Astra Control Center 24.02文档

Astra Control Center

NetApp
April 25, 2024

目录

Astra Control Center 24.02文档

发行说明

我们很高兴地宣布发布最新版本的Astra控制中心。

- "此版本的 Astra 控制中心包含哪些内容"
- "已知问题"
- "已知限制"

通过成为发送有关文档的反馈 "GitHub 贡献者" 或发送电子邮件至 doccomments@netapp.com 。

此版本的 Astra 控制中心中的新增功能

我们很高兴地宣布发布最新版本的Astra控制中心。

2024年3月15日(24.02.0)

新增功能和支持

- *在没有私有注册表的情况下部署Astra Control Center *：您不再需要将Astra Control Center映像推送到私有注册表或将其用作Astra Control环境的一部分。
- 小错误修复

已知问题和限制

- "此版本的已知问题"
- "此版本的已知限制"

(技术预览)声明性Kubarnetes工作流

此Astra Control Center版本包含声明性Kubarnetes功能、可用于从本机Kubarnetes自定义资源(CR)执行数据管理。

安装后 "Asta连接器" 在要管理的集群上、您将能够在UI或CR中执行以下基于CR的集群操作：

- "使用自定义资源定义应用程序"
- "定义存储分段"
- "保护整个集群"
- "备份应用程序"
- "创建快照"
- "为快照或备份创建计划"
- "从快照或备份还原应用程序"

2023年11月7日(23.10.0)

新增功能和支持

- 使用由**ONTAP**驱动程序提供支持的存储后端为应用程序提供备份和还原功能：为启用备份和还原操作

`ontap-nas-economy` 和一些 "简单的步骤"。

- 不可改变的备份：Astra Control现在支持 "不可更改的只读备份" 作为抵御恶意软件和其他威胁的额外安全层。
- * Astra Control配置程序简介*

  在23.10版中、Astra Control引入了一个新的软件组件、称为Astra Control配置程序、该组件可供所有获得许可的Astra Control用户使用。Asta Control配置程序提供对Asta三元数据以外的一组高级管理和存储配置功能的访问。所有Astra Control客户均可免费使用这些功能。

  - 开始使用**Astra Control**配置程序
    您可以 "启用Asta Control配置程序" 如果您已安装并配置环境以使用Asta Trident 23.10。
  - **Astra Control**配置程序功能

    Astra Control配置程序23.10版提供了以下功能：

    - 通过**Kerberos 5**加密增强存储后端安全性：您可以通过提高存储安全性 "正在启用加密" 托管集群和存储后端之间的流量。Astra Control配置程序支持通过NFSv4.1连接从Red Hat OpenShift集群到Azure NetApp Files和内部ONTAP卷进行Kerberos 5加密
    - 使用快照恢复数据：Astra Control配置程序可使用从快照快速原位还原卷 `TridentActionSnapshotRestore` (TSR) CR。
    - **SnapMirror**增强功能：在Astra Control无法直接连接到ONTAP集群或访问ONTAP凭据的环境中使用应用程序复制功能。通过此功能、您可以使用复制功能、而无需在Astra Control中管理存储后端或其凭据。
    - 使用为应用程序提供备份和还原功能 **`ontap-nas-economy`** 驱动程序支持的存储后端：如上所述以上。
- 支持管理使用**NVMe/TCP**存储的应用程序
  Astra Control现在可以管理由使用NVMe/TCP连接的永久性卷提供支持的应用程序。
- 默认情况下，执行挂钩处于关闭状态：从此版本开始，执行挂钩功能可以是 "enabled" 或禁用以提高安全性(默认情况下处于禁用状态)。如果尚未创建用于Astra Control的执行挂钩、则需要 "启用执行挂钩功能" 开始创建挂钩。如果您在此版本之前创建了执行挂钩、则执行挂钩功能将保持启用状态、您可以像往常一样使用挂钩。

已知问题和限制

- "此版本的已知问题"
- "此版本的已知限制"

2023年7月31日(23.07.0)

新增功能和支持

- "支持在延伸型配置中使用NetApp MetroCluster作为存储后端"
- "支持使用Longhorn作为存储后端"
- "现在、可以从同一个Kubnetes集群在ONTAP后端之间复制应用程序"
- "Astra Control Center现在支持使用"userPrincipalName"作为远程(LDAP)用户的备用登录属性"
- "在使用Asta Control Center执行复制故障转移后、可以运行新的执行挂钩类型"故障转移后""

- 克隆工作流现在仅支持实时克隆(托管应用程序的当前状态)。要从快照或备份克隆、请使用 "还原工作流"。

已知问题和限制

- "此版本的已知问题"
- "此版本的已知限制"

2023年5月18日(23.04.2)

此修补程序版本(23.04.2)用于Astra Control Center (23.04.0)提供对的支持 "Kubernetes CSI外部快照程序v6.1.0" 并修复了以下问题：

- 使用执行挂钩时的原位应用程序还原错误
- 存储分段服务存在连接问题

2023年4月25日(23.04.0)

新增功能和支持

- "默认情况下、新Astra Control Center安装启用了90天评估许可证"
- "增强的执行挂钩功能以及其他筛选选项"
- "现在、可以使用Asta Control Center在复制故障转移后运行执行挂钩"
- "支持将卷从"ONTAP - NAS经济型存储"类迁移到"ONTAP - NAS "存储类"
- "支持在还原操作期间包括或排除应用程序资源"
- "支持管理纯数据应用程序"

已知问题和限制

- "此版本的已知问题"
- "此版本的已知限制"

2022年11月22日(22.11.0)

新增功能和支持

- "支持跨多个命名空间的应用程序"
- "支持在应用程序定义中包括集群资源"
- "通过基于角色的访问控制(Role-Based Access Control、RBAC)集成增强了LDAP身份验证功能"
- "增加了对Kubernetes 1.25和Pod安全准入(PSA)的支持"
- "增强了备份、还原和克隆操作的进度报告功能"

已知问题和限制

- "此版本的已知问题"
- "此版本的已知限制"

2022年9月8日(22.08.1)

适用于Astra控制中心(22.08.0)的此修补程序版本(22.08.1)修复了使用NetApp SnapMirror复制应用程序时出现的小错误。

2022年8月10日(22.08.0)

新增功能和支持

- "使用NetApp SnapMirror技术复制应用程序"
- "改进了应用程序管理工作流"
- "增强的自行执行挂钩功能"

> 此版本已删除NetApp为特定应用程序提供的默认快照前和快照后执行挂钩。如果您升级到此版本、但没有为快照提供自己的执行挂钩、则Astra Control将仅创建崩溃状态一致的快照。请访问 "NetApp Verda" GitHub存储库、用于创建示例执行钩脚本、您可以根据环境进行修改。

- "支持VMware Tanzu Kubernetes Grid Integrated Edition (TKGI)"
- "支持Google Anthos"
- "LDAP配置(通过Astra Control API)"

已知问题和限制

- "此版本的已知问题"
- "此版本的已知限制"

2022 年 4 月 26 日（ 22.04.0 ）

新增功能和支持

- "命名空间基于角色的访问控制（ RBAC ）"
- "支持 Cloud Volumes ONTAP"
- "为 Astra 控制中心启用通用传入"
- "从 Astra Control 中删除存储分段"
- "支持 VMware Tanzu 产品组合"

已知问题和限制

- "此版本的已知问题"
- "此版本的已知限制"

2021 年 12 月 14 日（ 21.12 ）

新增功能和支持

- "应用程序还原"
- "执行挂钩"
- "支持使用命名空间范围的运算符部署的应用程序"

- "对上游 Kubernetes 和 Rancher 的其他支持"
- "Astra 控制中心升级"
- "用于安装的 Red Hat OperatorHub 选项"

已解决的问题

- "此版本已解决的问题"

已知问题和限制

- "此版本的已知问题"
- "此版本的已知限制"

2021 年 8 月 5 日（21.08）

初始版本的 Astra 控制中心。

- "它是什么"
- "了解架构和组件"
- "入门所需的资源"
- "安装" 和 "设置"
- "管理" 和 "保护" 应用程序
- "管理存储分段" 和 "存储后端"
- "管理帐户"
- "利用 API 实现自动化"

了解更多信息

- "此版本的已知问题"
- "此版本的已知限制"
- "早期版本的 Astra 控制中心文档"

已知问题

已知问题可确定可能妨碍您成功使用此版本产品的问题。

以下已知问题会影响当前版本：

- [如果在管理集群后添加了volumesnapshotclass、则应用程序备份和快照将失败]
- 如果kubeconfig"文件包含多个上下文、则使用Asta Control Center管理集群将失败
- 当 Astra Trident 脱机时，应用程序数据管理操作失败，并显示内部服务错误（500）
- 对ONTAP NAS经济型存储类的原位还原操作失败
- 使用Kerberos传输中加密时从备份还原可能会失败

- [对于保留策略已过期的存储分段、删除后备份数据仍会保留在存储分段中]

如果在管理集群后添加了volumesnapshotclass、则应用程序备份和快照将失败

在这种情况下、备份和快照失败、并显示`UI 500错误`。作为临时解决策 、刷新应用程序列表。

如果kubeconfig"文件包含多个上下文、则使用Asta Control Center管理集群将失败

不能将 kubeconfig 与多个集群和上下文结合使用。请参见 "知识库文章" 有关详细信息 …

当 Astra Trident 脱机时，应用程序数据管理操作失败，并显示内部服务错误（ 500 ）

如果应用程序集群上的 Astra Trident 脱机（并恢复联机），并且在尝试应用程序数据管理时遇到 500 个内部服务错误，请重新启动应用程序集群中的所有 Kubernetes 节点以还原功能。

对ONTAP NAS经济型存储类的原位还原操作失败

如果您对应用程序执行原位还原(将应用程序还原到其原始命名空间)、并且应用程序的存储类使用 `ontap-nas-economy` 驱动程序、如果未隐藏快照目录、则还原操作可能会失败。在原位还原之前、请按照中的说明进行操作 "为ONTAP NAS经济型操作启用备份和还原" 以隐藏快照目录。

使用Kerberos传输中加密时从备份还原可能会失败

将应用程序从备份还原到使用Kerberos传输中加密的存储后端时、还原操作可能会失败。此问题描述不会影响从快照还原或使用NetApp SnapMirror复制应用程序数据。

> 在对NFSv4卷使用Kerberos传输中加密时、请确保NFSv4卷使用正确的设置。请参阅的NetApp NFSv4域配置一节(第13页) "《NetApp NFSv4增强功能和最佳实践指南》"。

对于保留策略已过期的存储分段、删除后备份数据仍会保留在存储分段中

如果在存储分段的保留策略过期后删除应用程序的不可更改备份、则备份将从Astra Control中删除、而不是从存储分段中删除。此问题描述将在即将发布的版本中予以修复。

了解更多信息

- "已知限制"

已知限制

已知限制确定了本产品版本不支持的平台、设备或功能、或者这些平台、设备或功能无法与产品正确交互操作。仔细审查这些限制。

集群管理限制

- 同一集群不能由两个 Astra Control Center 实例管理
- Astra 控制中心无法管理两个命名相同的集群

基于角色的访问控制（**Role-Based Access Control**， **RBAC**）限制

- 具有命名空间 RBAC 限制的用户可以添加和取消管理集群
- [具有命名空间约束的成员无法访问克隆或还原的应用程序，直到管理员将命名空间添加到此限制中为止]
- [对于非连接器集群上的资源、可以忽略限制性角色约束]

应用程序管理限制

- [一个命名空间中的多个应用程序无法一起还原到另一个命名空间]
- Astra Control不支持每个命名空间使用多个存储类的应用程序
- Astra Control不会自动为云实例分配默认分段
- [使用按参考传递操作符安装的应用程序克隆可能会失败]
- [不支持对使用证书管理器的应用程序执行原位还原操作]
- 不支持已部署的应用程序，这些应用程序已启用 olm ，并且已部署集群范围
- 不支持使用 Helm 2 部署的应用程序
- 使用某些Snapshot控制器版本的Kubernetes 1.25或更高版本集群的快照可能会失败
- 删除 Astra Control Center 实例期间，备份和快照可能不会保留

一般限制

- LDAP用户和组限制
- Astra 控制中心中的 S3 存储分段不会报告可用容量
- Astra 控制中心不会验证您为代理服务器输入的详细信息
- 与 Postgres Pod 的现有连接导致故障
- <<"Activity"页面最多可显示100000个事件>>
- SnapMirror不支持将基于TCP的NVMe用于存储后端的应用程序

同一集群不能由两个 **Astra Control Center** 实例管理

如果要管理另一个 Astra Control Center 实例上的集群，应首先进行管理 "取消管理集群" 在另一个实例上管理之前，先从所管理的实例进行管理。从管理中删除集群后，执行以下命令以验证此集群是否未受管理：

```
oc get pods n -netapp-monitoring
```

此命名空间中不应运行任何 Pod ，或者此命名空间不应存在。如果其中任一项为 true ，则集群不受管理。

Astra 控制中心无法管理两个命名相同的集群

如果您尝试添加与已存在的集群同名的集群，则此操作将失败。如果未更改 Kubernetes 配置文件中的集群默认名称，则此问题描述最常发生在标准 Kubernetes 环境中。

作为临时解决策，请执行以下操作：

1. 编辑 `kubeadm-config` 配置映射：

```
kubectl edit configmaps -n kube-system kubeadm-config
```

2. 将 `clustername` 字段值从 `Kubernetes` （Kubernetes 默认名称）更改为唯一的自定义名称。
3. 编辑 kubeconfig （`.Kube/config`）。
4. 将集群名称从 `Kubernetes` 更新为唯一的自定义名称（在以下示例中使用`xyz-cluster`）。在 `clusters` 和 `Context` 部分进行更新，如以下示例所示：

```
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data:
ExAmPLERb2tCcjZ5K3E2Njk4eQotLExAMpLEORCBDRVJUSUZJQ0FURS0txxxxXX==
    server: https://x.x.x.x:6443
  name: xyz-cluster
contexts:
- context:
    cluster: xyz-cluster
    namespace: default
    user: kubernetes-admin
  name: kubernetes-admin@kubernetes
current-context: kubernetes-admin@kubernetes
```

具有命名空间 RBAC 限制的用户可以添加和取消管理集群

不应允许具有命名空间 RBAC 限制的用户添加或取消管理集群。由于当前的限制， Astra 不会阻止此类用户取消管理集群。

具有命名空间约束的成员无法访问克隆或还原的应用程序，直到管理员将命名空间添加到此限制中为止

任意 `member` 使用命名空间名称/ID限制RBAC的用户可以将应用程序克隆或还原到同一集群上的新命名空间或其组织帐户中的任何其他集群。但是，同一用户无法访问新命名空间中的克隆或还原应用程序。克隆或还原操作创建新命名空间后、帐户管理员/所有者可以编辑 `member` 受影响用户的用户帐户和更新角色约束、以授予对新命名空间的访问权限。

