



添加集群 Astra Control Service

NetApp
April 24, 2024

目录

- 将集群添加到Asta Control Service 1
 - 安装Asta Connector以管理集群 1
 - 添加由提供程序管理的集群 7
 - 添加自我管理集群 16

将集群添加到Asta Control Service

设置环境后，您可以创建 Kubernetes 集群，然后将其添加到 Astra Control Service 。这样、您就可以使用Astra Control Service来保护集群上的应用程序。

根据您需要添加到Astra Control Service的集群类型、您需要使用不同的步骤来添加集群。

- ["将公共提供商管理的集群添加到Astra Control Service"](#)：使用以下步骤添加具有公共IP地址且由云提供商管理的集群。您需要云提供商的服务主体帐户、服务帐户或用户帐户。
- ["将私有提供商管理的集群添加到Astra Control Service"](#)：使用以下步骤添加具有专用IP地址且由云提供商管理的集群。您需要云提供商的服务主体帐户、服务帐户或用户帐户。
- ["将公共自管理集群添加到Astra Control Service"](#)：使用以下步骤添加具有公共IP地址且由您的组织管理的集群。您需要为要添加的集群创建一个kubeconfigfile文件。
- ["将私有自管理集群添加到Astra Control Service"](#)：使用以下步骤添加具有专用IP地址且由您的组织管理的集群。您需要为要添加的集群创建一个kubeconfigfile文件。

安装Asta Connector以管理集群

Asta Connector是一款位于受管集群上的软件、可促进受管集群与Asta Control之间的通信。对于使用Asta Control Service管理的集群、Asta Connector有两个可用版本：

- *Asta Connector*的先前版本：["安装旧版本的Asta Connector"](#) 如果您计划使用非Kubernetes本机工作流管理集群、请在集群上执行此操作。
- 声明性Kubernetes Asta Connector：["为使用声明性Kubarnetes工作流管理的集群安装Asta Connector"](#) 如果您计划使用声明性Kubarnetes工作流管理集群、请在集群上执行此操作。在集群上安装Asta Connector后、该集群将自动添加到Asta Control中。



声明性Kubarnetes Asta Connector仅作为Asta Control早期采用者计划(EAP)的一部分提供。有关加入NetApp的信息、请联系您的EAP销售代表。

安装旧版本的Asta Connector

Asta Control Service使用以前版本的Asta Connector在Astra Control Service和使用非Kubernetes本机工作流管理的专用集群之间实现通信。您需要在要使用非Kubbernetes本机工作流管理的专用集群上安装Asta Connector。

早期版本的Asta Connector支持使用非Kubenetes本机工作流管理的以下类型的专用集群：

- Amazon Elelic Kubelnetes Service (EKS)
- Azure Kubernetes Service (AKS)
- Google Kubernetes Engine (GKEE)
- 基于AWS的Red Hat OpenShift服务(ROSA)
- 具有AWS PrivateLink的罗莎
- Red Hat OpenShift容器平台内部部署

关于此任务

- 执行这些步骤时、请对要使用Astra Control Service管理的专用集群执行这些命令。
- 如果使用的是Bastion主机、请从Bastion主机的命令行对这些命令执行问题描述。

开始之前

- 您需要访问要使用Astra Control Service管理的专用集群。
- 要在集群上安装Astra Connector操作员、您需要具有Kubbernetes管理员权限。

步骤

1. 在要使用非Kubbernetes本机工作流管理的专用集群上安装先前的Astra Connector运算符。运行此命令时、命名空间 `astra-connector-operator` 创建并将配置应用于命名空间：

```
kubectl apply -f https://github.com/NetApp/astra-connector-operator/releases/download/23.07.0-202310251519/astraconnector_operator.yaml
```

2. 确认操作员已安装并准备就绪：

```
kubectl get all -n astra-connector-operator
```

3. 从Astra Control获取API令牌。请参见 "[Astra Automation文档](#)" 有关说明，请参见。
4. 创建Astra-connector命名空间：

```
kubectl create ns astra-connector
```

5. 创建Astra Connector CR文件并将其命名为 `astra-connector-cr.yaml`。更新方括号<>中的值以匹配您的Astra Control环境和集群配置：

- **Astra：** 控制服务的Web UI <ASTRA_CONTROL_SERVICE_URL>。例如：

```
https://astra.netapp.io
```

- **Astra：** 您在上一步中获得的<ASTRA_CONTROL_SERVICE_API_TOKEN>标记。
- **AzAzure Kubermet：** (仅限集群)—专用<PRIVATE_AKS_CLUSTER_NAME>集群的集群名称。只有在添加专用AKS集群时、才会取消注释并填充此行。
- **Astra：** 从<ASTRA_CONTROL_ACCOUNT_ID> Web UI获取。选择页面右上角的图图标，然后选择*API access*。

```

apiVersion: netapp.astraconnector.com/v1
kind: AstraConnector
metadata:
  name: astra-connector
  namespace: astra-connector
spec:
  natssync-client:
    cloud-bridge-url: <ASTRA_CONTROL_SERVICE_URL>
  imageRegistry:
    name: theotw
    secret: ""
  astra:
    token: <ASTRA_CONTROL_SERVICE_API_TOKEN>
    #clusterName: <PRIVATE_AKS_CLUSTER_NAME>
    accountId: <ASTRA_CONTROL_ACCOUNT_ID>
    acceptEULA: yes

```

6. 在您填充之后 astra-connector-cr.yaml 使用正确值的文件、应用CR:

```
kubectl apply -f astra-connector-cr.yaml
```

7. 验证Asta Connector是否已完全部署:

```
kubectl get all -n astra-connector
```

8. 验证集群是否已注册到Astra Control:

```
kubectl get astraconnector -n astra-connector
```

您应看到类似于以下内容的输出:

NAME	REGISTERED	ASTRACONNECTORID
STATUS		
astra-connector	true	be475ae5-1511-4eaa-9b9e-712f09b0d065
Registered with Astra		



记下ASTRACONNECTRID; 将集群添加到Astra Control时将需要此ID。

下一步是什么？

现在您已安装Astra Connector、可以将私有集群添加到Astra Control Service了。

- ["将私有提供商管理的集群添加到Astra Control Service"](#)：使用以下步骤添加具有专用IP地址且由云提供商管理的集群。您需要云提供商的服务主体帐户、服务帐户或用户帐户。
- ["将私有自我管理集群添加到Astra Control Service"](#)：使用以下步骤添加具有专用IP地址且由您的组织管理的集群。您需要为要添加的集群创建一个kubeconfigfile文件。

有关详细信息 ...

