.20到1.23的集群
  - 运行VMware Tanzu Kubernetes Grid 1.4的集群
  - 运行VMware Tanzu Kubernetes Grid Integrated Edition 1.12.2的集群

  确保集群中有一个或多个工作节点，并且至少有 1 GB RAM 可用于运行遥测服务。

> 如果您计划将第二个 OpenShift 4.6 ， 4.7 或 4.8 集群添加为托管计算资源，则应确保已启用 Astra Trident 卷快照功能。请参见官方的 Astra Trident "说明" 使用 Astra Trident 启用和测试卷快照。

- 使用配置了的Astra Trident StorageClasses "支持的存储后端" (对于任何类型的集群都是必需的)
- 在备份 ONTAP 系统上设置的超级用户和用户 ID ，用于使用 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 sysm -anon 65534`
- 管理员定义的 Astra Trident `volumesnapshotclass` 对象。请参见 Astra Trident "说明" 使用 Astra Trident 启用和测试卷快照。
- 确保您仅为 Kubernetes 集群定义了一个默认存储类。

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   21.04.0
```

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. 检查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
NAME                    PROVISIONER                    RECLAIMPOLICY
VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
ontap-gold (default)    csi.trident.netapp.io          Delete
Immediate           true                   5d23h
thin                    kubernetes.io/vsphere-volume   Delete
Immediate           false                  6d
```

创建管理员角色 **kubeconfig**

执行这些步骤之前，请确保您的计算机上具有以下内容：

- 已安装 `kubectl` v1.19 或更高版本
- 具有活动上下文集群管理员权限的活动 kubeconfig

步骤

1. 按如下所示创建服务帐户：

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

      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. （可选）如果集群使用限制性的 POD 安全策略，该策略不允许创建特权 POD 或允许 Pod 容器中的进程以 root 用户身份运行，请为集群创建一个自定义 POD 安全策略，以使 Astra Control 能够创建和管理 Pod 。有关说明，请参见 "创建自定义 POD 安全策略"。
3. 按如下所示授予集群管理员权限：

   a. 创建一个 `ClusterRoleBindingm` 文件，该文件名为 `astracontrol — clusterrolebind.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: cluster-admin
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，将 ` <context>` 替换为适用于您的安装的正确上下文：

```
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"}
]
```

sec白 烟 数组中每个元素的索引均以 0 开头。在上面的示例中， `asacontrol service-account-dockercfg-vhz87` 的索引为 0 ， `asacontrol service-account-token-r59rk` 的索引为 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。将以下脚本开头的 `token_index` 替换为正确的值。

```
<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。保护集群凭据。

```
chmod 700 create-kubeconfig.sh
mv kubeconfig-sa.txt YOUR_CLUSTER_NAME_kubeconfig
```

下一步是什么？

确认满足了这些前提条件后，您便已准备就绪 "添加集群"。

了解更多信息

- "Trident 文档"
- "使用 Astra Control API"

添加自定义 TLS 证书

您可以删除现有的自签名 TLS 证书，并将其替换为由证书颁发机构（ CA ）签名的 TLS 证书。

您需要的内容

- 安装了 Astra 控制中心的 Kubernetes 集群
- 对集群上的命令 Shell 进行管理访问，以运行 `kubectl` 命令
- CA 中的专用密钥和证书文件

删除自签名证书

删除现有的自签名 TLS 证书。

1. 使用 SSH ，以管理用户身份登录到托管 Astra 控制中心的 Kubernetes 集群。
2. 使用以下命令查找与当前证书关联的 TLS 密钥，并将 ` <Acc-deployment-namespace>` 替换为 Astra Control Center 部署命名空间：

```
kubectl get certificate -n <ACC-deployment-namespace>
```

3. 使用以下命令删除当前安装的密钥和证书：

```
kubectl delete cert cert-manager-certificates -n <ACC-deployment-
namespace>
kubectl delete secret secure-testing-cert -n <ACC-deployment-namespace>
```

添加新证书

添加一个由 CA 签名的新 TLS 证书。

1. 使用以下命令使用 CA 中的专用密钥和证书文件创建新的 TLS 密钥，并将括号 <> 中的参数替换为相应的信息：

```
kubectl create secret tls <secret-name> --key <private-key-filename>
--cert <certificate-filename> -n <ACC-deployment-namespace>
```

2. 使用以下命令和示例编辑集群自定义资源定义（ CRD ）文件，并将 `spec.selfSigned` 值更改为 `spec.ca.secretName` ，以引用您先前创建的 TLS 密钥：

```
kubectl edit clusterissuers.cert-manager.io/cert-manager-certificates -n
<ACC-deployment-namespace>
....