对于非连接器集群上的资源、可以忽略限制性角色约束

- 如果要访问的资源属于安装了最新**Astra Connector**的集群：通过LDAP组成员资格为用户分配多个角色时，这些角色的约束条件将合并在一起。例如、如果具有本地查看器角色的用户加入绑定到成员角色的三个组、则该用户现在可以以查看器角色访问原始资源、并以成员角色访问通过组成员资格获得的资源。
- 如果要访问的资源属于未安装**Astra Connector**的集群：通过LDAP组成员资格为用户分配多个角色时，只有最宽松角色的限制才会生效。

一个命名空间中的多个应用程序无法一起还原到另一个命名空间

如果您在一个命名空间中管理多个应用程序(通过在Astra Control中创建多个应用程序定义)、则无法将所有应用程序还原到另一个命名空间。您需要将每个应用程序还原到其自己单独的命名空间。

Astra Control不支持每个命名空间使用多个存储类的应用程序

Astra Control支持每个命名空间使用一个存储类的应用程序。将应用程序添加到命名空间时、请确保该应用程序与命名空间中的其他应用程序具有相同的存储类。

Astra Control不会自动为云实例分配默认分段

Astra Control不会自动为任何云实例分配默认分段。您需要手动设置云实例的默认存储分段。如果未设置默认分段、您将无法在两个集群之间执行应用程序克隆操作。

使用按参考传递操作符安装的应用程序克隆可能会失败

Astra Control 支持使用命名空间范围的运算符安装的应用程序。这些操作员通常采用 " 按价值传递 " 架构，而不是 " 按参考传递 " 架构。以下是一些遵循这些模式的操作员应用程序：

- "Apache K8ssandra"

> 对于 K8ssandra ，支持原位还原操作。要对新命名空间或集群执行还原操作，需要关闭应用程序的原始实例。这是为了确保传输的对等组信息不会导致跨实例通信。不支持克隆应用程序。

- "Jenkins CI"
- "Percona XtraDB 集群"

Astra Control可能无法克隆使用"按参考传递"架构设计的运算符(例如CockroachDB运算符)。在这些类型的克隆操作期间，克隆的操作员会尝试引用源操作员提供的 Kubernetes 机密，尽管在克隆过程中他们拥有自己的新机密。克隆操作可能会失败，因为 Astra Control 不知道源运算符中的 Kubernetes 密钥。

> 在克隆操作期间、需要IngressClass资源或webhooks才能正常运行的应用程序不能在目标集群上定义这些资源。

不支持对使用证书管理器的应用程序执行原位还原操作

此版本的 Astra 控制中心不支持使用证书管理器原位还原应用程序。支持将还原操作还原到其他命名空间和克隆操作。

不支持已部署的应用程序，这些应用程序已启用 olm ，并且已部署集群范围

Astra 控制中心不支持使用集群范围的操作员执行应用程序管理活动。

不支持使用 Helm 2 部署的应用程序

如果您使用 Helm 部署应用程序，则 Astra 控制中心需要 Helm 版本 3 。完全支持管理和克隆使用 Helm 3 部署的应用程序（或从 Helm 2 升级到 Helm 3 ）。有关详细信息，请参见 "Astra 控制中心要求"。

使用某些Snapshot控制器版本的Kubernetes 1.25或更高版本集群的快照可能会失败

如果在运行1.25或更高版本的Kubernetes集群上安装了v1beta1版本的快照控制器API、则该集群的快照可能会失败。

作为临时解决策 、在升级现有Kubernetes 1.25或更高版本的安装时、请执行以下操作：

1. 删除任何现有的Snapshot CRD和任何现有的Snapshot控制器。
2. "卸载 Astra Trident"。
3. "安装快照CRD和快照控制器"。
4. "安装最新版本的Astra Trdent"。
5. "创建卷快照类"。

删除 Astra Control Center 实例期间，备份和快照可能不会保留

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。

LDAP用户和组限制

Astra控制中心最多支持5、000个远程组和10、000个远程用户。

如果某个LDAP实体(用户或组)的DN包含一个RDN、并且该RDN带有尾随空格、则Astra Control不支持该实体。

Astra 控制中心中的 S3 存储分段不会报告可用容量

在备份或克隆由 Astra 控制中心管理的应用程序之前，请检查 ONTAP 或 StorageGRID 管理系统中的存储分段信息。

Astra 控制中心不会验证您为代理服务器输入的详细信息

请确保您的安全 "输入正确的值" 建立连接时。

与 Postgres Pod 的现有连接导致故障

在 Postgres Pod 上执行操作时，不应直接在 Pod 中连接以使用 psql 命令。Astra Control 需要使用 psql 访问权限来冻结和解冻数据库。如果已建立连接，则快照，备份或克隆将失败。

"Activity"页面最多可显示100000个事件

Astra Control Activity页面最多可显示100、000个事件。要查看所有记录的事件、请使用检索这些事件 "Astra Control API"。

SnapMirror不支持将基于TCP的NVMe用于存储后端的应用程序

对于使用基于TCP协议的NVMe的存储后端、Astra控制中心不支持NetApp SnapMirror复制。

了解更多信息

- "已知问题"

入门

了解Astra Control

Astra Control 是 Kubernetes 应用程序数据生命周期管理解决方案，可简化有状态应用程序的操作。轻松保护、备份、复制和迁移Kubernetes工作负载、并即时创建有效的应用程序克隆。

功能

Astra Control 为 Kubernetes 应用程序数据生命周期管理提供了关键功能：

- 自动管理永久性存储
- 创建应用程序感知型按需快照和备份
- 自动执行策略驱动的快照和备份操作
- 将应用程序和数据从一个 Kubernetes 集群迁移到另一个集群
- 使用NetApp SnapMirror技术(Astra Control Center)将应用程序复制到远程系统
- 将应用程序从暂存克隆到生产
- 直观显示应用程序运行状况和保护状态
- 使用Web UI或API实施备份和迁移工作流

部署模式

Astra Control 有两种部署模式：

- * Astra Control Service*：NetApp管理的服务、可为多个云提供商环境中的Kubernetes集群以及自管理Kubernetes集群提供应用程序感知型数据管理。
- * Astra Control Center* ：自管理软件，可为内部环境中运行的 Kubernetes 集群提供应用程序感知型数据管理。Astra控制中心还可以安装在多个云提供商环境中、并具有一个NetApp Cloud Volumes ONTAP 存储后端。

	Astra 控制服务	**Astra** 控制中心
如何提供?	作为 NetApp 提供的一项完全托管的云服务	作为可下载、安装和管理的软件
它托管在何处?	基于 NetApp 选择的公有云	在您自己的Kubernetes集群上
如何更新?	由 NetApp 管理	您可以管理任何更新

	Astra 控制服务	**Astra** 控制中心

支持哪些**Kubednetes**分发版?	• 云提供商
	  ◦ Amazon Web Services
	  ▪ Amazon Elelic Kubelnetes Service (EKS)
	  ◦ Google Cloud
	  ▪ Google Kubernetes Engine （GKEE）
	  ◦ Microsoft Azure
	  ▪ Azure Kubernetes Service （AKS）
	• 自管理集群
	  ◦ Kubnetes (上游)
	  ◦ Rancher Kubernetes Engine （RKE）
	  ◦ Red Hat OpenShift 容器平台
	• 内部集群
	  ◦ Red Hat OpenShift容器平台内部部署
		• 基于Azure堆栈HCI的Azure Kubnetes Service
		• Google Anthos
		• Kubnetes (上游)
		• Rancher Kubernetes Engine （RKE）
		• Red Hat OpenShift 容器平台

	Astra 控制服务	Astra 控制中心
支持哪些存储后端?	• 云提供商 ◦ Amazon Web Services ▪ Amazon EBS ▪ 适用于 NetApp ONTAP 的 Amazon FSX ▪ "Cloud Volumes ONTAP" ◦ Google Cloud ▪ Google 持久磁盘 ▪ NetApp Cloud Volumes Service ▪ "Cloud Volumes ONTAP" ◦ Microsoft Azure ▪ Azure受管磁盘 ▪ Azure NetApp Files ▪ "Cloud Volumes ONTAP" • 自管理集群 ◦ Amazon EBS ◦ Azure受管磁盘 ◦ Google 持久磁盘 ◦ "Cloud Volumes ONTAP" ◦ NetApp MetroCluster ◦ "Longhorn" • 内部集群 ◦ NetApp MetroCluster ◦ NetApp ONTAP AFF 和 FAS 系统 ◦ NetApp ONTAP Select ◦ "Cloud Volumes ONTAP" ◦ "Longhorn"	• NetApp ONTAP AFF 和 FAS 系统 • NetApp ONTAP Select • "Cloud Volumes ONTAP" • "Longhorn"

Astra 控制服务的工作原理

Astra Control Service 是一种由 NetApp 管理的云服务，它始终处于启用状态，并使用最新功能进行更新。它利用多个组件实现应用程序数据生命周期管理。

从较高的层面来看， Astra Control Service 的工作原理如下：

- 您可以通过设置云提供商并注册 Astra 帐户开始使用 Astra Control Service 。

    - 对于 GKE- 集群， Astra Control Service 使用 "适用于 Google Cloud 的 NetApp Cloud Volumes Service" 或 Google Persistent Disk 作为永久性卷的存储后端。
    - 对于 AKS 集群， Astra Control Service 使用 "Azure NetApp Files" 或Azure受管磁盘作为永久性卷的存储后端。
    - 对于Amazon EKS集群、Astra Control Service使用 "Amazon Elastic Block Store" 或 "适用于 NetApp ONTAP 的 Amazon FSX" 作为永久性卷的存储后端。
- 您可以将第一个 Kubernetes 计算添加到 Astra Control Service 中。然后， Astra 控制服务将执行以下操作：
    - 在云提供商帐户中创建一个对象存储，该帐户是备份副本的存储位置。

      在 Azure 中， Astra Control Service 还会为 Blob 容器创建资源组，存储帐户和密钥。
    - 在集群上创建新的管理员角色和 Kubernetes 服务帐户。
    - 使用此新管理员角色在集群上安装link./概念/architution#Astra-control-components [Astra Control置备程序^]、并创建一个或多个存储类。
    - 如果您使用NetApp云服务存储产品作为存储后端、Astra控制服务将使用Astra控制配置程序为应用程序配置永久性卷。如果您使用Amazon EBS或Azure托管磁盘作为存储后端、则需要安装特定于提供商的CSI驱动程序。中提供了安装说明 "设置Amazon Web Services" 和 "使用 Azure 受管磁盘设置 Microsoft Azure"。
- 此时，您可以向集群添加应用程序。将在新的默认存储类上配置永久性卷。
- 然后，您可以使用 Astra Control Service 管理这些应用程序，并开始创建快照，备份和克隆。

Astra Control的免费计划支持您管理帐户中多达10个命名空间。如果您要管理10个以上的计划、则需要通过从"免费计划"升级到"高级计划"来设置计费。

Astra 控制中心的工作原理

Astra 控制中心在您自己的私有云中本地运行。

Astra控制中心支持Kuburenet集群、其中Astra控制配置程序配置了存储类、并具有ONTAP存储后端。

Astra Control Center提供有限的(7天的指标)监控和遥测功能、还可通过开放式指标端点导出到Kubarnetes本机监控工具(如Prometheus和Grafana)。

Astra 控制中心完全集成到 AutoSupport 和 Active IQ 生态系统中，可为用户和 NetApp 支持提供故障排除和使用信息。

您可以使用90天嵌入式评估许可证试用Astra Control Center。在评估Astra Control Center时、您可以通过电子邮件和社区选项获得支持。此外，您还可以从产品支持信息板访问知识库文章和文档。

要安装和使用 Astra 控制中心，您需要满足特定的要求 "要求"。

从较高的层面来看， Astra 控制中心的工作原理如下：

- 您可以在本地环境中安装 Astra Control Center 。详细了解如何操作 "安装 Astra 控制中心"。
- 您可以完成一些设置任务，例如：
    - 设置许可

- 添加第一个集群。
- 添加在添加集群时发现的存储后端。
- 添加用于存储应用程序备份的对象存储分段。

详细了解如何操作 "设置 Astra 控制中心"。

您可以将应用程序添加到集群中。或者、如果要管理的集群中已有一些应用程序、则可以使用Astra控制中心对其进行管理。然后、使用Astra控制中心创建快照、备份、克隆和复制关系。

有关详细信息 ...

- "Astra Control Service 文档"
- "Astra 控制中心文档"
- "Astra Trident 文档"
- "Astra Control API文档"
- "ONTAP 文档"

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。确保您的环境满足这些要求、以部署和运行Astra Control Center。

支持的主机集群Kubennetes环境

Astra Control Center已通过以下Kubenetes主机环境的验证：

> 确保您选择托管Astra Control Center的Kubennet环境满足环境官方文档中列出的基本资源要求。

主机集群上的Kubnetes分发	支持的版本
基于Azure堆栈HCI的Azure Kubnetes Service	采用AKS 1.24.11至1.26.6的Azure Stack HCI 21H2和22H2
Google Anthos	1.15至1.16 (请参见 Google Anthos入口要求)
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

主机集群资源要求

除了环境的资源要求之外， Astra 控制中心还需要以下资源：

> 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。

- **CPU**扩展：托管环境中所有节点的CPU都必须启用AVX扩展。
- 工作节点：总共至少3个工作节点、每个节点具有4个CPU核心和12 GB RAM
- **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。
  - 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。"
  - 使用 `kubectl get nsxlbmonitors -A` 命令以查看是否已将服务监控器配置为接受传入流量。如果存在一个，则不应安装 MetalLB ，因为现有服务监控器将覆盖任何新的负载平衡器配置。
  - 在任何要由 Astra Control 管理的应用程序集群上禁用 TKG 或 TKGi 默认存储类强制实施。您可以通过编辑来执行此操作 `TanzuKubernetesCluster` 命名空间集群上的资源。
  - 在TKG或TKGi环境中部署Astra Control Center时、请注意Astra Control配置程序的特定要求：
    - 集群必须支持有权限的工作负载。
    - 。 `--kubelet-dir` 标志应设置为kubelet目录的位置。默认情况下、此值为 `/var/vcap/data/kubelet`。
    - 使用指定kubelet位置 `--kubelet-dir` 已知适用于Trident操作员、Helm和 `tridentctl` 部署。

服务网格要求

强烈建议您在Astra Control Center主机集群上安装受支持的vilano版本的Istio服务网格。请参见 "支持的版本" 支持的伊斯托伊奥版本。Itio服务网格的品牌版本(例如OpenShift Service Mesh)未通过Astra Control Center的验证。

要将Astra Control Center与主机集群上安装的Isio服务网格集成、必须将此集成作为Astra Control Center的一部分 "安装" 而不是独立于此过程。

> 如果在主机集群上安装和使用Astra Control Center而不配置服务网格、则可能会产生严重的安全影响。

Astra Trident

如果您要在此版本中使用Asta三端安装程序而不是Asta Control配置程序、则支持Asta三端安装程序23.04及更高版本。Asta Control Center需要 Asta Control配置程序 在未来版本中。