- ["添加集群"](#)

(技术预览)安装声明性Kubarnetes Astra Connector

使用声明性Kubarnetes工作流管理的集群使用Astra Connector在受管集群和Astra Control之间实现通信。您需要在要使用声明性Kubarnetes工作流管理的所有集群上安装Astra Connector。

您可以使用Kubarnetes命令和自定义资源(Custom Resource、CR)文件安装声明性Kubarnetes Astra Connector。

关于此任务

- 执行这些步骤时、请在要使用Astra Control进行管理的集群上执行这些命令。
- 如果使用的是Bastion主机、请从Bastion主机的命令行对这些命令执行问题描述。

开始之前

- 您需要访问要使用Astra Control管理的集群。
- 要在集群上安装Astra Connector操作员、您需要具有Kubbernetes管理员权限。



如果为集群配置了强制实施Pod安全接入(这是Kubernetes 1.25及更高版本集群的默认设置)、则需要对相应的卷空间启用PSA限制。请参见 ["使用Astra Control准备用于集群管理的环境"](#) 有关说明，请参见。

步骤

1. 在要使用声明性Kubarnetes工作流管理的集群上安装A作用 连接器操作员。运行此命令时、命名空间 `astra-connector-operator` 创建并将配置应用于命名空间：

```
kubectl apply -f https://github.com/NetApp/astra-connector-operator/releases/download/24.02.0-202403151353/astraconnector_operator.yaml
```

2. 确认操作员已安装并准备就绪：

```
kubectl get all -n astra-connector-operator
```

3. 从Astra Control获取API令牌。请参见 "[Astra Automation文档](#)" 有关说明，请参见。
4. 使用令牌创建密钥。将<API_TOKEN>替换为您从Astra Control收到的令牌：

```
kubectl create secret generic astra-token \
--from-literal=apiToken=<API_TOKEN> \
-n astra-connector
```

5. 创建Docker密钥以提取Astra Connector映像。将括号<>中的值替换为您环境中的信息：



您可以在Astra Control Web UI中找到<ASTRA_CONTROL_ACCOUNT_ID>。在Web UI中，选择页面右上角的图图标，然后选择*API access*。

```
kubectl create secret docker-registry regcred \
--docker-username=<ASTRA_CONTROL_ACCOUNT_ID> \
--docker-password=<API_TOKEN> \
-n astra-connector \
--docker-server=cr.astra.netapp.io
```

6. 创建Astra Connector CR文件并将其命名为 `astra-connector-cr.yaml`。更新方括号<>中的值以匹配您的Astra Control环境和集群配置：

- <ASTRA_CONTROL_ACCOUNT_ID>：在上一步中从Astra Control Web UI获取。
- <CLUSTER_NAME>：应在Astra Control中分配此集群的名称。
- <ASTRA_CONTROL_URL>：Astra Control的Web UI URL。例如：

```
https://astra.control.url
```

```

apiVersion: astra.netapp.io/v1
kind: AstraConnector
metadata:
  name: astra-connector
  namespace: astra-connector
spec:
  astra:
    accountId: <ASTRA_CONTROL_ACCOUNT_ID>
    clusterName: <CLUSTER_NAME>
    #Only set `skipTLSValidation` to `true` when using the default
    self-signed
    #certificate in a proof-of-concept environment.
    skipTLSValidation: false #Should be set to false in production
    environments
    tokenRef: astra-token
  natsSyncClient:
    cloudBridgeURL: <ASTRA_CONTROL_HOST_URL>
  imageRegistry:
    name: cr.astra.netapp.io
    secret: regcred

```

7. 在您填充之后 astra-connector-cr.yaml 使用正确值的文件、应用CR:

```
kubectl apply -n astra-connector -f astra-connector-cr.yaml
```

8. 验证Asta Connector是否已完全部署:

```
kubectl get all -n astra-connector
```

9. 验证集群是否已注册到Astra Control:

```
kubectl get astraconnectors.astra.netapp.io -A
```

您应看到类似于以下内容的输出:

NAMESPACE	NAME	REGISTERED	ASTRACONNECTORID
STATUS			
astra-connector	astra-connector	true	00ac8-2cef-41ac-8777-ed0583e
	Registered with Astra		

10. 验证该集群是否显示在Astra Control Web UI的*集群*页面上的受管集群列表中。

添加由提供程序管理的集群

将公共提供商管理的集群添加到Astra Control Service

设置云环境后、您可以创建Kubennetes集群、然后将其添加到Astra Control Service中。

- [创建 Kubernetes 集群](#)
- [将集群添加到Astra Control Service](#)
- [\[更改默认存储类\]](#)

创建 Kubernetes 集群

如果您还没有集群、则可以创建满足要求的集群 "[Amazon Elastic Kubernetes Service \(EKS\)的Astra Control Service要求](#)"。如果您还没有集群、则可以创建满足要求的集群 "[Google Kubernetes Engine （ GKEE ） 的 Astra Control Service 要求](#)"。如果您还没有集群、则可以创建满足要求的集群 "[采用 Azure NetApp Files 的 Azure Kubernetes Service （ AKS ） 的 Astra 控制服务要求](#)" 或 "[采用 Azure 受管磁盘的 Azure Kubernetes Service （ AKS ） 的 Astra Control Service 要求](#)"。



Astra控制服务支持使用Azure Active Directory (Azure AD)进行身份验证和身份管理的AKS集群。创建集群时、请按照中的说明进行操作 "[正式文档](#)" 将集群配置为使用Azure AD。您需要确保集群满足AKS管理的Azure AD集成的要求。

将集群添加到Astra Control Service

登录到 Astra Control Service 后，第一步是开始管理集群。在将集群添加到Astra Control Service之前、您需要执行特定任务并确保集群满足特定要求。

在管理Azure Kubenetes Service和Google Kubenetes Engine集群时、请注意、Astra Control配置程序安装和生命周期管理有两种选择：

- 您可以使用Astra Control Service自动管理Astra Control置管程序的生命周期。要执行此操作、请确保未在要使用Astra Control Service管理的集群上安装Astra Trent、并且未启用Astra Control配置程序。在这种情况下、Astra Control Service会在您开始管理集群时自动启用Astra Control配置程序、并自动处理Astra Control配置程序升级。
- 您可以自行管理Astra Control配置程序的生命周期。为此、请先在集群上启用Astra Control配置程序、然后再使用Astra Control Service管理集群。在这种情况下、Astra Control Service检测到Astra Control配置程序已启用、不会重新安装它或管理Astra Control配置程序升级。请参见 "[启用Astra Control配置程序](#)" 有关步骤、请启用Astra Control配置程序。

在使用Astra Control Service管理Amazon Web Services集群时、如果需要只能与Astra Control配置程序一起使用的存储后端、则需要先在集群上手动启用Astra Control配置程序、然后再使用Astra Control Service进行管理。请参见 "[启用Astra Control配置程序](#)" 了解启用Astra Control配置程序的步骤。

Amazon Web Services

- 您应拥有包含创建集群的IAM用户凭据的JSON文件。 ["了解如何创建IAM用户"](#)。
- Amazon FSx for NetApp ONTAP需要Astra Control配置程序。如果您计划使用Amazon FSx for NetApp ONTAP作为EKS集群的存储后端、请参阅中的Astra Control配置程序信息 ["EKS集群要求"](#)。
- (可选)如果需要提供 `kubectl` 集群对非集群创建者的其他IAM用户的命令访问权限、请参见中的说明 ["在Amazon EKS中创建集群后、如何为其他IAM用户和角色提供访问权限?"](#)。
- 如果您计划使用NetApp Cloud Volumes ONTAP 作为存储后端、则需要将Cloud Volumes ONTAP 配置为使用Amazon Web Services。请参见Cloud Volumes ONTAP ["设置文档"](#)。