#spec:
#  selfSigned: {}

spec:
  ca:
    secretName: <secret-name>
```

3. 使用以下命令和示例输出验证所做的更改是否正确以及集群是否已准备好验证证书，并将 ` <Acc-deployment-namespace>` 替换为 Astra Control Center 部署命名空间：

```
kubectl describe clusterissuers.cert-manager.io/cert-manager-
certificates -n <ACC-deployment-namespace>
....

Status:
  Conditions:
    Last Transition Time:  2021-07-01T23:50:27Z
    Message:               Signing CA verified
    Reason:                KeyPairVerified
    Status:                True
    Type:                  Ready
Events:                    <none>
```

4. 使用以下示例创建 `certificate.yaml` 文件，将括号中的占位值替换为相应的信息：

```
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  name: <certificate-name>
  namespace: <ACC-deployment-namespace>
spec:
  secretName: <certificate-secret-name>
  duration: 2160h # 90d
  renewBefore: 360h # 15d
  dnsNames:
  - <astra.dnsname.example.com> #Replace with the correct Astra Control
Center DNS address
  issuerRef:
    kind: ClusterIssuer
    name: cert-manager-certificates
```

5. 使用以下命令创建证书：

```
kubectl apply -f certificate.yaml
```

6. 使用以下命令和示例输出，验证是否已正确创建证书以及是否使用您在创建期间指定的参数（例如名称，持续时间，续订截止日期和 DNS 名称）。

```
kubectl describe certificate -n <ACC-deployment-namespace>
....

Spec:
  Dns Names:
    astra.example.com
  Duration:  125h0m0s
  Issuer Ref:
    Kind:        ClusterIssuer
    Name:        cert-manager-certificates
  Renew Before:  61h0m0s
  Secret Name:   <certificate-secret-name>
Status:
  Conditions:
    Last Transition Time:  2021-07-02T00:45:41Z
    Message:               Certificate is up to date and has not expired
    Reason:                Ready
    Status:                True
    Type:                  Ready
  Not After:               2021-07-07T05:45:41Z
  Not Before:              2021-07-02T00:45:41Z
  Renewal Time:            2021-07-04T16:45:41Z
  Revision:                1
Events:                    <none>
```

7. 使用以下命令和示例编辑传入 CRD TLS 选项以指向新的证书密钥，并将括号 <> 中的占位符值替换为相应的信息：

```
kubectl edit ingressroutes.traefik.containo.us -n <ACC-deployment-
namespace>
....

# tls:
#    options:
#      name: default
#    secretName: secure-testing-cert
#    store:
#      name: default

 tls:
    options:
      name: default
    secretName: <certificate-secret-name>
    store:
      name: default
```

8. 使用 Web 浏览器浏览到 Astra 控制中心的部署 IP 地址。
9. 验证证书详细信息是否与您安装的证书的详细信息匹配。
10. 导出证书并将结果导入到 Web 浏览器中的证书管理器中。

创建自定义 POD 安全策略

Astra Control 需要在其管理的集群上创建和管理 Kubernetes Pod 。如果集群使用的限制性 POD 安全策略不允许创建特权 POD 或允许 Pod 容器中的进程以 root 用户身份运行，则需要创建限制性较低的 POD 安全策略，以使 Astra Control 能够创建和管理这些 Pod 。

步骤

1. 为集群创建一个限制性低于默认值的 POD 安全策略，并将其保存在文件中。例如：

```yaml
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: astracontrol
  annotations:
    seccomp.security.alpha.kubernetes.io/allowedProfileNames: '*'
spec:
  privileged: true
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  volumes:
  - '*'
  hostNetwork: true
  hostPorts:
  - min: 0
    max: 65535
  hostIPC: true
  hostPID: true
  runAsUser:
    rule: 'RunAsAny'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'RunAsAny'
  fsGroup:
    rule: 'RunAsAny'
```

2. 为 POD 安全策略创建新角色。

```
kubectl-admin create role psp:astracontrol \
    --verb=use \
    --resource=podsecuritypolicy \
    --resource-name=astracontrol
```

3. 将新角色绑定到服务帐户。

```
kubectl-admin create rolebinding default:psp:astracontrol \
    --role=psp:astracontrol \
    --serviceaccount=astracontrol-service-account:default
```

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 Control Center 时，在 Astra_control_center_min.YAML 自定义资源定义（ CRD ）文件的 spec.astraAddress 字段中输入您设置的完全限定域名（ FQDN ）。电子邮件是您在 Astra_control_center_min.YAML CRD 的 spec.email 字段中设置的值。

- 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 *
- 从左侧导航栏中，选择 * 帐户 * > * 许可证 * 。
- 选择 * 添加许可证 * 。
- 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。您也可以使用 `kubectl get nodes` 命令验证新工作节点的运行状况。

- 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

- 我的帐户中存在未创建的快照。它们来自何处？ *

在某些情况下， Astra Control 会在备份，克隆或还原过程中自动创建快照。

- 我的应用程序使用多个 PV 。Astra Control 是否会为所有这些 PVC 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。