Asta Control配置程序

要使用Astra Control配置程序高级存储功能、必须安装Astra Trident 23.10或更高版本并启用 "Astra Control配置程序功能"。要使用最新的Astra Control配置程序功能、您需要最新版本的Astra三端和Astra Control Center。

- 与**Asta Control Center**一起使用的**Asta Control**配置程序的最低版本：已安装并配置Asta Control配置程序23.10或更高版本。

采用**Asta**三端磁盘的**ONTAP**配置

- 存储类：在集群上至少配置一个存储类。如果配置了默认存储类、请确保它是唯一具有默认指定的存储类。
- 存储驱动程序和工作节点：确保为集群中的工作节点配置适当的存储驱动程序，以便Pod可以与后端存储进行交互。Astra 控制中心支持由 Astra Trident 提供的以下 ONTAP 驱动程序：
  - `ontap-nas`
  - `ontap-san`
  - `ontap-san-economy` (此存储类类型不支持应用程序复制)
  - `ontap-nas-economy` (快照和应用程序复制策略不适用于此存储类类型)

存储后端

请确保您有一个受支持的后端、并具有足够的容量。

- 所需存储后端容量：至少500 GB可用
- 支持的后端：Astra Control Center支持以下存储后端：
  - NetApp ONTAP 9.9.1或更高版本的AFF、FAS和ASA系统
  - NetApp ONTAP Select 9.9.1或更高版本
  - NetApp Cloud Volumes ONTAP 9.9.1或更高版本
  - (适用于Astra控制中心技术预览版) NetApp ONTAP 9.10.1或更高版本、用于数据保护操作、作为技术预览版提供
  - Longhorn 1.5.0或更高版本
    - 需要手动创建卷SnapshotClass对象。请参见 "Longhorn文档" 有关说明，请参见。
  - NetApp MetroCluster
    - 受管Kubbernetes集群必须采用延伸型配置。
  - 支持的云提供商提供存储后端

ONTAP 许可证

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

- FlexClone
- SnapMirror：可选。只有在使用SnapMirror技术复制到远程系统时才需要。请参见 "SnapMirror许可证信息"。
- S3许可证：可选。只有ONTAP S3存储分段才需要

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

NetApp MetroCluster

使用NetApp MetroCluster作为存储后端时、您需要执行以下操作：

- 在您使用的Asta三端驱动程序中、将SVM管理LIF指定为后端选项
- 确保您拥有相应的ONTAP许可证

要配置MetroCluster LIF、请参阅每个驱动程序的以下选项和示例：

- "SAN"
- "NAS"

Asta Control Center许可证

Astra Control Center需要Astra Control Center许可证。安装Astra Control Center时、已激活4、800个CPU单元的嵌入式90天评估版许可证。如果您需要更多容量或不同的评估条款、或者要升级到完整许可证、则可以从NetApp获得不同的评估许可证或完整许可证。您需要一个许可证来保护应用程序和数据。

您可以通过注册获取免费试用版来试用Astra Control Center。您可以通过注册进行注册 "此处"。

要设置许可证、请参见 "使用 90 天评估许可证"。

要了解有关许可证工作原理的详细信息、请参见 "许可"。

网络要求

配置操作环境以确保Astra Control Center可以正确通信。需要以下网络配置：

- **FQDN**地址:您必须拥有Astra Control Center的FQDN地址。
- 访问互联网：您应确定是否可以从外部访问互联网。否则、某些功能可能会受到限制、例如向发送支持包 "NetApp 支持站点"。
- 端口访问：Astra Control Center的运行环境使用以下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Astra控制中心与用于访问Astra控制中心的系统之间的双向端口处于打开状态
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）

源	目标	**Port**	协议	目的
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
Astra 控制中心	托管Kubbernetes集群	443/6443 **NOTE**：受管集群使用的端口可能因集群而异。请参见集群软件供应商提供的文档。	HTTPS	与受管集群通信-确保托管Asta Control Center的集群与每个受管集群之间的此端口均处于打开状态

内部 **Kubernetes** 集群的传入

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

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

有关详细信息，请参见 "设置传入以进行负载平衡"。

Google Anthos入口要求

如果在Google Anthos集群上托管Astra Control Center、请注意、默认情况下、Google Anthos包括MetalLB负载平衡器和Istio入口服务、您只需在安装期间使用Astra Control Center的通用入口功能即可。请参见 "Astra Control Center安装文档" 了解详细信息。

支持的 **Web** 浏览器

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

应用程序集群的其他要求

如果您计划使用以下Astra控制中心功能、请记住这些要求：

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

下一步行动

查看 "快速入门" 概述。

Astra 控制中心快速入门

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

1 查看 **Kubernetes** 集群要求

确保您的环境满足以下要求：

- Kubernetes集群*
- "确保主机集群满足操作环境要求"
- "为内部Kubernetes集群的负载平衡配置传入"

存储集成

- "确保您的环境包含Astra Control配置程序"
- "启用Astra Control配置程序高级管理和存储配置功能"
- "准备集群工作节点"
- "配置存储后端"
- "配置存储类"
- "安装卷快照控制器"
- "创建卷快照类"
- ONTAP 凭据*
- "配置ONTAP 凭据"

2 下载并安装**Astra**控制中心

完成以下安装任务：

- "从NetApp 支持站点 下载页面下载Astra控制中心"
- 获取NetApp许可证文件：
  - 如果您正在评估Astra Control Center、则已包含嵌入式评估许可证
  - "如果您已购买Astra Control Center、请生成许可证文件"
- "安装 Astra 控制中心"
- "执行其他可选配置步骤"

3 完成一些初始设置任务

完成一些基本任务以开始使用：

- "添加许可证"

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

4 使用 **Astra** 控制中心

完成Asta Control Center设置后、请使用Asta Control UI或 "Astra Control API" 要开始管理和保护应用程序、请执行以下操作：

- "管理帐户"：用户、角色、LDAP、凭据等。
- "管理通知"
- "管理应用程序"：定义要管理的资源。
- "保护应用程序"：配置保护策略以及复制、克隆和迁移应用程序。

有关详细信息 ...

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

安装概述

选择并完成以下 Astra 控制中心安装过程之一：

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

根据您的环境、安装Astra控制中心后可能需要进行其他配置：

- "安装后配置Astra控制中心"

使用标准流程安装 **Astra** 控制中心

要安装Astra Control Center、请下载安装映像并执行以下步骤。您可以使用此操作步骤在互联网连接或通风环境中安装 Astra 控制中心。

有关Astra控制中心安装过程的演示、请参见 "此视频"。

开始之前

- 满足环境前提条件： "开始安装之前，请为 Astra Control Center 部署准备您的环境"。

> 在第三个容错域或二级站点中部署A作用 力控制中心。对于应用程序复制和无缝灾难恢复、建议执行此操作。

- 确保服务运行状况良好：检查所有API服务是否均处于运行状况良好且可用：

```
kubectl get apiservices
```

- 确保具有可路由的**FQDN**：您计划使用的Astra FQDN可以路由到集群。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。
- 配置证书管理器：如果集群中已存在证书管理器，则需要执行某些操作 "前提条件步骤" 这样、Astra控制中心就不会尝试安装自己的证书管理器。默认情况下、Astra控制中心会在安装期间安装自己的证书管理器。
- **(**仅限**ONTAP SAN**驱动程序**)**启用多路径：如果使用的是ONTAP SAN驱动程序、请确保在所有Kubernetes集群上启用了多路径。

您还应考虑以下事项：

- 获取**NetApp Astra**控件映像注册表的访问权限：

  您可以选择从NetApp映像注册表中获取Astra控件的安装映像和增强功能、例如Astra控件配置程序。

  a. 记录您登录注册表所需的Astra Control帐户ID。

     您可以在Astra Control Service Web UI中查看您的帐户ID。选择页面右上角的图图标，选择*API access*并记下您的帐户ID。

  b. 在同一页面中，选择*Generate API t令牌*并将API令牌字符串复制到剪贴板，然后将其保存在编辑器中。

  c. 登录到Asta Control注册表：

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

- 安装用于安全通信的服务网格：强烈建议使用保护Astra Control主机集群通信通道的安全 "支持的服务网格"。

> 只能在Astra Control Center期间将Astra Control Center与服务网格集成 "安装" 而不是独立于此过程。不支持从网格化环境切换回非网格化环境。

  要使用Isio服务网格、您需要执行以下操作：

  - 添加 `istio-injection:enabled` label 在部署Asta Control Center之前将Asta命名空间添加到Asta命名空间。
  - 使用 `Generic` 入口设置 并为提供备用入口 外部负载平衡。
  - 对于Red Hat OpenShift集群、您需要进行定义 `NetworkAttachmentDefinition` 在所有关联的Astra Control Center名空间上 (`netapp-acc-operator`， `netapp-acc`， `netapp-monitoring` 或任何已替换的自定义卷)。

```
cat <<EOF | oc -n netapp-acc-operator create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF

cat <<EOF | oc -n netapp-acc create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF

cat <<EOF | oc -n netapp-monitoring create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF
```

步骤

要安装 Astra 控制中心，请执行以下步骤：

- 下载并提取Astra控制中心
- [如果使用本地注册表、请完成其他步骤]
- [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

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

下载并提取**Astra**控制中心

从以下位置之一下载Astra Control Center映像：

- **Astra**控制服务映像注册表：如果您不对Astra控制中心映像使用本地注册表，或者如果您更喜欢使用此方法

从NetApp 支持站点 下载捆绑包，请使用此选项。

- **Astra**：如果将本地注册表与NetApp 支持站点 控制中心映像一起使用，请使用此选项。

Astra Control图像注册表

1. 登录Asta Control Service。
2. 在信息板上，选择*Deploy a self-managed instance* of Asta Control*。
3. 按照说明登录到Astra Control映像注册表、提取Astra Control Center安装映像并提取该映像。

NetApp 支持站点

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

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

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

此时将显示输出 `Verified OK` 验证成功后。

3. 从Astra Control Center捆绑包中提取映像：

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

如果使用本地注册表、请完成其他步骤

如果您计划将Astra控制中心捆绑包推送到本地注册表、则需要使用NetApp Astra kubect命令行插件。

安装**NetApp Astra kubectl**插件

要安装最新的NetApp Astra kubecl命令行插件、请完成以下步骤。

开始之前

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

如果您已从先前安装中安装了插件、 "确保您已安装最新版本" 在完成这些步骤之前。

步骤

1. 列出可用的NetApp Astra kubect插件二进制文件：

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

```
ls kubectl-astra/
```

2. 将操作系统和CPU架构所需的文件移至当前路径、并将其重命名为 `kubectl-astra`：

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

将映像添加到注册表

1. 如果您计划将Astra Control Center捆绑包推送到本地注册表、请为容器引擎完成相应的步骤顺序：

Docker

a. 更改为tarball的根目录。您应看到 `acc.manifest.bundle.yaml` 文件和以下目录：

```
acc/
kubectl-astra/
acc.manifest.bundle.yaml
```

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

- 将<BUNDLE_FILE> 替换为Astra Control捆绑包文件的名称 (`acc.manifest.bundle.yaml` ）。
- 将<MY_FULL_REGISTRY_PATH> 替换为Docker存储库的URL；例如 "<a href="https://<docker-registry>"" class="bare">https://<docker-registry>"。
- 将<MY_REGISTRY_USER> 替换为用户名。
- 将<MY_REGISTRY_TOKEN> 替换为注册表的授权令牌。

```
kubectl astra packages push-images -m <BUNDLE_FILE> -r
<MY_FULL_REGISTRY_PATH> -u <MY_REGISTRY_USER> -p
<MY_REGISTRY_TOKEN>
```

Podman

a. 更改为tarball的根目录。您应看到此文件和目录：

```
acc/
kubectl-astra/
acc.manifest.bundle.yaml
```

b. 登录到注册表：

```
podman login <YOUR_REGISTRY>
```

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

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

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=24.02.0-69
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed
's/Loaded image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

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

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=24.02.0-69
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed
's/Loaded image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

根据您的注册表配置、此脚本创建的映像路径应类似于以下内容：

```
https://downloads.example.io/docker-astra-control-
prod/netapp/astra/acc/24.02.0-69/image:version
```

2. 更改目录：

```
cd manifests
```

为具有身份验证要求的注册表设置命名空间和密钥

1. 导出Astra Control Center主机集群的kubeconfig:

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

> 在完成安装之前、请确保您的kubeconfig"指向要安装Astra Control Center的集群。

2. 如果您使用的注册表需要身份验证，则需要执行以下操作：

   a. 创建 `NetApp-Acc-operator` 命名空间：

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

   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：

   > 占位符 `your_registry_path` 应与您先前上传的映像的位置匹配(例如、`[Registry_URL]/netapp/astra/astracc/24.02.0-69`）。

```
kubectl create secret docker-registry astra-registry-cred -n netapp-acc-
operator --docker-server=cr.astra.netapp.io --docker
-username=[astra_account_id] --docker-password=[astra_api_token]
```

+

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

+

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

   a. 创建 `netapp-acc` (或自定义命名的)命名空间。

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

   b. 为创建密钥 `netapp-acc` (或自定义命名的)命名空间。根据您的注册表首选项、添加Docker信息并运行相应的命令之一：

```
kubectl create secret docker-registry astra-registry-cred -n [netapp-acc or custom namespace] --docker-server=cr.astra.netapp.io --docker-username=[astra_account_id] --docker-password=[astra_api_token]
```

```
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]
```

安装 **Astra** 控制中心操作员

1. (仅限本地注册表)如果使用的是本地注册表、请完成以下步骤：

   a. 打开Asta控制中心操作员部署YAML：

   ```
   vim astra_control_center_operator_deploy.yaml
   ```

   ⓘ 以下步骤将提供一个标注的YAML示例。

   b. 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: astra-registry-cred}]
   ```

   c. 更改 `ASTRA_IMAGE_REGISTRY` 。 `kube-rbac-proxy` 将映像推送到注册表路径中 上一步。

   d. 更改 `ASTRA_IMAGE_REGISTRY` 。 `acc-operator-controller-manager` 将映像推送到注册表路径中 上一步。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    control-plane: controller-manager
  name: acc-operator-controller-manager
  namespace: netapp-acc-operator
spec:
  replicas: 1
  selector:
    matchLabels:
      control-plane: controller-manager
  strategy:
    type: Recreate
  template:
```

```
    metadata:
      labels:
        control-plane: controller-manager
    spec:
      containers:
      - args:
        - --secure-listen-address=0.0.0.0:8443
        - --upstream=http://127.0.0.1:8080/
        - --logtostderr=true
        - --v=10
        image: ASTRA_IMAGE_REGISTRY/kube-rbac-proxy:v4.8.0
        name: kube-rbac-proxy
        ports:
        - containerPort: 8443
          name: https
      - args:
        - --health-probe-bind-address=:8081
        - --metrics-bind-address=127.0.0.1:8080
        - --leader-elect
        env:
        - name: ACCOP_LOG_LEVEL
          value: "2"
        - name: ACCOP_HELM_INSTALLTIMEOUT
          value: 5m
        image: ASTRA_IMAGE_REGISTRY/acc-operator:24.02.68
        imagePullPolicy: IfNotPresent
        livenessProbe:
          httpGet:
            path: /healthz
            port: 8081
          initialDelaySeconds: 15
          periodSeconds: 20
        name: manager
        readinessProbe:
          httpGet:
            path: /readyz
            port: 8081
          initialDelaySeconds: 5
          periodSeconds: 10
        resources:
          limits:
            cpu: 300m
            memory: 750Mi
          requests:
            cpu: 100m
            memory: 75Mi
```

```
        securityContext:
          allowPrivilegeEscalation: false
      imagePullSecrets: []
      securityContext:
        runAsUser: 65532
      terminationGracePeriodSeconds: 10
```