Microsoft Azure

- 您应拥有包含在创建服务主体时Azure命令行界面输出的JSON文件。 ["了解如何设置服务主体"](#)。

如果未将 Azure 订阅 ID 添加到 JSON 文件中，您也需要此 ID 。

- 如果您计划使用NetApp Cloud Volumes ONTAP 作为存储后端、则需要将Cloud Volumes ONTAP 配置为与Microsoft Azure配合使用。请参见Cloud Volumes ONTAP ["设置文档"](#)。

Google Cloud

- 您应拥有具有所需权限的服务帐户的服务帐户密钥文件。 ["了解如何设置服务帐户"](#)。
- 如果您计划使用NetApp Cloud Volumes ONTAP 作为存储后端、则需要将Cloud Volumes ONTAP 配置为与Google Cloud配合使用。请参见Cloud Volumes ONTAP ["设置文档"](#)。

步骤

1. (可选)如果要添加Amazon EKS集群或要自行管理Astra Control配置程序的安装和升级、请在此集群上启用Astra Control配置程序。请参见 ["启用Astra Control配置程序"](#) 了解支持步骤。
2. 在浏览器中打开Astra Control Service Web UI。
3. 在信息板上，选择 * 管理 Kubernetes 集群 * 。

按照提示添加集群。

4. 提供商：选择您的云提供商、然后提供创建新云实例所需的凭据或选择要使用的现有云实例。
5. * Amazon Web Services*：上传JSON文件或从剪贴板粘贴JSON文件的内容、以提供有关Amazon Web Services IAM用户帐户的详细信息。

JSON文件应包含创建集群的IAM用户的凭据。

6. * Microsoft Azure*：通过上传 JSON 文件或从剪贴板粘贴此 JSON 文件的内容来提供有关 Azure 服务主体的详细信息。

JSON 文件应包含创建服务主体时 Azure 命令行界面的输出。它还可以包含您的订阅 ID ，以便自动添加到 Astra 。否则，您需要在提供 JSON 后手动输入 ID 。

7. * Google Cloud Platform*：通过上传文件或粘贴剪贴板中的内容来提供服务帐户密钥文件。

Astra 控制服务使用此服务帐户发现在 Google Kubernetes Engine 中运行的集群。

8. 其他：此选项卡仅适用于自行管理的集群。

- a. 云实例名称：为要在添加此集群时创建的新云实例提供一个名称。了解更多信息 ["云实例"](#)。
- b. 选择 * 下一步 *。

Astra Control Service会显示一个集群列表、您可以从中进行选择。

c. 集群：从列表中选择要添加到Astra Control Service的集群。



从集群列表中选择时，请注意*Eligibility*列。如果集群"不符合条件"或"部分符合条件"、请将鼠标悬停在状态上方以确定集群是否具有问题描述。例如，它可能会标识集群没有工作节点。

d. 选择 * 下一步 *。

e. (可选)存储：(可选)选择默认情况下希望部署到此集群中的Kubernetes应用程序使用的存储类。

9. 要为集群选择新的默认存储类，请启用*Assign a new default storage class*复选框。

10. 从列表中选择新的默认存储类。



每个云提供商存储服务都会显示以下价格、性能和弹性信息：

- Cloud Volumes Service for Google Cloud：价格、性能和弹性信息
- Google Persistent Disk：没有价格、性能或弹性信息
- Azure NetApp Files：性能和弹性信息
- Azure受管磁盘：无可用的价格、性能或弹性信息
- Amazon Elastic Block Store：没有价格、性能或弹性信息
- 适用于NetApp ONTAP 的Amazon FSX：没有价格、性能或弹性信息
- NetApp Cloud Volumes ONTAP：没有价格、性能或弹性信息

每个存储类均可使用以下服务之一：

- ["适用于 Google Cloud 的 Cloud Volumes Service"](#)
- ["Google 持久磁盘"](#)
 - ["Azure NetApp Files"](#)
 - ["Azure 受管磁盘"](#)
 - ["Amazon Elastic Block Store"](#)
 - ["适用于 NetApp ONTAP 的 Amazon FSX"](#)
 - ["NetApp Cloud Volumes ONTAP"](#)

了解更多信息 ["Amazon Web Services集群的存储类"](#)。了解更多信息 ["AKS 集群的存储类"](#)。了解更多信息 ["GKE 集群的存储类"](#)。

a. 选择 * 下一步 *。

b. 审核和批准：审核配置详细信息。

c. 选择*Add*将集群添加到Astra Control Service。

结果

如果这是您为此云提供程序添加的第一个集群、Astra Control Service将为此云提供程序创建一个对象存储、用于备份在符合条件的集群上运行的应用程序。(在为此云提供程序添加后续集群时、不会再创建其他对象存储。)如果指定了默认存储类、则Astra控制服务将设置您指定的默认存储类。对于在Amazon Web Services或Google Cloud Platform中管理的集群、Astra Control Service还会在集群上创建管理员帐户。这些操作可能需要几分钟时间。

更改默认存储类

您可以更改集群的默认存储类。

使用Astra Control更改默认存储类

您可以在Astra Control中更改集群的默认存储类。如果集群使用先前安装的存储后端服务、则可能无法使用此方法更改默认存储类(不能选择*设置为默认值*操作)。在这种情况下、您可以 [\[使用命令行更改默认存储类\]](#)。

步骤

1. 在 Astra 控制服务 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*。

使用命令行更改默认存储类

您可以使用Kubernetes命令更改集群的默认存储类。无论集群的配置如何、此方法都有效。

步骤

1. 登录到Kubernetes集群。
2. 列出集群中的存储类：

```
kubectl get storageclass
```

3. 从默认存储类中删除默认指定。将<SC_NAME> 替换为存储类的名称：

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":  
{"annotations":{"storageclass.kubernetes.io/is-default-  
class":"false"}}}'
```

4. 将其他存储类标记为默认值。将<SC_NAME> 替换为存储类的名称：

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":  
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

5. 确认新的默认存储类：