2. 安装 Astra 控制中心操作员：

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

展开样本响应：

```
namespace/netapp-acc-operator created
customresourcedefinition.apiextensions.k8s.io/astracontrolcenters.as
tra.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
```

3. 验证Pod是否正在运行：

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

配置 **Astra** 控制中心

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

```
vim astra_control_center.yaml
```

以下步骤将提供一个标注的YAML示例。

2. 修改或确认以下设置：

帐户名称

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

AstraVersion

正在设置 ...	指导	Type	示例
`astraVersion`	要部署的Astra控制中心版本。无需对此设置执行任何操作、因为此值将预先填充。	string	`24.02.0-69`

AstraAddress

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

AutoSupport

您在此部分中所做的选择将决定您是否参与NetApp主动支持应用程序NetApp Active IQ以及数据的发送位置。需要互联网连接(端口442)、所有支持数据均会匿名化。

正在设置 ...	使用 ...	指导	Type	示例
`autoSupport.enrolled`	两者之一 `enrolled` 或 `url` 必须选择字段	更改 `enrolled` 用于将AutoSupport连接到 `false` 对于不具有Internet连接或保留的站点 `true` 对于已连接站点。的设置 `true` 允许将匿名数据发送给NetApp以供支持。默认选择为 `false` 和表示不会向NetApp发送任何支持数据。	布尔值	`false` (此值为默认值)
`autoSupport.url`	两者之一 `enrolled` 或 `url` 必须选择字段	此URL用于确定匿名数据的发送位置。	string	`https://support.netapp.com/asupprod/post/1.0/postAsup`

email

正在设置 ...	指导	Type	示例
`email`	更改 `email` 字符串到默认的初始管理员地址。复制此电子邮件地址以在中使用 后续步骤。此电子邮件地址将用作初始帐户的用户名、用于登录到UI、并在Astra Control中收到事件通知。	string	`admin@example.com`

firstName

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

姓氏

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

imageRegistry

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

正在设置 ...	使用 ...	指导	Type	示例
`imageRegistry.name`	Required	托管部署Asta Control Center所需的所有映像的Asta Control映像注册表的名称。该值将预先填充、除非配置了本地注册表、否则不需要执行任何操作。对于本地注册表、请将此现有值替换为您在中推送图像的映像注册表的名称 上一步。请勿使用 `http://` 或 `https://` 注册表名称。	string	`cr.astra.netapp.io` (默认) `example.registry.com/astra` (本地注册表示例)
`imageRegistry.secret`	可选	用于通过映像注册表进行身份验证的Kubernetes密钥的名称。该值将预先填充、除非您配置了本地注册表以及在中为该注册表输入的字符串、否则不需要执行任何操作 `imageRegistry.name` 需要密钥。 重要说明：如果您使用的本地注册表不需要授权、则必须将其删除 `secret` 行内 `imageRegistry` 否则安装将失败。	string	`astra-registry-cred`

存储类

正在设置 ...	指导	Type	示例
`storageClass`	更改 `storageClass` 价值来自 `ontap-gold` 安装所需的另一个存储类资源。运行命令 `kubectl get sc` 以确定已配置的现有存储类。必须在清单文件中输入Asta Control配置程序配置的存储类之一 (`astra-control-center-<version>.manifest`)、并将用于Astra PV。如果未设置、则会使用默认存储类。 注意：如果配置了默认存储类、请确保它是唯一具有默认标注的存储类。	string	`ontap-gold`

volumeReclaimPolicy

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

正在载入类型

正在设置 ...	指导	Type	选项
`ingressType`	请使用以下入口类型之一： 通用 (`ingressType: "Generic"`)(默认) 如果您正在使用另一个入口控制器或希望使用您自己的入口控制器、请使用此选项。部署Astra Control Center后、您需要配置 "入口控制器" 以使用URL公开Astra控制中心。 重要信息：如果您要将服务网格与Astra Control Center结合使用、则必须选择 `Generic` 作为入口类型并设置您自己的 "入口控制器"。 * AccTraefik* (`ingressType: "AccTraefik"`) 如果您不希望配置入口控制器、请使用此选项。这将部署Astra控制中心 `traefik` 网关作为Kubernetes loadbalancer类型的服务。 Astra控制中心使用类型为"loadbalancer"的服务 (`svc/traefik`)、并要求为其分配可访问的外部IP地址。如果您的环境允许使用负载平衡器、但您尚未配置一个平衡器、则可以使用MetalLB或其他外部服务负载平衡器为该服务分配外部IP地址。在内部 DNS 服务器配置中，您应将为 Astra 控制中心选择的 DNS 名称指向负载平衡的 IP 地址。 注意：有关"load平衡器"和传入服务类型的详细信息、请参见 "要求"。	string	• `Generic` (这是默认值) • `AccTraefik`

可扩展大小

正在设置 ...	指导	Type	选项
`scaleSize`	默认情况下、Astra将使用高可用性(HA) `scaleSize` 的 `Medium` ，可在HA中部署大多数服务，并部署多个副本以实现冗余。使用 `scaleSize` 作为 `Small`A作用 是减少所有服务的副本数量，但主要服务除外，以减少使用量。提示：`Medium` 部署包含大约100个Pod (不包括瞬时工作负载)。100个Pod基于一个三主节点和三个工作节点配置)。请注意您问题描述 的环境中可能存在的每POD网络限制限制、尤其是在考虑灾难恢复方案时。	string	• `Small` • `Medium` (这是默认值)

AstraResourcesScal

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

其他值

将以下附加值添加到Astra控制中心CR中、以防止安装已知问题描述：

```
additionalValues:
    keycloak-operator:
      livenessProbe:
        initialDelaySeconds: 180
      readinessProbe:
        initialDelaySeconds: 180
```

CRD

您在本节中的选择决定了Astra控制中心应如何处理CRD。

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

在完成安装之前、请确保为您的配置选择了正确的存储类和入口类型。

示例**Astra_control_cCenter.yaml**

```
apiVersion: astra.netapp.io/v1
kind: AstraControlCenter
metadata:
  name: astra
spec:
  accountName: "Example"
  astraVersion: "ASTRA_VERSION"
  astraAddress: "astra.example.com"
  autoSupport:
    enrolled: true
  email: "[admin@example.com]"
  firstName: "SRE"
  lastName: "Admin"
  imageRegistry:
    name: "[cr.astra.netapp.io or your_registry_path]"
    secret: "astra-registry-cred"
  storageClass: "ontap-gold"
  volumeReclaimPolicy: "Retain"
  ingressType: "Generic"
  scaleSize: "Medium"
  astraResourcesScaler: "Default"
  additionalValues:
    keycloak-operator:
      livenessProbe:
        initialDelaySeconds: 180
      readinessProbe:
        initialDelaySeconds: 180
  crds:
    externalTraefik: false
    externalCertManager: false
```

完成 **Astra** 控制中心和操作员安装

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

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

2. 如果您正在Astra Control Center中使用服务网格、请将以下标签添加到 `netapp-acc` 或自定义命名空间：

> 您的入口类型 (`ingressType`)必须设置为 `Generic` 在Astra Control Center CR中、然后继续执行此命令。

```
kubectl label ns [netapp-acc or custom namespace] istio-
injection:enabled
```

3. （建议） "启用严格的MTLS" 对于Isio service Mesh：

```
kubectl apply -n istio-system -f - <<EOF
apiVersion: security.istio.io/v1beta1
kind: PeerAuthentication
metadata:
  name: default
spec:
  mtls:
    mode: STRICT
EOF
```

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

> A作用 力控制中心操作员将自动检查环境要求。缺少 "要求" 发生原因 您的安装是否失败或Astra 控制中心是否无法正常运行。请参见 下一节 检查与自动系统检查相关的警告消息。

验证系统状态

您可以使用kubectl命令验证系统状态。如果您更喜欢使用 OpenShift ，则可以使用同等的 oc 命令执行验证步骤。

步骤

1. 验证安装过程是否未生成与验证检查相关的警告消息：

```
kubectl get acc [astra or custom Astra Control Center CR name] -n
[netapp-acc or custom namespace] -o yaml
```

> A作用 力控制中心操作员日志中还会报告其他警告消息。

2. 更正自动需求检查报告的环境中的任何问题。

> 您可以通过确保环境满足来更正问题 "要求" A作用 控制中心。

3. 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。

展开以显示样本响应

```
acc-helm-repo-5bd77c9ddd-8wxm2             1/1     Running     0
1h
activity-5bb474dc67-8l9ss                  1/1     Running     0
1h
activity-5bb474dc67-qbrtq                  1/1     Running     0
1h
api-token-authentication-6wbj2             1/1     Running     0
1h
api-token-authentication-9pgw6             1/1     Running     0
1h
api-token-authentication-tqf6d             1/1     Running     0
1h
asup-5495f44dbd-z4kft                      1/1     Running     0
1h
authentication-6fdd899858-5x45s            1/1     Running     0
1h
bucketservice-84d47487d-n9xgp              1/1     Running     0
1h
bucketservice-84d47487d-t5jhm              1/1     Running     0
1h
cert-manager-5dcb7648c4-hbldc              1/1     Running     0
1h
cert-manager-5dcb7648c4-nr9qf              1/1     Running     0
1h
cert-manager-cainjector-59b666fb75-bk2tf   1/1     Running     0
1h
cert-manager-cainjector-59b666fb75-pfnck   1/1     Running     0
1h
cert-manager-webhook-c6f9b6796-ngz2x       1/1     Running     0
1h
cert-manager-webhook-c6f9b6796-rwtbn       1/1     Running     0
1h
certificates-5f5b7b4dd-52tnj               1/1     Running     0
1h
certificates-5f5b7b4dd-gtjbx               1/1     Running     0
1h
certificates-expiry-check-28477260-dz5vw   0/1     Completed   0
1h
cloud-extension-6f58cc579c-lzfmv           1/1     Running     0
1h
cloud-extension-6f58cc579c-zw2km           1/1     Running     0
1h
cluster-orchestrator-79dd5c8d95-qjg92      1/1     Running     0
```

```
1h
composite-compute-85dc84579c-nz82f         1/1     Running     0
1h
composite-compute-85dc84579c-wx2z2         1/1     Running     0
1h
composite-volume-bff6f4f76-789nj           1/1     Running     0
1h
composite-volume-bff6f4f76-kwnd4           1/1     Running     0
1h
credentials-79fd64f788-m7m8f               1/1     Running     0
1h
credentials-79fd64f788-qnc6c               1/1     Running     0
1h
entitlement-f69cdbd77-4p2kn                1/1     Running     0
1h
entitlement-f69cdbd77-hswm6                1/1     Running     0
1h
features-7b9585444c-7xd7m                  1/1     Running     0
1h
features-7b9585444c-dcqwc                  1/1     Running     0
1h
fluent-bit-ds-crq8m                        1/1     Running     0
1h
fluent-bit-ds-gmgq8                        1/1     Running     0
1h
fluent-bit-ds-gzr4f                        1/1     Running     0
1h
fluent-bit-ds-j6sf6                        1/1     Running     0
1h
fluent-bit-ds-v4t9f                        1/1     Running     0
1h
fluent-bit-ds-x7j59                        1/1     Running     0
1h
graphql-server-6cc684fb46-2x8lr            1/1     Running     0
1h
graphql-server-6cc684fb46-bshbd            1/1     Running     0
1h
hybridauth-84599f79fd-fjc7k                1/1     Running     0
1h
hybridauth-84599f79fd-s9pmn                1/1     Running     0
1h
identity-95df98cb5-dvlmz                   1/1     Running     0
1h
identity-95df98cb5-krf59                   1/1     Running     0
1h
influxdb2-0                                1/1     Running     0
```

```
1h
keycloak-operator-6d4d688697-cfq8b         1/1     Running     0
1h
krakend-5d5c8f4668-7bq8g                   1/1     Running     0
1h
krakend-5d5c8f4668-t8hbn                   1/1     Running     0
1h
license-689cdd4595-2gsc8                   1/1     Running     0
1h
license-689cdd4595-g6vwk                   1/1     Running     0
1h
login-ui-57bb599956-4fwgz                  1/1     Running     0
1h
login-ui-57bb599956-rhztb                  1/1     Running     0
1h
loki-0                                     1/1     Running     0
1h
metrics-facade-846999bdd4-f7jdm            1/1     Running     0
1h
metrics-facade-846999bdd4-lnsxl            1/1     Running     0
1h
monitoring-operator-6c9d6c4b8c-ggkrl       2/2     Running     0
1h
nats-0                                     1/1     Running     0
1h
nats-1                                     1/1     Running     0
1h
nats-2                                     1/1     Running     0
1h
natssync-server-6df7d6cc68-9v2gd           1/1     Running     0
1h
nautilus-64b7fbdd98-bsgwb                  1/1     Running     0
1h
nautilus-64b7fbdd98-djlhw                  1/1     Running     0
1h
openapi-864584bccc-75nlv                   1/1     Running     0
1h
openapi-864584bccc-zh6bx                   1/1     Running     0
1h
polaris-consul-consul-server-0             1/1     Running     0
1h
polaris-consul-consul-server-1             1/1     Running     0
1h
polaris-consul-consul-server-2             1/1     Running     0
1h
polaris-keycloak-0                         1/1     Running     2 (1h
```

```
ago)      1h
polaris-keycloak-1                          1/1     Running     0
1h
polaris-keycloak-db-0                       1/1     Running     0
1h
polaris-keycloak-db-1                       1/1     Running     0
1h
polaris-keycloak-db-2                       1/1     Running     0
1h
polaris-mongodb-0                           1/1     Running     0
1h
polaris-mongodb-1                           1/1     Running     0
1h
polaris-mongodb-2                           1/1     Running     0
1h
polaris-ui-66476dcf87-f6s8j                 1/1     Running     0
1h
polaris-ui-66476dcf87-ztjk7                 1/1     Running     0
1h
polaris-vault-0                             1/1     Running     0
1h
polaris-vault-1                             1/1     Running     0
1h
polaris-vault-2                             1/1     Running     0
1h
public-metrics-bfc4fc964-x4m79              1/1     Running     0
1h
storage-backend-metrics-7dbb88d4bc-g78cj    1/1     Running     0
1h
storage-provider-5969b5df5-hjvcm            1/1     Running     0
1h
storage-provider-5969b5df5-r79ld            1/1     Running     0
1h
task-service-5fc9dc8d99-4q4f4               1/1     Running     0
1h
task-service-5fc9dc8d99-8l5zl               1/1     Running     0
1h
task-service-task-purge-28485735-fdzkd      1/1     Running     0
12m
telegraf-ds-2rgm4                           1/1     Running     0
1h
telegraf-ds-4qp6r                           1/1     Running     0
1h
telegraf-ds-77frs                           1/1     Running     0
1h
telegraf-ds-bc725                           1/1     Running     0
```

```
1h
telegraf-ds-cvmxf                          1/1     Running     0
1h
telegraf-ds-tqzgj                          1/1     Running     0
1h
telegraf-rs-5wtd8                          1/1     Running     0
1h
telemetry-service-6747866474-5djnc         1/1     Running     0
1h
telemetry-service-6747866474-thb7r         1/1     Running     1 (1h
ago)      1h
tenancy-5669854fb6-gzdzf                   1/1     Running     0
1h
tenancy-5669854fb6-xvsm2                   1/1     Running     0
1h
traefik-8f55f7d5d-4lgfw                    1/1     Running     0
1h
traefik-8f55f7d5d-j4wt6                    1/1     Running     0
1h
traefik-8f55f7d5d-p6gcq                    1/1     Running     0
1h
trident-svc-7cb5bb4685-54cnq               1/1     Running     0
1h
trident-svc-7cb5bb4685-b28xh               1/1     Running     0
1h
vault-controller-777b9bbf88-b5bqt          1/1     Running     0
1h
vault-controller-777b9bbf88-fdfd8          1/1     Running     0
1h
```

4. (可选)观看 `acc-operator` 用于监控进度的日志：

```
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。如果日志中指示的集群注册失败、您可以尝试通过重新注册 "在UI中添加集群工作流" 或 API 。

5. 在所有Pod运行时、验证安装是否成功 (`READY` 为 `True`)并获取登录Astra Control Center时要使用的初始设置密码：

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

响应：

```
NAME    UUID                                   VERSION     ADDRESS
READY
astra   9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f  24.02.0-69
10.111.111.111  True
```

> 复制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` ）。

设置传入以进行负载平衡

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

部署Astra Control Center后、您需要配置传入控制器、以便使用URL公开Astra Control Center。

设置步骤因所使用的入口控制器类型而异。Astra控制中心支持多种传入控制器类型。这些设置过程提供了一些常见传入控制器类型的示例步骤。

开始之前

- 所需 "入口控制器" 应已部署。
- 。 "入口类" 应已创建与入口控制器对应的。

Istio入口的步骤

1. 配置Istio入口。

   > 此操作步骤 假定使用"默认"配置文件部署Istio。

2. 为传入网关收集或创建所需的证书和专用密钥文件。

   您可以使用CA签名或自签名证书。公用名必须为Astra地址(FQDN)。

   命令示例：

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

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

   命令示例：

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

密钥名称应与`istio-Infile.yaml`文件中提供的`spec.tls.secretName`匹配。

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

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

5. 应用更改：

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

6. 检查入口状态：

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

响应：

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

7. 完成Astra控制中心安装。

nginx 入口控制器的步骤

1. 创建类型的密钥 `kubernetes.io/tls` 中的TLS专用密钥和证书 `netapp-acc` (或自定义命名的)命名空间、如中所述 "TLS 密钥"。
2. 在中部署传入资源 `netapp-acc` (或自定义命名的)命名空间 (`nginx-Ingress.yaml` 在此示例中使用)：

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

3. 应用更改：

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

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

OpenShift 入口控制器的步骤

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

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

登录到 **Astra** 控制中心 **UI**

安装Astra Control Center后、您将更改默认管理员的密码、并登录到Astra Control Center UI信息板。

步骤

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

   您可以在登录后创建自定义证书。

3. 在Astra Control Center登录页面上、输入您用于的值 `email` 在中 `astra_control_center.yaml` CR时间 您安装了 Astra 控制中心、后跟初始设置密码 (`ACC-[UUID]`）。

   如果您输入的密码三次不正确，管理员帐户将锁定 15 分钟。

4. 选择 * 登录 * 。
5. 根据提示更改密码。

   如果这是您第一次登录、但您忘记了密码、并且尚未创建任何其他管理用户帐户、请联系 "NetApp 支持" 以获得密码恢复帮助。

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

对安装进行故障排除

如果任何服务处于 `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。

选项

- 要检查 Astra 控制中心操作员日志，请输入以下内容：

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

- 要检查Asta Control Center CR的输出：

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

其他安装过程

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

下一步行动

- (可选)根据您的环境、完成安装后操作 "配置步骤"。
- "安装Astra Control Center、登录到UI并更改密码后、您需要设置许可证、添加集群、启用身份验证、管理存储以及添加存储分段"。

配置外部证书管理器

如果Kubernetes集群中已存在证书管理器、则需要执行一些前提步骤、以使Astra控制中心不会安装自己的证书管理器。

步骤

1. 确认已安装证书管理器：

```
kubectl get pods -A | grep 'cert-manager'
```

响应示例：

```
cert-manager   essential-cert-manager-84446f49d5-sf2zd              1/1
Running    0      6d5h
cert-manager   essential-cert-manager-cainjector-66dc99cc56-9ldmt   1/1
Running    0      6d5h
cert-manager   essential-cert-manager-webhook-56b76db9cc-fjqrq      1/1
Running    0      6d5h
```

2. 为创建证书/密钥对 `astraAddress` FQDN：

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

响应示例：

```
Generating a 2048 bit RSA private key
..................+++
.........................................+++
writing new private key to 'tls.key'
```

3. 使用先前生成的文件创建密钥：

```
kubectl create secret tls selfsigned-tls --key tls.key --cert tls.crt -n
<cert-manager-namespace>
```

响应示例：

```
secret/selfsigned-tls created
```

4. 创建 `ClusterIssuer` 文件*精确*如下、但包含的命名空间位置 `cert-manager` Pod的安装：

```
apiVersion: cert-manager.io/v1
kind: ClusterIssuer
metadata:
  name: astra-ca-clusterissuer
  namespace: <cert-manager-namespace>
spec:
  ca:
    secretName: selfsigned-tls
```

```
kubectl apply -f ClusterIssuer.yaml
```

响应示例：

```
clusterissuer.cert-manager.io/astra-ca-clusterissuer created
```

5. 验证是否已 `ClusterIssuer` 已正确启动。 `Ready` 必须为 `True` 在继续操作之前：

```
kubectl get ClusterIssuer
```

响应示例：

```
NAME                      READY   AGE
astra-ca-clusterissuer    True    9s
```

6. 完成 "Astra 控制中心安装过程"。有一个 "Astra控制中心集群YAML的所需配置步骤" 其中、您可以更改CRD值以指示证书管理器是外部安装的。您必须在安装期间完成此步骤、以使Astra控制中心能够识别外部证书管理器。

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。

开始之前

- 满足环境前提条件： "开始安装之前，请为 Astra Control Center 部署准备您的环境"。

> 在第三个容错域或二级站点中部署A作用 力控制中心。对于应用程序复制和无缝灾难恢复、建议执行此操作。

- 确保集群操作员和**API**服务运行正常：
  - 在OpenShift集群中、确保所有集群操作员均处于运行状况良好的状态：

    ```
    oc get clusteroperators
    ```

  - 在OpenShift集群中、确保所有API服务均处于运行状况良好的状态：

    ```
    oc get apiservices
    ```

- 确保具有可路由的**FQDN**：您计划使用的Astra FQDN可以路由到集群。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。
- 获取**OpenShift**权限：要执行所述的安装步骤、您需要拥有对Red Hat OpenShift容器平台的所有必要权限和访问权限。
- 配置证书管理器：如果集群中已存在证书管理器，则需要执行某些操作 "前提条件步骤" 这样、Astra控制中心就不会安装自己的证书管理器。默认情况下、Astra控制中心会在安装期间安装自己的证书管理器。
- 设置**Kubernetes**入口控制器：如果您有一个Kubernetes入口控制器来管理对服务的外部访问、例如集群中的负载平衡、则需要将其设置为与Astra Control Center配合使用：

a. 创建操作员命名空间：

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

b. "完成设置" 适用于您的入口控制器类型。

- **(仅限ONTAP SAN驱动程序)启用多路径：**如果使用的是ONTAP SAN驱动程序、请确保在所有Kubernetes集群上启用了多路径。

您还应考虑以下事项：

- 获取**NetApp Astra**控件映像注册表的访问权限：

  您可以选择从NetApp映像注册表中获取Astra控件的安装映像和增强功能、例如Astra控件配置程序。

  a. 记录您登录注册表所需的Astra Control帐户ID。

     您可以在Astra Control Service Web UI中查看您的帐户ID。选择页面右上角的图图标，选择*API access*并记下您的帐户ID。

  b. 在同一页面中，选择*Generate API t令牌*并将API令牌字符串复制到剪贴板，然后将其保存在编辑器中。

  c. 登录到Asta Control注册表：

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

- 安装用于安全通信的服务网格：强烈建议使用保护Astra Control主机集群通信通道的安全 "支持的服务网格"。

  > 只能在Astra Control Center期间将Astra Control Center与服务网格集成 "安装" 而不是独立于此过程。不支持从网格化环境切换回非网格化环境。

  要使用Isio服务网格、您需要执行以下操作：

  - 添加 `istio-injection:enabled` 在部署Asta Control Center之前、请标记Asta命名空间。
  - 使用 `Generic` 入口设置 并为提供备用入口 "外部负载平衡"。
  - 对于Red Hat OpenShift集群、您需要进行定义 `NetworkAttachmentDefinition` 在所有关联的Astra Control Center名空间上 (`netapp-acc-operator`， `netapp-acc`， `netapp-monitoring` 或任何已替换的自定义卷)。

```
cat <<EOF | oc -n netapp-acc-operator create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF

cat <<EOF | oc -n netapp-acc create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF

cat <<EOF | oc -n netapp-monitoring create -f -
apiVersion: "k8s.cni.cncf.io/v1"
kind: NetworkAttachmentDefinition
metadata:
  name: istio-cni
EOF
```

步骤

- 下载并提取Astra控制中心
- [如果使用本地注册表、请完成其他步骤]
- [找到操作员安装页面]
- [安装操作员]
- 安装 Astra 控制中心

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

下载并提取**Astra**控制中心

从以下位置之一下载Astra Control Center映像：

- **Astra**控制服务映像注册表：如果您不对Astra控制中心映像使用本地注册表，或者如果您更喜欢使用此方法从NetApp 支持站点 下载捆绑包，请使用此选项。
- **Astra**：如果将本地注册表与NetApp 支持站点 控制中心映像一起使用，请使用此选项。

Astra Control图像注册表

1. 登录Asta Control Service。
2. 在信息板上，选择*Deploy a self-managed instance* of Asta Control*。
3. 按照说明登录到Astra Control映像注册表、提取Astra Control Center安装映像并提取该映像。

NetApp 支持站点

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

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

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

   此时将显示输出 `Verified OK` 验证成功后。

3. 从Astra Control Center捆绑包中提取映像：

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

如果使用本地注册表、请完成其他步骤

如果您计划将Astra控制中心捆绑包推送到本地注册表、则需要使用NetApp Astra kubect命令行插件。

安装**NetApp Astra kubectl**插件

要安装最新的NetApp Astra kubecl命令行插件、请完成以下步骤。

开始之前

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

如果您已从先前安装中安装了插件、 "确保您已安装最新版本" 在完成这些步骤之前。

步骤

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

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

```
ls kubectl-astra/
```

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

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

将映像添加到注册表

1. 如果您计划将Astra Control Center捆绑包推送到本地注册表、请为容器引擎完成相应的步骤顺序：

Docker

a. 更改为tarball的根目录。您应看到 `acc.manifest.bundle.yaml` 文件和以下目录：

```
acc/
kubectl-astra/
acc.manifest.bundle.yaml
```

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

- 将<BUNDLE_FILE> 替换为Astra Control捆绑包文件的名称 (`acc.manifest.bundle.yaml` ）。
- 将<MY_FULL_REGISTRY_PATH> 替换为Docker存储库的URL；例如 "<a href="https://<docker-registry>"" class="bare">https://<docker-registry>"。
- 将<MY_REGISTRY_USER> 替换为用户名。
- 将<MY_REGISTRY_TOKEN> 替换为注册表的授权令牌。

```
kubectl astra packages push-images -m <BUNDLE_FILE> -r
<MY_FULL_REGISTRY_PATH> -u <MY_REGISTRY_USER> -p
<MY_REGISTRY_TOKEN>
```

Podman

a. 更改为tarball的根目录。您应看到此文件和目录：

```
acc/
kubectl-astra/
acc.manifest.bundle.yaml
```

b. 登录到注册表：

```
podman login <YOUR_REGISTRY>
```

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

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

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=24.02.0-69
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed
's/Loaded image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

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

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=24.02.0-69
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed
's/Loaded image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

根据您的注册表配置、此脚本创建的映像路径应类似于以下内容：

```
https://downloads.example.io/docker-astra-control-
prod/netapp/astra/acc/24.02.0-69/image:version
```

2. 更改目录：