```
kubectl get storageclass
```

将私有提供商管理的集群添加到Astra Control Service

您可以使用Astra Control Service管理专用Google Kubernetes Engine (GKE)集群。以下说明假定您已创建专用AKS或OpenShift集群、并准备了一种安全方法来远程访问该集群；有关创建和访问专用AKS或OpenShift集群的详细信息、请参阅以下文档：

- ["适用于私有AKS集群的Azure文档"](#)
- ["适用于私有OpenShift集群的Azure文档"](#)

您可以使用Astra Control Service管理私有Azure Kubernetes Service (AKS)集群以及AKS中的私有Red Hat OpenShift集群。以下说明假定您已创建专用AKS或OpenShift集群、并准备了一种安全方法来远程访问该集群；有关创建和访问专用AKS或OpenShift集群的详细信息、请参阅以下文档：

- ["适用于私有AKS集群的Azure文档"](#)
- ["适用于私有OpenShift集群的Azure文档"](#)

您可以使用Astra Control Service管理专用Amazon Elastic Kubernetes Service (EKS)集群。以下说明假定您已创建一个专用EKS集群、并已准备好一种安全方法来远程访问该集群；有关创建和访问专用EKS集群的详细信息、请参阅 ["Amazon EKS文档"](#)。

要将专用集群添加到Astra Control Service、您需要执行以下任务：

1. [安装A作用 连接器](#)
2. [\[设置永久性存储\]](#)
3. [将私有提供商管理的集群添加到Astra Control Service](#)

安装A作用 连接器

在添加专用集群之前、您需要在此集群上安装Astra Connector、以便Astra Control可以与其通信。请参见 ["为使用非Kubnetes本机工作流管理的专用集群安装以前版本的Astra Connector"](#) 有关说明，请参见。

设置永久性存储

为集群配置永久性存储。有关配置永久性存储的详细信息、请参见入门文档：

- ["使用 Azure NetApp Files 设置 Microsoft Azure"](#)
- ["使用 Azure 受管磁盘设置 Microsoft Azure"](#)

- ["设置Amazon Web Services"](#)
- ["设置 Google Cloud"](#)

将私有提供商管理的集群添加到**Astra Control Service**

现在、您可以将专用集群添加到Astra Control Service。

在管理Azure Kubernetes Service和Google Kubernetes Engine集群时、请注意、Astra Control配置程序安装和生命周期管理有两种选择：

- 您可以使用Astra Control Service自动管理Astra Control置管程序的生命周期。要执行此操作、请确保未在要使用Astra Control Service管理的集群上安装Astra Trent、并且未启用Astra Control配置程序。在这种情况下、Astra Control Service会在您开始管理集群时自动启用Astra Control配置程序、并自动处理Astra Control配置程序升级。
- 您可以自行管理Astra Control配置程序的生命周期。为此、请先在集群上启用Astra Control配置程序、然后再使用Astra Control Service管理集群。在这种情况下、Astra Control Service检测到Astra Control配置程序已启用、不会重新安装它或管理Astra Control配置程序升级。请参见 ["启用Astra Control配置程序"](#) 有关步骤、请启用Astra Control配置程序。

在使用Astra Control Service管理Amazon Web Services集群时、如果需要只能与Astra Control配置程序一起使用的存储后端、则需要先在集群上手动启用Astra Control配置程序、然后再使用Astra Control Service进行管理。请参见 ["启用Astra Control配置程序"](#) 了解启用Astra Control配置程序的步骤。

Amazon Web Services

- 您应拥有包含创建集群的IAM用户凭据的JSON文件。 ["了解如何创建IAM用户"](#)。
- Amazon FSx for NetApp ONTAP需要Astra Control配置程序。如果您计划使用Amazon FSx for NetApp ONTAP作为EKS集群的存储后端、请参阅中的Astra Control配置程序信息 ["EKS集群要求"](#)。
- (可选)如果需要提供 `kubectl` 集群对非集群创建者的其他IAM用户的命令访问权限、请参见中的说明 ["在Amazon EKS中创建集群后、如何为其他IAM用户和角色提供访问权限?"](#)。
- 如果您计划使用NetApp Cloud Volumes ONTAP 作为存储后端、则需要将Cloud Volumes ONTAP 配置为使用Amazon Web Services。请参见Cloud Volumes ONTAP ["设置文档"](#)。

Microsoft Azure

- 您应拥有包含在创建服务主体时Azure命令行界面输出的JSON文件。 ["了解如何设置服务主体"](#)。

如果未将 Azure 订阅 ID 添加到 JSON 文件中，您也需要此 ID 。

- 如果您计划使用NetApp Cloud Volumes ONTAP 作为存储后端、则需要将Cloud Volumes ONTAP 配置为与Microsoft Azure配合使用。请参见Cloud Volumes ONTAP ["设置文档"](#)。

Google Cloud

- 您应拥有具有所需权限的服务帐户的服务帐户密钥文件。 ["了解如何设置服务帐户"](#)。
- 如果集群为专用集群，则会显示 ["授权网络"](#) 必须允许 Astra 控制服务 IP 地址：

52.188.218.166/32
- 如果您计划使用NetApp Cloud Volumes ONTAP 作为存储后端、则需要将Cloud Volumes ONTAP 配置为与Google Cloud配合使用。请参见Cloud Volumes ONTAP ["设置文档"](#)。

步骤

1. (可选)如果要添加Amazon EKS集群或要自行管理Astra Control配置程序的安装和升级、请在此集群上启用Astra Control配置程序。请参见 ["启用Astra Control配置程序"](#) 了解支持步骤。
2. 在浏览器中打开Astra Control Service Web UI。
3. 在信息板上，选择 * 管理 Kubernetes 集群 *。

按照提示添加集群。

4. 提供商：选择您的云提供商、然后提供创建新云实例所需的凭据或选择要使用的现有云实例。
5. * Amazon Web Services*：上传JSON文件或从剪贴板粘贴JSON文件的内容、以提供有关Amazon Web Services IAM用户帐户的详细信息。

JSON文件应包含创建集群的IAM用户的凭据。

6. * Microsoft Azure*：通过上传 JSON 文件或从剪贴板粘贴此 JSON 文件的内容来提供有关 Azure 服务主体的详细信息。

JSON 文件应包含创建服务主体时 Azure 命令行界面的输出。它还可以包含您的订阅 ID ，以便自动添加到 Astra 。否则，您需要在提供 JSON 后手动输入 ID 。

7. * Google Cloud Platform*：通过上传文件或粘贴剪贴板中的内容来提供服务帐户密钥文件。

Astra 控制服务使用此服务帐户发现在 Google Kubernetes Engine 中运行的集群。

8. 其他：此选项卡仅适用于自行管理的集群。

- a. 云实例名称：为要在添加此集群时创建的新云实例提供一个名称。了解更多信息 ["云实例"](#)。
- b. 选择 * 下一步 *。

Astra Control Service会显示一个集群列表、您可以从中进行选择。

c. 集群：从列表中选择要添加到Astra Control Service的集群。



从集群列表中选择时，请注意*Eligibility*列。如果集群"不符合条件"或"部分符合条件"、请将鼠标悬停在状态上方以确定集群是否具有问题描述。例如，它可能会标识集群没有工作节点。

9. 选择 * 下一步 *。

10. (可选)存储：(可选)选择默认情况下希望部署到此集群中的Kubernetes应用程序使用的存储类。

- a. 要为集群选择新的默认存储类，请启用*Assign a new default storage class*复选框。
- b. 从列表中选择新的默认存储类。



每个云提供商存储服务都会显示以下价格、性能和弹性信息：

- Cloud Volumes Service for Google Cloud：价格、性能和弹性信息
- Google Persistent Disk：没有价格、性能或弹性信息
- Azure NetApp Files：性能和弹性信息
- Azure受管磁盘：无可用的价格、性能或弹性信息
- Amazon Elastic Block Store：没有价格、性能或弹性信息
- 适用于NetApp ONTAP 的Amazon FSX：没有价格、性能或弹性信息
- NetApp Cloud Volumes ONTAP：没有价格、性能或弹性信息

每个存储类均可使用以下服务之一：

- ["适用于 Google Cloud 的 Cloud Volumes Service"](#)
- ["Google 持久磁盘"](#)
- ["Azure NetApp Files"](#)
- ["Azure 受管磁盘"](#)
- ["Amazon Elastic Block Store"](#)
- ["适用于 NetApp ONTAP 的 Amazon FSX"](#)
- ["NetApp Cloud Volumes ONTAP"](#)

了解更多信息 ["Amazon Web Services集群的存储类"](#)。了解更多信息 ["AKS 集群的存储类"](#)。了解更多信息 ["GKE 集群的存储类"](#)。

- c. 选择 * 下一步 *。
- d. 审核和批准：审核配置详细信息。
- e. 选择*Add*将集群添加到Astra Control Service。

结果

如果这是您为此云提供程序添加的第一个集群、Astra Control Service将为此云提供程序创建一个对象存储、用于备份在符合条件的集群上运行的应用程序。(在为此云提供程序添加后续集群时、不会再创建其他对象存储。)如果指定了默认存储类、则Astra控制服务将设置您指定的默认存储类。对于在Amazon Web Services或Google Cloud Platform中管理的集群、Astra Control Service还会在集群上创建管理员帐户。这些操作可能需要几分钟时间。

更改默认存储类

您可以更改集群的默认存储类。

使用Astra Control更改默认存储类

您可以在Astra Control中更改集群的默认存储类。如果集群使用先前安装的存储后端服务、则可能无法使用此方法更改默认存储类(不能选择*设置为默认值*操作)。在这种情况下、您可以 [\[使用命令行更改默认存储类\]](#)。

步骤

1. 在 Astra 控制服务 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*。

使用命令行更改默认存储类

您可以使用Kubernetes命令更改集群的默认存储类。无论集群的配置如何、此方法都有效。

步骤

1. 登录到Kubernetes集群。
2. 列出集群中的存储类：