```
cd manifests
```

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：

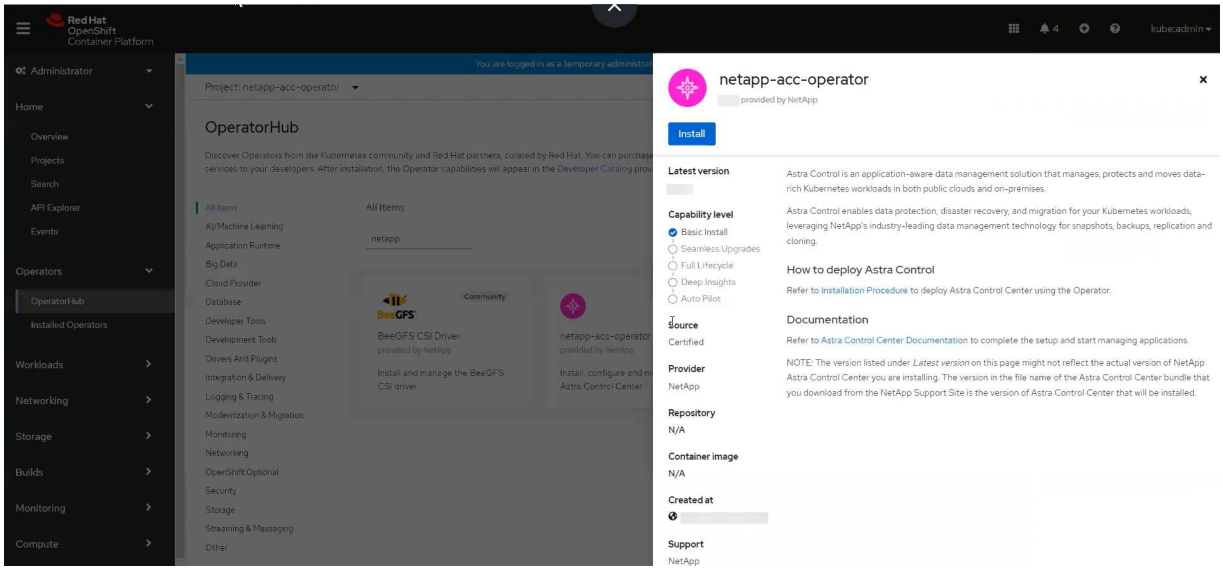
Red Hat OpenShift Web控制台

a. 登录到 OpenShift 容器平台 UI 。

b. 从侧面菜单中，选择 * 运算符 > OperatorHub * 。

> 使用此运算符只能升级到Astra Control Center的当前版本。

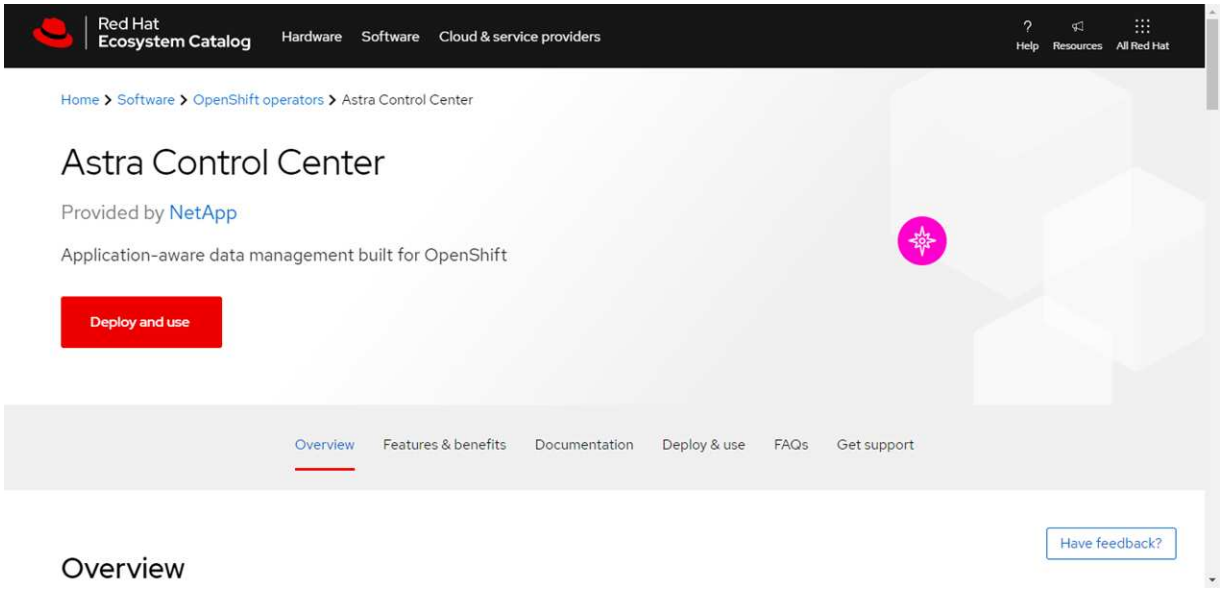
c. 搜索 `netapp-acc` 并选择NetApp Asta控制中心操作员。



Red Hat 生态系统目录

a. 选择 NetApp Astra 控制中心 "运算符"。

b. 选择*部署和使用*。



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。

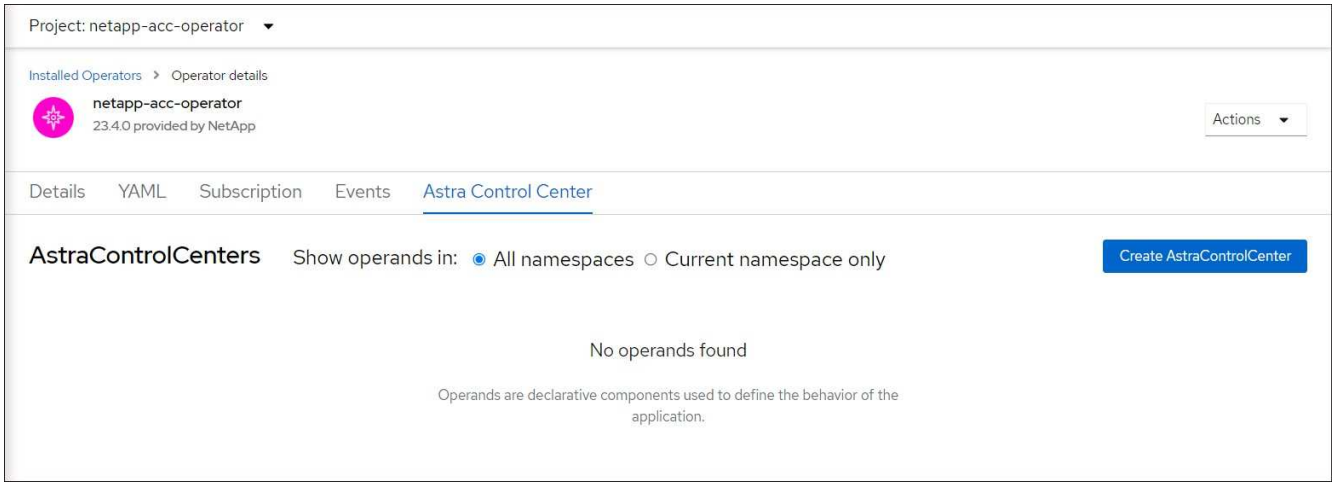
   c. 选择 * 安装 * 。

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



2. 填写 `Create AstraControlCenter` Form 字段：

   a. 保留或调整 Astra 控制中心名称。

   b. 为Astra控制中心添加标签。

   c. 启用或禁用自动支持。建议保留自动支持功能。

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

   e. 输入Asta Control Center版本、例如24.02.0-69。

   f. 输入帐户名称，电子邮件地址和管理员姓氏。

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

   h. 选择安装的比例大小。

> 默认情况下、Astra将使用高可用性(HA) `scaleSize` 的 `Medium`，可在HA中部署大多数服务，并部署多个副本以实现冗余。使用 `scaleSize` 作为 `Small`A作用 是减少所有服务的副本数量，但主要服务除外，以减少使用量。

i. 选择入口类型：

- 通用 (`ingressType: "Generic"`)(默认)

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

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

  如果您不希望配置入口控制器、请使用此选项。这将部署Astra控制中心 `traefik` 网关作为Kubernetes的"loadbalancer"类型服务。

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

> 有关"负载平衡器"和传入服务类型的详细信息、请参见 "要求"。

a. 在*Image Registry*中，除非配置了本地注册表，否则请使用默认值。对于本地注册表、请将此值替换为您在上一步中推送映像的本地映像注册表路径。请止步 `http://` 或 `https://` 在地址字段中。

b. 如果您使用的映像注册表需要身份验证、请输入映像密钥。

> 如果您使用的注册表需要身份验证、 在集群上创建密钥。

c. 输入管理员的名字。

d. 配置资源扩展。

e. 提供默认存储类。

> 如果配置了默认存储类、请确保它是唯一具有默认标注的存储类。

f. 定义 CRD 处理首选项。

3. 选择YAML视图以查看您选择的设置。

4. 选择 `Create` 。

创建注册表密钥

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

1. 为Astra控制中心操作员创建命名空间：

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

2. 在此命名空间中创建密钥：

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

> Astra Control仅支持Docker注册表机密。

3. 完成中的其余字段 创建AstraControlCenter表单字段。

下一步行动

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

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

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

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

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

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

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

在 Amazon Web Services 中部署 Astra 控制中心

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

AWS所需的功能

在AWS中部署Astra Control Center之前、您需要满足以下条件：

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

- "满足 Astra 控制中心的要求"。
- NetApp Cloud Central account
- 如果使用OCP、则Red Hat OpenShift Container Platform (OCP)权限(在命名空间级别用于创建Pod)
- AWS 凭据，访问 ID 和机密密钥，具有用于创建存储分段和连接器的权限
- AWS 帐户弹性容器注册（ Elastic Container Registry ， ECR ）访问和登录
- 要访问Astra Control UI、需要AWS托管区域和Amazon 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 Container Platform 4.11至4.13

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、Asta Control Center还需要特定的资源。请参见 "Astra 控制中心运营环境要求"。

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

AWS 部署概述

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

下面详细介绍了其中每个步骤。

1. 确保您具有足够的 IAM 权限。
2. 在 AWS 上安装 RedHat OpenShift 集群。
3. 配置 AWS。
4. 配置适用于AWS的NetApp BlueXP。
5. 安装适用于AWS的Astra控制中心。

确保您具有足够的 IAM 权限

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

请参见 "初始 AWS 凭据"。

在 AWS 上安装 RedHat OpenShift 集群

在 AWS 上安装 RedHat OpenShift 容器平台集群。

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

配置 AWS

接下来、将AWS配置为创建虚拟网络、设置EC2计算实例以及创建AWS S3存储分段。如果无法访问NetApp

Astra控制中心映像注册表、则还需要创建一个Elastic Container Registry (ECR)来托管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"。

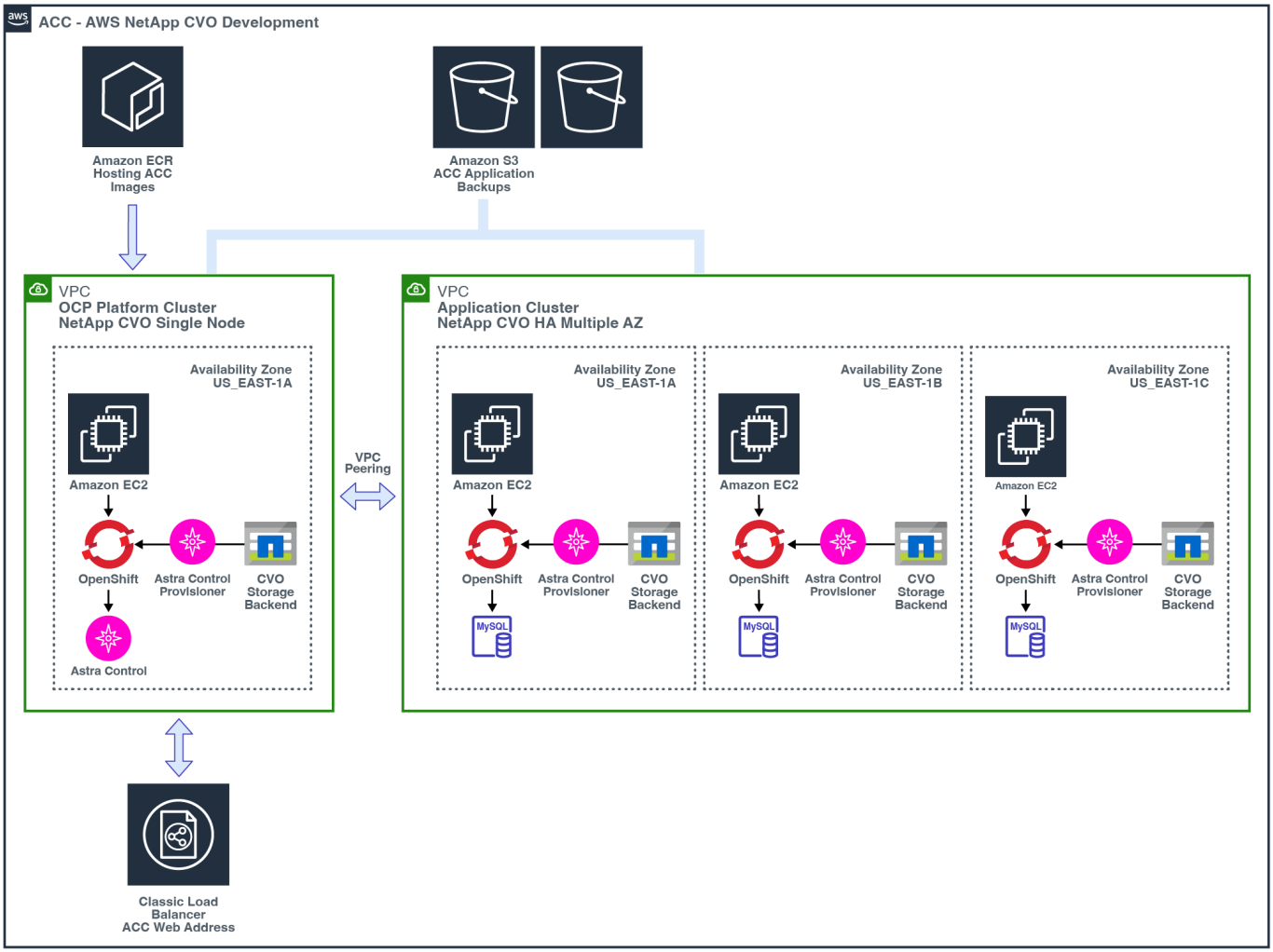
1. 创建AWS虚拟网络。
2. 查看 EC2 计算实例。这可以是 AWS 中的裸机服务器或 VM 。
3. 如果实例类型尚未与主节点和工作节点的 Astra 最低资源要求匹配，请更改 AWS 中的实例类型以满足 Astra 要求。 请参见 "Astra 控制中心要求"。
4. 至少创建一个 AWS S3 存储分段来存储备份。
5. (可选)如果无法访问NetApp映像注册表、请执行以下操作：
   a. 创建AWS Elastic Container Registry (ECR)以托管所有Astra Control Center映像。

      > 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 。

   b. 将A作用 力控制中心图像推送到您定义的注册表。

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

以下是 AWS 部署示例：



配置适用于**AWS**的**NetApp BlueXP**

使用NetApp BlueXP (以前称为Cloud Manager)创建工作空间、向AWS添加连接器、创建工作环境并导入集群。

按照BlueXP文档完成以下步骤。请参见以下内容：

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

步骤

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

b. 请注意右上角的Asta Control配置程序版本。

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。Asta Control配置程序会在导入和发现过程中自动安装。

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

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

安装适用于AWS的Astra控制中心

请遵循标准 "Astra 控制中心安装说明"。

> AWS使用通用S3存储分段类型。

在Google Cloud Platform中部署Astra控制中心

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

GCP所需的功能

在GCP中部署Astra Control Center之前、您需要以下各项：

- Astra Control Center 许可证。请参见 "Astra 控制中心许可要求"。
- "满足 Astra 控制中心的要求"。
- NetApp Cloud Central account
- 如果使用OCP、则为Red Hat OpenShift Container Platform (OCP) 4.11至4.13
- 如果使用OCP、则Red Hat OpenShift Container Platform (OCP)权限(在命名空间级别用于创建Pod)
- GCP服务帐户、具有创建存储分段和连接器的权限

GCP的操作环境要求

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

除了环境的资源要求之外、Asta Control Center还需要特定的资源。请参见 "Astra 控制中心运营环境要求"。

GCP部署概述

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

下面详细介绍了其中每个步骤。

1. 在GCP上安装RedHat OpenShift集群。
2. 创建GCP项目和虚拟私有云。
3. 确保您具有足够的 IAM 权限。

4. 配置GCP。
5. 为GCP配置NetApp BlueXP。
6. 安装适用于GCP的Astra控制中心。

在GCP上安装RedHat OpenShift集群

第一步是在GCP上安装RedHat OpenShift集群。

有关安装说明，请参见以下内容：

- "在GCP中安装OpenShift集群"
- "创建GCP服务帐户"

创建GCP项目和虚拟私有云

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

> OpenShift 可能会创建自己的资源组。此外、您还应定义GCP VPC。请参见 OpenShift 文档。

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

确保您具有足够的 IAM 权限

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

请参见 "初始GCP凭据和权限"。

配置GCP

接下来、配置GCP以创建VPC、设置计算实例以及创建Google Cloud Object Storage。如果无法访问NetApp Astra控制中心映像注册表、您还需要创建一个Google容器注册表来托管Astra控制中心映像、并将这些映像推送到此注册表。

按照GCP文档完成以下步骤。请参见在GCP中安装OpenShift集群。

1. 在GCP中创建一个GCP项目和VPC、该项目和VPC计划用于具有CVO后端的OCP集群。
2. 查看计算实例。此服务器可以是GCP中的裸机服务器或VM。
3. 如果实例类型尚未与主节点和工作节点的Astra最低资源要求匹配、请在GCP中更改实例类型以满足Astra要求。请参见 "Astra 控制中心要求"。
4. 至少创建一个GCP Cloud Storage Bucket以存储备份。
5. 创建存储分段访问所需的密钥。
6. (可选)如果无法访问NetApp映像注册表、请执行以下操作：
    a. 创建Google容器注册表以托管Asta Control Center映像。
    b. 为所有Astra控制中心映像设置用于Docker推/拉的Google容器注册表访问权限。

示例：可以通过输入以下脚本将Astra Control Center映像推送到此注册表：

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

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