```
kubectl get storageclass
```

3. 从默认存储类中删除默认指定。将<SC_NAME> 替换为存储类的名称：

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":  
{"annotations":{"storageclass.kubernetes.io/is-default-  
class":"false"}}}'
```

4. 将其他存储类标记为默认值。将<SC_NAME> 替换为存储类的名称：

```
kubectl patch storageclass <SC_NAME> -p '{"metadata": {"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

5. 确认新的默认存储类：

```
kubectl get storageclass
```

添加自我管理集群

将公共自我管理集群添加到Astra Control Service

设置环境后，您可以创建 Kubernetes 集群，然后将其添加到 Astra Control Service 。

自我管理集群是指直接配置和管理的集群。Astra Control Service支持在公共云环境中运行的自我管理集群。您可以通过上传将自我管理集群添加到Astra Control Service kubeconfig.yaml 文件您需要确保集群满足此处所述的要求。

支持的Kubnetes分发版

您可以使用Astra Control Service管理以下类型的公共自我管理集群：

Kubbernetes分发	支持的版本
Kubnetes (上游)	1.27至1.29
Rancher Kubernetes Engine （RKE）	RKE 1：版本1.24.17、1.25.13、1.26.8、带RANcher Manager 2.7.9 RKE 2：版本1.23.16和1.24.13、带Randcher Manager 2.6.13 RKE 2：版本1.24.17、1.25.14、1.26.9、带RANcher Manager 2.7.9
Red Hat OpenShift 容器平台	4.12至4.14

以下说明假定您已创建一个自行管理的集群。

- [将集群添加到Asta Control Service](#)
- [\[更改默认存储类\]](#)

将集群添加到Asta Control Service

登录到 Astra Control Service 后，第一步是开始管理集群。在将集群添加到Astra Control Service之前、您需要执行特定任务并确保集群满足特定要求。

自管理集群是指直接配置和管理的集群。Astra Control Service支持在公共云环境中运行的自管理集群。您的自行管理集群可以使用Astra控件配置程序与NetApp存储服务连接、也可以使用容器存储接口(CSI)驱动程序与Amazon Elastic Block Store (EBS)、Azure托管磁盘和Google持久磁盘连接。

Astra控制服务支持使用以下Kubernetes分发版的自管理集群：

- Red Hat OpenShift 容器平台
- Rancher Kubernetes引擎
- 上游Kubernetes

您的自管理集群需要满足以下要求：

- 集群必须可通过Internet访问。
- 如果您正在使用或计划使用已启用CSI驱动程序的存储、则必须在集群上安装相应的CSI驱动程序。有关使用CSI驱动程序集成存储的详细信息、请参阅存储服务文档。
- 您可以访问仅包含一个上下文元素的集群kubeconfigfile文件。请遵循 ["这些说明"](#) 生成kubeconfig文件。
- 如果要使用引用私有证书颁发机构(CA)的kubeconfigfile文件添加集群、请将以下行添加到 cluster kubeconfig"文件的部分。这样、Astra Control便可添加集群：

```
insecure-skip-tls-verify: true
```

- ***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。
- **Astra Control**配置程序要求：要管理集群、您应正确配置Astra Control配置程序(包括其Astra三项功能组件)。
 - 查看**Astra**三端环境要求：在安装或升级Astra Control配置程序之前、请查看 ["支持的前端、后端和主机配置"](#)。
 - 启用**Astra Control**配置程序功能：强烈建议您安装Astra Trident 23.10或更高版本并启用 ["Astra Control配置程序高级存储功能"](#)。在未来版本中、如果Astra Control配置程序未启用、则Astra Control将不支持Astra Trident。
 - 配置存储后端：必须至少有一个存储后端 ["已在Astra Trident中配置"](#) 在集群上。
 - 配置存储类：必须至少有一个存储类 ["已在Astra Trident中配置"](#) 在集群上。如果配置了默认存储类，请确保该存储类是具有默认标注的*Only"存储类。
 - 配置卷快照控制器并安装卷快照类：["安装卷快照控制器"](#) 以便可以在Astra Control中创建快照。 ["创建"](#) 至少一个 VolumeSnapshotClass 使用Astra三端功能。

步骤

1. 在信息板上，选择 * 管理 Kubernetes 集群 *。

按照提示添加集群。

2. 提供程序：选择*其他*选项卡以添加有关自行管理的集群的详细信息。

- a. 其他：通过上传提供有关自管理集群的详细信息 kubeconfig.yaml 文件或粘贴的内容 kubeconfig.yaml 文件。



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。

3. 凭据名称：提供要上传到Astra Control的自管理集群凭据的名称。默认情况下，凭据名称会自动填充为集群的名称。

4. 专用路由标识符：此字段仅适用于专用集群。

5. 选择 * 下一步 *。

6. (可选)存储：(可选)选择默认情况下希望部署到此集群中的Kubernetes应用程序使用的存储类。

- a. 要为集群选择新的默认存储类，请启用*Assign a new default storage class*复选框。
b. 从列表中选择新的默认存储类。



每个云提供商存储服务都会显示以下价格、性能和弹性信息：

- Cloud Volumes Service for Google Cloud：价格、性能和弹性信息
- Google Persistent Disk：没有价格、性能或弹性信息
- Azure NetApp Files：性能和弹性信息
- Azure受管磁盘：无可用的价格、性能或弹性信息
- Amazon Elastic Block Store：没有价格、性能或弹性信息
- 适用于NetApp ONTAP 的Amazon FSX：没有价格、性能或弹性信息
- NetApp Cloud Volumes ONTAP：没有价格、性能或弹性信息

每个存储类均可使用以下服务之一：

- ["适用于 Google Cloud 的 Cloud Volumes Service"](#)
- ["Google 持久磁盘"](#)
 - ["Azure NetApp Files"](#)
 - ["Azure 受管磁盘"](#)
 - ["Amazon Elastic Block Store"](#)
 - ["适用于 NetApp ONTAP 的 Amazon FSX"](#)
 - ["NetApp Cloud Volumes ONTAP"](#)

了解更多信息 ["Amazon Web Services集群的存储类"](#)。了解更多信息 ["AKS 集群的存储类"](#)。了解更多信息 ["GKE 集群的存储类"](#)。

c. 选择 * 下一步 *。

d. 审核和批准：审核配置详细信息。

e. 选择*Add*将集群添加到Astra Control Service。

更改默认存储类

您可以更改集群的默认存储类。

使用Astra Control更改默认存储类

您可以在Astra Control中更改集群的默认存储类。如果集群使用先前安装的存储后端服务、则可能无法使用此方法更改默认存储类(不能选择*设置为默认值*操作)。在这种情况下、您可以 [\[使用命令行更改默认存储类\]](#)。

步骤

1. 在 Astra 控制服务 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*。

使用命令行更改默认存储类

您可以使用Kubernetes命令更改集群的默认存储类。无论集群的配置如何、此方法都有效。

步骤

1. 登录到Kubernetes集群。
2. 列出集群中的存储类：

```
kubectl get storageclass
```

3. 从默认存储类中删除默认指定。将<SC_NAME> 替换为存储类的名称：

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":  
{"annotations":{"storageclass.kubernetes.io/is-default-  
class":"false"}}}'
```

4. 将其他存储类标记为默认值。将<SC_NAME> 替换为存储类的名称：

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":  
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

5. 确认新的默认存储类：

```
kubectl get storageclass
```

将私有自管理集群添加到Astra Control Service

设置环境后，您可以创建 Kubernetes 集群，然后将其添加到 Astra Control Service 。

自管理集群是指直接配置和管理的集群。Astra Control Service支持在公共云环境中运行的自管理集群。您可以通过上传将自管理集群添加到Astra Control Service kubeconfig.yaml 文件您需要确保集群满足此处所述的要求。

支持的Kubnetes分发版

您可以使用Astra Control Service管理以下类型的专用自管理集群：

Kubbernetes分发	支持的版本
Kubnetes (上游)	1.27至1.29
Rancher Kubernetes Engine （RKE）	RKE 1：版本1.24.17、1.25.13、1.26.8、带RANcher Manager 2.7.9 RKE 2：版本1.23.16和1.24.13、带Randcher Manager 2.6.13 RKE 2：版本1.24.17、1.25.14、1.26.9、带RANcher Manager 2.7.9
Red Hat OpenShift 容器平台	4.12至4.14

以下说明假定您已创建私有集群并准备好了远程访问该集群的安全方法。

要将专用集群添加到Astra Control Service、您需要执行以下任务：

1. [安装A作用 连接器](#)
2. [\[设置永久性存储\]](#)
3. [将专用自管理集群添加到Astra Control Service](#)

安装A作用 连接器

在添加专用集群之前、您需要在此集群上安装Astra Connector、以便Astra Control可以与其通信。请参见 ["为使用非Kubnetes本机工作流管理的专用集群安装以前版本的Astra Connector"](#) 有关说明，请参见。

设置永久性存储

为集群配置永久性存储。有关配置永久性存储的详细信息、请参见入门文档：

- ["使用 Azure NetApp Files 设置 Microsoft Azure"](#)
- ["使用 Azure 受管磁盘设置 Microsoft Azure"](#)
- ["设置Amazon Web Services"](#)
- ["设置 Google Cloud"](#)

将专用自管理集群添加到Astra Control Service

现在、您可以将专用集群添加到Astra Control Service。

自管理集群是指直接配置和管理的集群。Astra Control Service支持在公共云环境中运行的自管理集群。您的自行管理集群可以使用Astra控件配置程序与NetApp存储服务连接、也可以使用容器存储接口(CSI)驱动程序与Amazon Elastic Block Store (EBS)、Azure托管磁盘和Google持久磁盘连接。

Astra控制服务支持使用以下Kubernetes分发版的自管理集群：

- Red Hat OpenShift 容器平台
- Rancher Kubernetes引擎
- 上游Kubernetes

您的自管理集群需要满足以下要求：

- 集群必须可通过Internet访问。
- 如果您正在使用或计划使用已启用CSI驱动程序的存储、则必须在集群上安装相应的CSI驱动程序。有关使用CSI驱动程序集成存储的详细信息、请参阅存储服务文档。
- 您可以访问仅包含一个上下文元素的集群kubeconfigfile文件。请遵循 ["这些说明"](#) 生成kubeconfig文件。
- 如果要使用引用私有证书颁发机构(CA)的kubeconfigfile文件添加集群、请将以下行添加到 cluster kubeconfig"文件的部分。这样、Astra Control便可添加集群：

```
insecure-skip-tls-verify: true
```

- ***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。
- **Astra Control**配置程序要求：要管理集群、您应正确配置Astra Control配置程序(包括其Astra三项功能组件)。
 - 查看**Astra**三端环境要求：在安装或升级Astra Control配置程序之前、请查看 ["支持的前端、后端和主机配置"](#)。
 - 启用**Astra Control**配置程序功能：强烈建议您安装Astra Trident 23.10或更高版本并启用 ["Astra Control配置程序高级存储功能"](#)。在未来版本中、如果Astra Control配置程序未启用、则Astra Control将不支持Astra Trident。
 - 配置存储后端：必须至少有一个存储后端 ["已在Astra Trident中配置"](#) 在集群上。
 - 配置存储类：必须至少有一个存储类 ["已在Astra Trident中配置"](#) 在集群上。如果配置了默认存储类，请确保该存储类是具有默认标注的*Only"存储类。
 - 配置卷快照控制器并安装卷快照类：["安装卷快照控制器"](#) 以便可以在Astra Control中创建快照。 ["创建"](#) 至少一个 VolumeSnapshotClass 使用Astra三端功能。

步骤

1. 在信息板上，选择 * 管理 Kubernetes 集群 *。

按照提示添加集群。

2. 提供程序：选择*其他*选项卡以添加有关自行管理的集群的详细信息。
3. 其他：通过上传提供有关自管理集群的详细信息 kubeconfig.yaml 文件或粘贴的内容 kubeconfig.yaml 文件。



创建自己的 kubeconfig file中、您只能定义*一*上下文元素。请参见 ["这些说明"](#) 有关创建的信息 kubeconfig 文件。

4. 凭据名称：提供要上传到Astra Control的自管理集群凭据的名称。默认情况下，凭据名称会自动填充为集群的名称。
5. 专用路由标识符：输入专用路由标识符，您可以从Astra Connector获取该标识符。如果您通过查询Astra Connector `kubectl get astraconnector -n astra-connector` 命令中、专用路由标识符称为 `ASTRACONNECTORID`。



专用路由标识符是与Astra Connector关联的名称、Astra可通过Astra管理专用Kubernetes集群。在这种情况下、专用集群是指不向Internet公开其API服务器的Kubernetes集群。