```
manifestfile=acc.manifest.bundle.yaml
GCP_CR_REGISTRY=<target GCP image registry>
ASTRA_REGISTRY=<source Astra Control Center image registry>

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

7. 设置 DNS 区域。

为GCP配置NetApp BlueXP

使用NetApp BlueXP (以前称为Cloud Manager)创建工作空间、向GCP添加连接器、创建工作环境并导入集群。

按照BlueXP文档完成以下步骤。请参见 "GCP中的Cloud Volumes ONTAP 入门"。

开始之前

- 使用所需的IAM权限和角色访问GCP服务帐户

步骤

1. 将凭据添加到BlueXP。请参见 "正在添加GCP帐户"。
2. 为GCP添加一个连接器。
   a. 选择"GCP"作为提供程序。
   b. 输入GCP凭据。请参见 "从BlueXP在GCP中创建连接器"。
   c. 确保连接器正在运行，然后切换到该连接器。
3. 为您的云环境创建一个工作环境。
   a. 位置："GCP"
   b. 类型： Cloud Volumes ONTAP HA
4. 导入 OpenShift 集群。集群将连接到您刚刚创建的工作环境。
   a. 选择 * K8s* > * 集群列表 * > * 集群详细信息 * ，查看 NetApp 集群详细信息。

b. 请注意右上角的Asta Control配置程序版本。

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

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

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

> Cloud Volumes ONTAP 可以作为单个节点运行、也可以在高可用性(HA)中运行。如果已启用HA、请记下在GCP中运行的HA状态和节点部署状态。

安装适用于**GCP**的**Astra**控制中心

请遵循标准 "Astra 控制中心安装说明"。

> GCP使用通用S3存储分段类型。

1. 生成Docker密钥以提取用于Astra控制中心安装的映像：

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

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

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

Azure所需的功能

在Azure中部署Astra Control Center之前、您需要满足以下条件：

- Astra Control Center 许可证。请参见 "Astra 控制中心许可要求"。
- "满足 Astra 控制中心的要求"。
- NetApp Cloud Central account
- 如果使用OCP、则为Red Hat OpenShift Container Platform (OCP) 4.11至4.13
- 如果使用OCP、则Red Hat OpenShift Container Platform (OCP)权限(在命名空间级别用于创建Pod)
- 具有用于创建存储分段和连接器的权限的 Azure 凭据

Azure 的操作环境要求

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

除了环境的资源要求之外、Asta Control Center还需要特定的资源。请参见 "Astra 控制中心运营环境要求"。

Azure 部署概述

下面简要介绍了适用于 Azure 的 Astra 控制中心的安装过程。

下面详细介绍了其中每个步骤。

1. 在 Azure 上安装 RedHat OpenShift 集群。
2. 创建 Azure 资源组。
3. 确保您具有足够的 IAM 权限。
4. 配置 Azure。
5. 为Azure配置NetApp BlueXP (以前称为Cloud Manager)。
6. 安装和配置适用于Azure的Astra控制中心。

在 Azure 上安装 RedHat OpenShift 集群

第一步是在 Azure 上安装 RedHat OpenShift 集群。

有关安装说明，请参见以下内容：

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

创建 Azure 资源组

至少创建一个 Azure 资源组。

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

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

确保您具有足够的 IAM 权限

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

请参见 "Azure 凭据和权限"。

配置 Azure

接下来、将Azure配置为创建虚拟网络、设置计算实例以及创建Azure Blb容器。如果您无法访问NetApp Astra控制中心映像注册表、则还需要创建Azure容器注册表(ACR)来托管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。您需要使用存储帐户来创建要在Astra Control Center中用作存储分段的容器。
6. 创建存储分段访问所需的密钥。
7. (可选)如果无法访问NetApp映像注册表、请执行以下操作：

   a. 创建Azure容器注册表(ACR)以托管Asta控制中心映像。
   b. 为所有Astra Control Center映像设置Docker推送/拉取的ACR访问权限。
   c. 使用以下脚本将Astra Control Center映像推送到此注册表：

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

   - 示例 * ：

```
manifestfile=acc.manifest.bundle.yaml
AZ_ACR_REGISTRY=<target Azure ACR image registry>
ASTRA_REGISTRY=<source Astra Control Center image registry>

while IFS= read -r image; do
    echo "image: $ASTRA_REGISTRY/$image $AZ_ACR_REGISTRY/$image"
    root_image=${image%:*}
    echo $root_image
    docker pull $ASTRA_REGISTRY/$image
    docker tag $ASTRA_REGISTRY/$image $AZ_ACR_REGISTRY/$image
    docker push $AZ_ACR_REGISTRY/$image
done < acc.manifest.bundle.yaml
```

8. 设置 DNS 区域。

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

使用BlueXP (以前称为Cloud Manager)创建工作空间、向Azure添加连接器、创建工作环境并导入集群。

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

开始之前

使用所需的 IAM 权限和角色访问 Azure 帐户

步骤

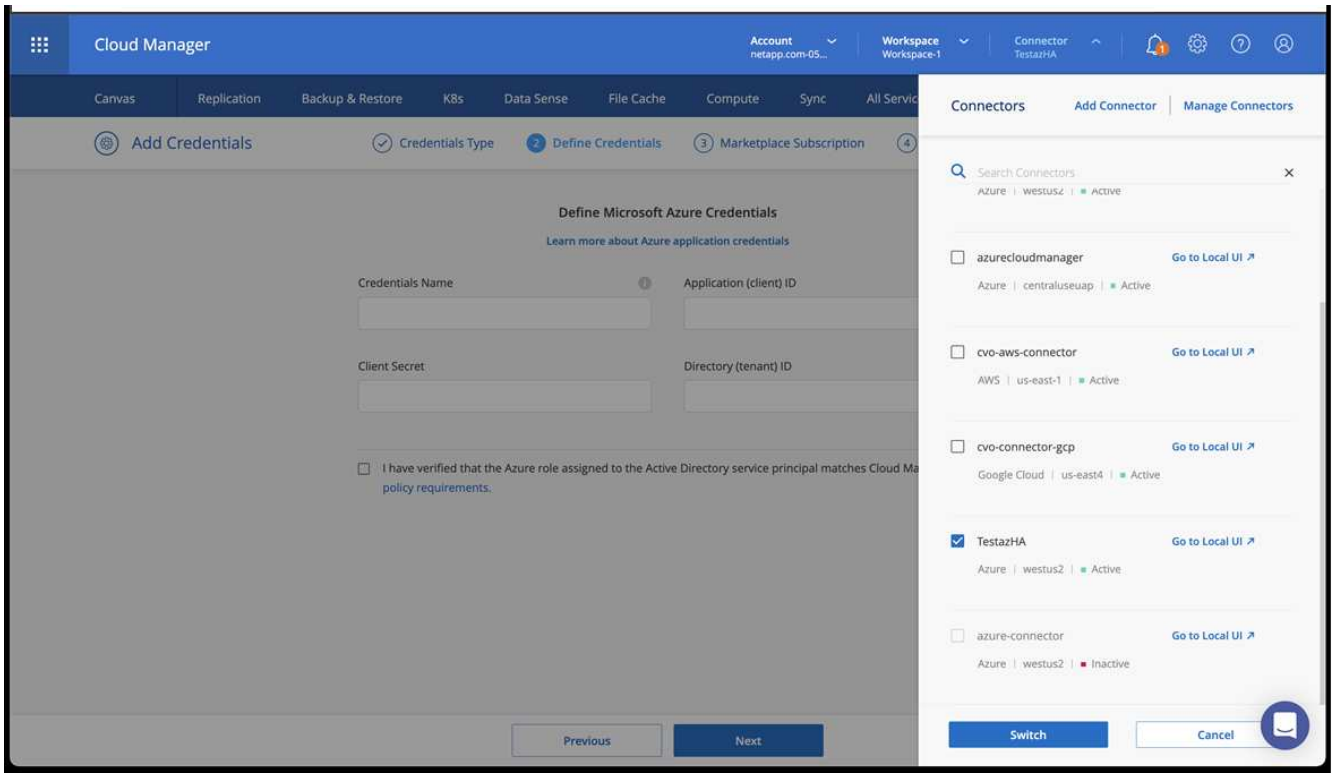
1. 将凭据添加到BlueXP。
2. 添加适用于 Azure 的连接器。请参见 "BlueXP策略"。