6. 选择 * 下一步 *。
7. (可选)存储：(可选)选择默认情况下希望部署到此集群中的Kubernetes应用程序使用的存储类。
 - a. 要为集群选择新的默认存储类，请启用*Assign a new default storage class*复选框。
 - b. 从列表中选择新的默认存储类。



每个云提供商存储服务都会显示以下价格、性能和弹性信息：

- Cloud Volumes Service for Google Cloud：价格、性能和弹性信息
- Google Persistent Disk：没有价格、性能或弹性信息
- Azure NetApp Files：性能和弹性信息
- Azure受管磁盘：无可用的价格、性能或弹性信息
- Amazon Elastic Block Store：没有价格、性能或弹性信息
- 适用于NetApp ONTAP 的Amazon FSX：没有价格、性能或弹性信息
- NetApp Cloud Volumes ONTAP：没有价格、性能或弹性信息

每个存储类均可使用以下服务之一：

- ["适用于 Google Cloud 的 Cloud Volumes Service"](#)
- ["Google 持久磁盘"](#)
- ["Azure NetApp Files"](#)
- ["Azure 受管磁盘"](#)
- ["Amazon Elastic Block Store"](#)
- ["适用于 NetApp ONTAP 的 Amazon FSX"](#)
- ["NetApp Cloud Volumes ONTAP"](#)

了解更多信息 ["Amazon Web Services集群的存储类"](#)。了解更多信息 ["AKS 集群的存储类"](#)。了解更

多信息 ["GKE 集群的存储类"](#)。

- c. 选择 * 下一步 *。
- d. 审核和批准：审核配置详细信息。
- e. 选择*Add*将集群添加到Astra Control Service。

更改默认存储类

您可以更改集群的默认存储类。

使用Astra Control更改默认存储类

您可以在Astra Control中更改集群的默认存储类。如果集群使用先前安装的存储后端服务、则可能无法使用此方法更改默认存储类(不能选择*设置为默认值*操作)。在这种情况下、您可以 [\[使用命令行更改默认存储类\]](#)。

步骤

1. 在 Astra 控制服务 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*。

使用命令行更改默认存储类

您可以使用Kubernetes命令更改集群的默认存储类。无论集群的配置如何、此方法都有效。

步骤

1. 登录到Kubernetes集群。
2. 列出集群中的存储类：

```
kubectl get storageclass
```

3. 从默认存储类中删除默认指定。将<SC_NAME> 替换为存储类的名称：

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":  
{"annotations":{"storageclass.kubernetes.io/is-default-  
class":"false"}}}'
```

4. 将其他存储类标记为默认值。将<SC_NAME> 替换为存储类的名称：

```
kubectl patch storageclass <SC_NAME> -p '{"metadata":  
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

5. 确认新的默认存储类：

```
kubectl get storageclass
```

检查Asta三端安装版本

要添加使用Asta Control置备程序或Asta三端对存储服务使用的自管理集群、请确保已安装的Asta三端对等版本为23.10或最新版本。

步骤

1. 确定您正在运行的Astra三项目标版本：

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

如果安装了Astra Trident、则会显示类似于以下内容的输出：

NAME	VERSION
trident	24.02.0

如果未安装Astra Trident、您将看到类似于以下内容的输出：

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

2. 执行以下操作之一：

- 如果您运行的是Asta三端凹凸版23.01或更早版本、请使用这些版本 ["说明"](#) 在升级到Asta Control配置程序之前、升级到Asta三端到最新版本。您可以 ["执行直接升级"](#) 如果您的Asta三端存储在版本24.02的四个版本的窗口中、则将Astra Control配置程序更新为24.02。例如、您可以直接从Asta三端23.04升级到Asta Control配置程序24.02。
- 如果您运行的是Astra Trident 23.10或更高版本、请验证Astra Control配置程序是否已启用 ["enabled"](#)。Asta Control配置程序不能用于23.10之前的Asta Control Center版本。 ["升级Astra Control配置程序"](#) 以便它与您要升级的Astra Control Center版本相同、以访问最新功能。

3. 确保Pod正在运行：

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

4. 检查存储类是否正在使用受支持的Astra 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kubefconfig文件

您可以使用kubefconfig"文件将集群添加到Astra Control Service。根据要添加的集群类型、您可能需要使用特定步骤为集群手动创建kubefconfigfile文件。

- [为Amazon EKS集群创建kubefconfig.文件](#)
- [为AWS \(ROSA\)集群上的Red Hat OpenShift Service创建一个kubefconfigfile文件](#)
- [\[为其他类型的集群创建kubefconfig.文件\]](#)

为Amazon EKS集群创建kubefconfig.文件

按照以下说明为Amazon EKS集群创建kubefconfigfile文件和永久令牌密钥。EKS中托管的集群需要永久令牌密钥。

步骤

1. 按照亚马逊文档中的说明生成kubefconfig:

["为Amazon EKS集群创建或更新kubefconconfig文件"](#)

2. 按如下所示创建服务帐户：

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

根据需要调整服务帐户名称。命名空间 `kube-system` 这些步骤需要。如果您在此处更改了服务帐户名称、则应在以下步骤中应用相同的更改。

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

+

```
apiVersion: v1
kind: ServiceAccount
metadata:
  name: astra-admin-account
  namespace: kube-system
```

3. 应用服务帐户：

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

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

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

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: astra-admin-binding
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: astra-admin-account
  namespace: kube-system
```

5. 应用集群角色绑定：

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

6. 创建名为的服务帐户令牌机密文件 astracontrol-secret.yaml。

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

```
apiVersion: v1
kind: Secret
metadata:
  annotations:
    kubernetes.io/service-account.name: astra-admin-account
  name: astra-admin-account
  namespace: kube-system
type: kubernetes.io/service-account-token
```

7. 应用令牌密钥:

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

8. 检索令牌密钥:

```
kubectl get secret astra-admin-account -n kube-system -o  
jsonpath='{.data.token}' | base64 -d
```

9. 更换 user 部分的AWS EKS kubeconfigconfig文件以及令牌、如以下示例所示：

```
user:
  token: k8s-aws-
  v1.aHR0cHM6Ly9zdHMudXMtd2VzdC0yLmFtYXpvcnF3cy5jb20vP0FjdGlrbj1HZXRdYWxsZ
  XJJZGVudGl0eSZWZXXzaW9uPTIwMTetMDYtMTUmWC1BbXotQWxnbn3JpdGhtPUFXUzQtSE1BQ
  y1TSEEyNTYmWC1BbXotQ3JlZGVudGlhbD1BS01BM1JEWddkU0haWU9LSEQ2SyUyRjIwMjMwN
  DAzJTJGdXMtd2VzdC0yJTJGc3RzJTJGYXdzNF9yZXF1ZXN0JlgtQW16LURhdGU9MjAyMzA0M
  DNUMjA0MzQwWiZYLUFteilFeHBpcmVzPTYwJlgtQW16LVNpZ25lZEhlYWRLcnM9aG9zdCUzQ
  ngtazhzLWF3cy1pZCZYLUFteilTaWduYXRlcuU9YjU4ZWM0NzdiM2NkZGYxNGRhNzU4MGI2Z
  WQ2zyZ2NzI2YWIwM2UyNThjMjRhNTJjNmVhNjc4MTRlNjJkOTg2Mg
```

为AWS (ROSA)集群上的Red Hat OpenShift Service创建一个kubeconfigfile文件

按照以下说明为Red Hat OpenShift Service on AWS (ROSA)集群创建kubeconfigTM文件。

步骤

1. 登录到ROSA集群。
2. 创建服务帐户：

```
oc create sa astracontrol-service-account
```

3. 添加集群角色：

```
oc adm policy add-cluster-role-to-user cluster-admin -z astracontrol-  
service-account
```

4. 使用以下示例、创建一个服务帐户机密配置文件：

```
<strong>secret-astra-sa.yaml</strong>
```

```
apiVersion: v1  
kind: Secret  
metadata:  
  name: secret-astracontrol-service-account  
  annotations:  
    kubernetes.io/service-account.name: "astracontrol-service-account"  
type: kubernetes.io/service-account-token
```

5. 创建密钥：

```
oc create -f secret-astra-sa.yaml
```

6. 编辑您创建的服务帐户、并将Astra Control服务帐户机密名称添加到中 secrets 部分。

```
oc edit sa astracontrol-service-account
```

```
apiVersion: v1  
imagePullSecrets:  
- name: astracontrol-service-account-dockercfg-dvfcd  
kind: ServiceAccount  
metadata:  
  creationTimestamp: "2023-08-04T04:18:30Z"  
  name: astracontrol-service-account  
  namespace: default  
  resourceVersion: "169770"  
  uid: 965fa151-923f-4fbd-9289-30cad15998ac  
secrets:  
- name: astracontrol-service-account-dockercfg-dvfcd  
- name: secret-astracontrol-service-account ####ADD THIS ONLY####
```

7. 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-dvfcd"},
{ "name": "secret-astracontrol-service-account"}
]
```

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-dvfcd 将为0、并为创建索引 secret-astracontrol-service-account 将为1。在输出中、记下服务帐户密钥的索引编号。在下一步中、您将需要此索引编号。

8. 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
```

9. (可选)将kubeconfig重命名为集群的有意义名称。


```
mv kubeconfig-sa YOUR_CLUSTER_NAME_kubeconfig
```

为其他类型的集群创建**kubeconfig**文件

按照以下说明为原生集群、上游Kubernetes集群和Red Hat OpenShift集群创建有限或扩展的角色kubeconfig文件。

对于使用kubeconfig"管理的集群、您可以选择为Astra Control Service创建有限权限或扩展权限管理员角色。

如果您适用场景的环境发生以下任一情况、则此操作步骤可帮助您创建一个单独的kubeconfig:

- 您希望限制Astra Control对其管理的集群的权限
- 您使用多个环境、并且不能使用在安装期间配置的默认Astra Control kubeconfig,否则在您的环境中使用单一环境的有限角色将不起作用

开始之前

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:

- 答 "支持的版本" 已安装kubect.
- 对要使用Astra Control Service添加和管理的集群的kubect访问权限



对于此操作步骤、您不需要对运行Astra控制服务的集群进行kubect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`。

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

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

b. 应用服务帐户:

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

2. 创建以下具有足够权限的集群角色之一、以使集群由Astra Control管理:

集群角色受限

此角色包含由Asta Control管理集群所需的最低权限：

- a. 创建 ClusterRole 文件、例如、astra-admin-account.yaml。

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

```
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
- replicaset
- replicationcontrollers
- replicationcontrollers/scale
- rolebindings
- roles
- secrets
- serviceaccounts
- services
- statefulsets
- tridentmirrorrelationships
- tridentnapshotinfos
- 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
```

b. (仅适用于OpenShift集群)在末尾附加以下内容 `astra-admin-account.yaml` 文件:

```
# OpenShift security
- apiGroups:
  - security.openshift.io
  resources:
  - securitycontextconstraints
  verbs:
  - use
  - update
```

c. 应用集群角色:

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

已扩展集群角色

此角色包含要由Asta Control管理的集群的扩展权限。如果您使用多个环境，并且无法使用在安装期间配置的默认Asta Control kubeconfig,则可以使用此角色，否则在您的环境中，只使用一个环境的有限角色将不起作用:



以下内容 ClusterRole 步骤是一个常规Kubernetes示例。有关特定于您的环境的说明、请参见Kubernetes分发版的文档。

a. 创建 ClusterRole 文件、例如、 `astra-admin-account.yaml`。

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

```

apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: astra-admin-account
rules:
- apiGroups:
  - '*'
  resources:
  - '*'
  verbs:
  - '*'
- nonResourceURLs:
  - '*'
  verbs:
  - '*'

```

b. 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。

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

```

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

```

b. 应用集群角色绑定：

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

4. 创建并应用令牌密钥:

- a. 创建名为的令牌机密文件 `secret-astracontrol-service-account.yaml`。

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

```
apiVersion: v1
kind: Secret
metadata:
  name: secret-astracontrol-service-account
  namespace: default
  annotations:
    kubernetes.io/service-account.name: "astracontrol-service-
account"
type: kubernetes.io/service-account-token
```

- b. 应用令牌密钥:

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

5. 通过将令牌密钥名称添加到、将其添加到服务帐户 `secrets` 数组(以下示例中的最后一行):

```
kubectl edit sa astracontrol-service-account
```

```

apiVersion: v1
imagePullSecrets:
- name: astracontrol-service-account-dockercfg-48xhx
kind: ServiceAccount
metadata:
  annotations:
    kubectl.kubernetes.io/last-applied-configuration: |

{"apiVersion":"v1","kind":"ServiceAccount","metadata":{"annotations":{},"name":"astracontrol-service-account","namespace":"default"},"creationTimestamp":"2023-06-14T15:25:45Z","name":"astracontrol-service-account","namespace":"default","resourceVersion":"2767069","uid":"2ce068c4-810e-4a96-ada3-49cbf9ec3f89"}
secrets:
- name: astracontrol-service-account-dockercfg-48xhx
<strong>- name: secret-astracontrol-service-account</strong>

```

6. 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-48xhx"},
{ "name": "secret-astracontrol-service-account"}
]

```

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-48xhx 将为0、并为创建索引 secret-astracontrol-service-account 将为1。在输出中、记下服务帐户密钥的索引编号。在下一步中、您将需要此索引编号。

7. 按如下所示生成 kubeconfig：

- a. 创建 create-kubeconfig.sh 文件
- b. 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=astraccontrol-service-account
NAMESPACE=default
NEW_CONTEXT=astraccontrol
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
```

c. 获取用于将其应用于 Kubernetes 集群的命令。

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

8. (可选)将kubeconfig重命名为集群的有意义名称。

```
mv kubeconfig-sa YOUR_CLUSTER_NAME_kubeconfig
```

版权信息

版权所有 © 2024 NetApp, Inc.。保留所有权利。中国印刷。未经版权所有者事先书面许可，本文档中受版权保护的任何部分不得以任何形式或通过任何手段（图片、电子或机械方式，包括影印、录音、录像或存储在电子检索系统中）进行复制。

从受版权保护的 NetApp 资料派生的软件受以下许可和免责声明的约束：

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

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

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

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

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

商标信息

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