a. 选择 * Azure * 作为提供程序。

b. 输入 Azure 凭据，包括应用程序 ID ，客户端密钥和目录（租户） ID 。

请参见 "从BlueXPr.在Azure中创建连接器"。

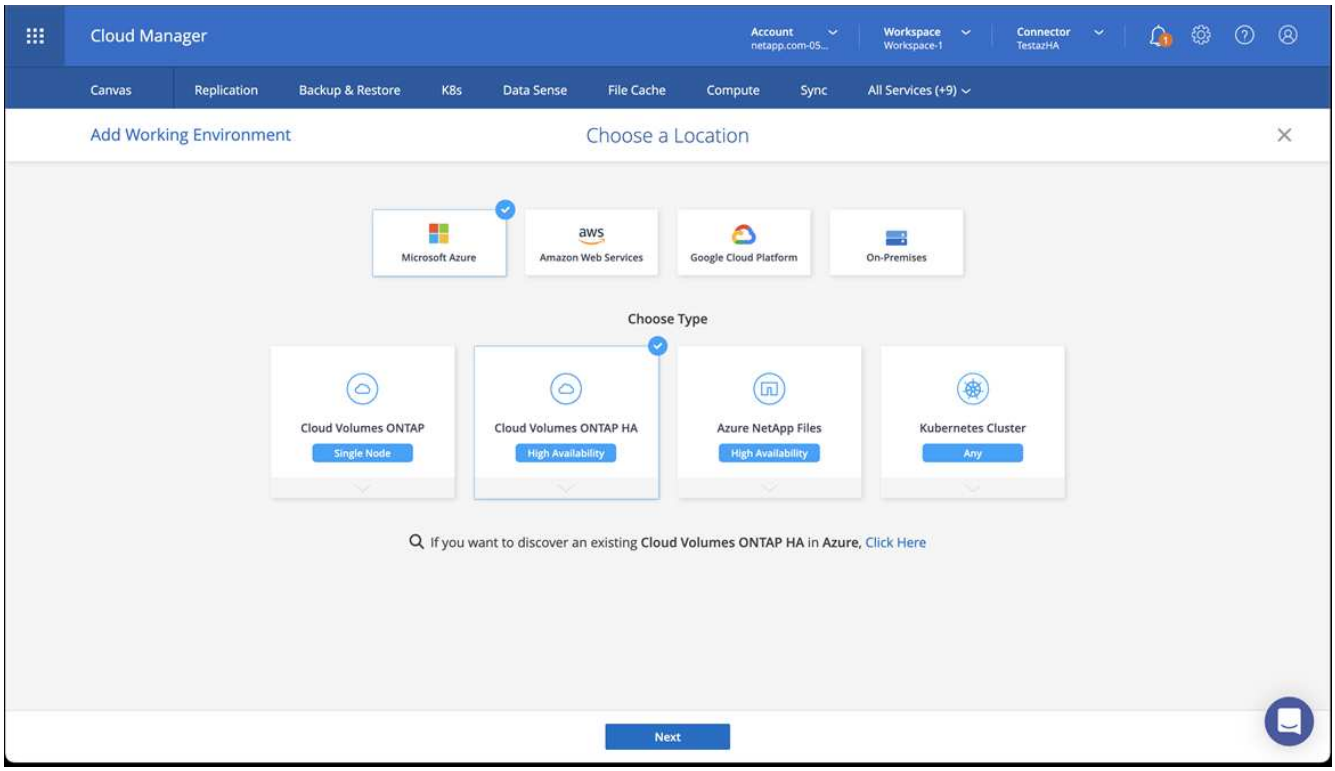
3. 确保连接器正在运行，然后切换到该连接器。



4. 为您的云环境创建一个工作环境。

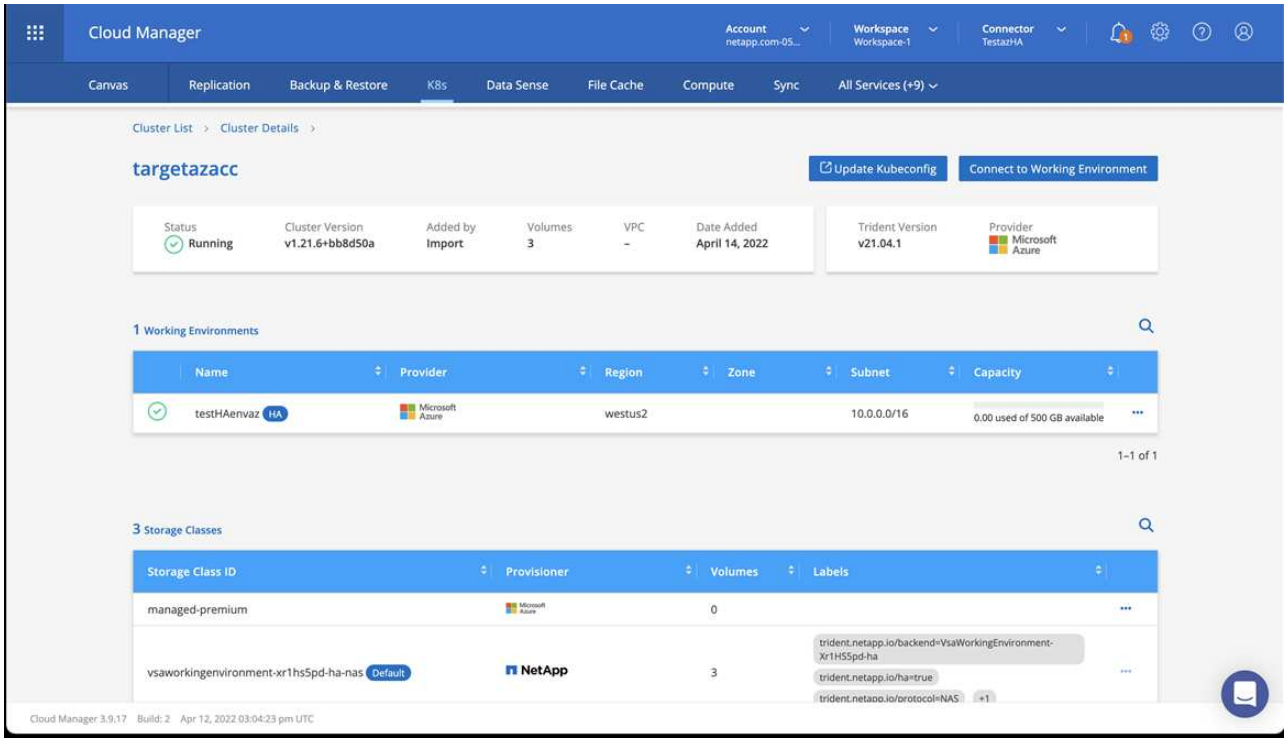
a. 位置： "Microsoft Azure" 。

b. 键入： Cloud Volumes ONTAP HA 。



5. 导入 OpenShift 集群。集群将连接到您刚刚创建的工作环境。

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



   b. 请注意右上角的Asta Control配置程序版本。

   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。

Asta Control配置程序会在导入和发现过程中自动安装。

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

安装和配置适用于**Azure**的**Astra**控制中心

按照标准安装 Astra 控制中心 "安装说明"。

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

安装后配置**Astra**控制中心

根据您的环境、安装Astra控制中心后可能需要进行其他配置。

消除资源限制

某些环境使用ResourceQuotas和LimitRanges对象来防止命名空间中的资源占用集群上的所有可用CPU和内存。Astra控制中心未设置最大限制、因此不符合这些资源的要求。如果您的环境采用这种方式配置、则需要从计划安装Astra控制中心的命名空间中删除这些资源。

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

步骤

1. 在中获取资源配额 `netapp-acc` (或自定义名称)命名空间：

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

响应：

```
NAME          AGE   REQUEST                                          LIMIT
pods-high     16s   requests.cpu: 0/20, requests.memory: 0/100Gi
limits.cpu: 0/200, limits.memory: 0/1000Gi
pods-low      15s   requests.cpu: 0/1, requests.memory: 0/1Gi
limits.cpu: 0/2, limits.memory: 0/2Gi
pods-medium   16s   requests.cpu: 0/10, requests.memory: 0/20Gi
limits.cpu: 0/20, limits.memory: 0/200Gi
```

2. 按名称删除所有资源配额：

```
kubectl delete resourcequota pods-high -n [netapp-acc or custom
namespace]
```

```
kubectl delete resourcequota pods-low -n [netapp-acc or custom
namespace]
```

```
kubectl delete resourcequota pods-medium -n [netapp-acc or custom
namespace]
```

3. 在中获取限制范围 `netapp-acc` (或自定义名称)命名空间：

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

响应：

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

4. 按名称删除限制范围：

```
kubectl delete limitrange cpu-limit-range -n [netapp-acc or custom
namespace]
```

添加自定义 TLS 证书

默认情况下、Astra控制中心对传入控制器流量(仅在某些配置中)和Web浏览器的Web UI身份验证使用自签名TLS证书。对于生产环境、您应删除现有的自签名TLS证书、并将其替换为由证书颁发机构(CA)签名的TLS证书。

> 默认的自签名证书用于两种类型的连接：
>
> - 通过HTTPS连接到Astra控制中心Web UI
> - 传入控制器流量(仅当 `ingressType: "AccTraefik"` 属性已在中设置 `astra_control_center.yaml` 在安装Astra Control Center期间生成文件)
>
> 替换默认TLS证书将替换用于对这些连接进行身份验证的证书。

开始之前

- 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>
```

使用命令行添加新证书

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>
```

CRD：

```
#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>
```

响应：

```
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：

> 此示例使用 `dnsNames` 属性以指定Asta Control Center DNS地址。Astra Control Center不支持使用公用名(Common Name、CN)属性指定DNS地址。

```
apiVersion: cert-manager.io/v1
kind: Certificate
metadata:
  <strong>name: <certificate-name></strong>
  namespace: <ACC-deployment-namespace>
spec:
  <strong>secretName: <certificate-secret-name></strong>
  duration: 2160h # 90d
  renewBefore: 360h # 15d
  dnsNames:
  <strong>- <astra.dnsname.example.com></strong> #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>
```

响应：

```
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TLS存储CRD以指向新证书密钥名称、并将括号<>中的占位符值替换为适当的信息

```
kubectl edit tlsstores.traefik.io -n <ACC-deployment-namespace>
```

CRD：

```
...
spec:
  defaultCertificate:
    secretName: <certificate-secret-name>
```

8. 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.io -n <ACC-deployment-namespace>
```

CRD：

```
...
 tls:
    secretName: <certificate-secret-name>
```

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

设置 Astra 控制中心

添加 Astra 控制中心的许可证

安装Astra Control Center时、已安装嵌入式评估版许可证。如果您正在评估Astra Control Center、则可以跳过此步骤。

您可以使用Astra Control UI或添加新许可证 "Astra Control API"。

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

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

> 要更新现有评估版或完整许可证、请参见 "更新现有许可证"。

开始之前

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

步骤

1. 登录到 Astra 控制中心 UI 。
2. 选择 * 帐户 * > * 许可证 * 。
3. 选择 * 添加许可证 * 。
4. 浏览到您下载的许可证文件（ NLF ）。
5. 选择 * 添加许可证 * 。

- 帐户 * > * 许可证 * 页面显示许可证信息，到期日期，许可证序列号，帐户 ID 和使用的 CPU 单元。

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

启用**Asta Control**配置程序

Astra Trident 23.10及更高版本提供了使用Astra Control配置程序的选项、允许获得许可的Astra Control用户访问高级存储配置功能。除了基于标准Asta三端CSI的功能之外、Asta Control配置程序还提供了此扩展功能。

在即将推出的Astra Control更新中、Astra Control配置程序将取代Astra Trandent作为存储配置程序和流程编排程序、并且Astra Control必须使用它。因此、强烈建议Asta Control用户启用Asta Control配置程序。Asta三元数据将继续保持开源状态、并使用NetApp的新CSI和其他功能进行发布、维护、支持和更新。

关于此任务

如果您是Astra控制中心的许可用户、并且希望使用Astra控制配置程序功能、则应遵循此操作步骤。如果您是Astra三端数据库的用户、并且希望使用Astra Control配置程序提供的其他功能、而不同时使用Astra Control、则还应遵循此操作步骤。

对于每种情况、默认情况下在Astra Trident 24.02中不会启用配置程序功能、必须启用此功能。

开始之前

如果要启用Asta Control配置程序、请先执行以下操作：

Asta Control为用户提供Asta Control Center

- 获取**Astra**控制中心许可证：您需要 "Asta Control Center许可证" 启用Astra Control配置程序并访问它提供的功能。
- 安装或升级到**Astra Control Center 23.10**或更高版本：如果您计划在Astra Control中使用最新的Astra Control配置程序功能(24.02)，则需要最新的Astra Control Center版本(24.02)。
- *确认集群具有一个AMD64*系统架构：Astra Control配置程序映像在amd64和ARM64 CPU架构中都提供，但Astra Control Center仅支持amd64。
- 获取用于注册表访问的**Asta Control**服务帐户：如果要使用Asta Control注册表而不是NetApp 支持站点 来下载Asta Control配置程序映像、请完成的注册 "Asta Control Service帐户"。填写并提交表单并创建BlueXP帐户后、您将收到Astra Control Service欢迎电子邮件。
- 如果您安装了**Astra**三端安装程序，请确认其版本在四个版本的窗口内：如果您的Astra三端安装程序在版本24.02的四个版本窗口内，则可以使用Astra Control置备程序直接升级到Astra三端安装程序24.02。例如、您可以直接从Asta三端23.04升级到24.02。

仅适用于**Asta Control**配置程序用户

- 获取**Astra**控制中心许可证：您需要 "Asta Control Center许可证" 启用Astra Control配置程序并访问它提供的功能。
- 如果您安装了**Astra**三端安装程序，请确认其版本在四个版本的窗口内：如果您的Astra三端安装程序在版本24.02的四个版本窗口内，则可以使用Astra Control置备程序直接升级到Astra三端安装程序24.02。例如、您可以直接从Asta三端23.04升级到24.02。
- 获取**Astra Control Service**帐户以访问注册表：您需要访问注册表才能下载Astra Control配置程序映像。要开始使用、请完成的注册 "Asta Control Service帐户"。填写并提交表单并创建BlueXP帐户后、您将收到Astra Control Service欢迎电子邮件。

(步骤1)获取Asta Control配置程序映像

Asta控制中心用户可以使用Asta控制注册表或NetApp 支持站点 方法获取Asta控制配置程序映像。要在不使用Astra Control的情况下使用Astra Control配置程序的Astra Trent用户应使用注册表方法。

Astra Control图像注册表

> 在此操作步骤中、您可以使用Podman而不是Docker执行命令。如果您使用的是Windows环境、建议使用PowerShell。

1. 访问NetApp Astra控件映像注册表：
   a. 登录到Astra Control Service Web UI、然后选择页面右上角的图图标。
   b. 选择* API访问*。
   c. 记下您的帐户ID。
   d. 在同一页面中，选择*Generate API t令牌*并将API令牌字符串复制到剪贴板，然后将其保存在编辑器中。
   e. 使用您的首选方法登录Astra Control注册表：

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

```
crane auth login cr.astra.netapp.io -u <account-id> -p <api-
token>
```

2. (仅限自定义注册表)按照以下步骤将图像移动到自定义注册表。如果您不使用注册表、请按照中的三端修复操作符步骤进行操作 "下一节"。
   a. 从注册表中提取Asta Control配置程序映像：

> 提取的映像不支持多个平台、只支持与提取映像的主机相同的平台、例如Linux amd64。

```
docker pull cr.astra.netapp.io/astra/trident-acp:24.02.0
--platform <cluster platform>
```

示例

```
docker pull cr.astra.netapp.io/astra/trident-acp:24.02.0 --platform
linux/amd64
```

   a. 标记图像：

```
docker tag cr.astra.netapp.io/astra/trident-acp:24.02.0
<my_custom_registry>/trident-acp:24.02.0
```

b. 将映像推送到自定义注册表：

```
docker push <my_custom_registry>/trident-acp:24.02.0
```

> 您可以使用"删除副本"来替代运行以下Docker命令：
> `crane copy cr.astra.netapp.io/astra/trident-acp:24.02.0 <my_custom_registry>/trident-acp:24.02.0`

NetApp 支持站点

1. 下载Asta Control配置程序包 (`trident-acp-[version].tar`) "Astra Control Center下载页面"。
2. (建议但可选)下载Astra Control Center的证书和签名捆绑包(astra-control-crier-certs -[version].tar.gz)、以验证trident - acp-[version] tar捆绑包的签名。

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

```
openssl dgst -sha256 -verify certs/AstraControlCenterDockerImages-public.pub -signature certs/trident-acp-[version].tar.sig trident-acp-[version].tar
```

3. 加载Asta Control配置程序映像：

```
docker load < trident-acp-24.02.0.tar
```

响应：

```
Loaded image: trident-acp:24.02.0-linux-amd64
```

4. 标记图像：

```
docker tag trident-acp:24.02.0-linux-amd64 <my_custom_registry>/trident-acp:24.02.0
```

5. 将映像推送到自定义注册表：

```
docker push <my_custom_registry>/trident-acp:24.02.0
```

(第2步)在Asta Trdent中启用Asta Control配置程序

确定原始安装方法是否使用 "运算符(手动或使用Helm)或trdentcd" 并根据原始方法完成相应的步骤。

Asta三端操作员

1. "下载Asta三端安装程序并解压缩"。
2. 如果您尚未安装Astra三端安装程序、或者您从初始Astra三端安装程序中删除了操作员、请完成以下步骤：
   a. 创建客户需求日：

   ```
   kubectl create -f
   deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.y
   aml
   ```

   b. 创建三项命名空间 (`kubectl create namespace trident`)或确认三项命名空间仍然存在（`kubectl get all -n trident`）。如果已删除此命名空间、请重新创建它。
3. 将Astra Trdent更新到24.02.0：

   > 对于运行Kubornetes 1.24或更早版本的集群、请使用 `bundle_pre_1_25.yaml`。对于运行Kubernetes 1.25或更高版本的集群、请使用 `bundle_post_1_25.yaml`。

   ```
   kubectl -n trident apply -f trident-installer/deploy/<bundle-
   name.yaml>
   ```

4. 验证Astra trident是否正在运行：

   ```
   kubectl get torc -n trident
   ```

   响应：

   ```
   NAME       AGE
   trident    21m
   ```

5. 如果您有一个使用机密的注册表，请创建一个用于提取Astra Control置备程序映像的密钥：

   ```
   kubectl create secret docker-registry <secret_name> -n trident
   --docker-server=<my_custom_registry> --docker-username=<username>
   --docker-password=<token>
   ```

6. 编辑TridentOrchestrator CR并进行以下编辑：

   ```
   kubectl edit torc trident -n trident
   ```

a. 为Astra三端映像设置自定义注册表位置或从Astra Control注册表中提取该映像 (`tridentImage: <my_custom_registry>/trident:24.02.0` 或 `tridentImage: netapp/trident:24.02.0`）。

b. 启用Asta Control配置程序 (`enableACP: true`）。

c. 设置Asta Control配置程序映像的自定义注册表位置或将其从Asta Control注册表中提取 (`acpImage: <my_custom_registry>/trident-acp:24.02.0` 或 `acpImage: cr.astra.netapp.io/astra/trident-acp:24.02.0`）。

d. 如果您已建立 图像拉取密钥 在本操作步骤的前面部分、您可以在此处设置它们 (`imagePullSecrets: - <secret_name>`）。使用您在前面步骤中创建的相同名称机密名称。

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  tridentImage: <registry>/trident:24.02.0
  enableACP: true
  acpImage: <registry>/trident-acp:24.02.0
  imagePullSecrets:
  - <secret_name>
```

7. 保存并退出文件。部署过程将自动开始。
8. 验证是否已创建操作员、部署和副本集。

```
kubectl get all -n trident
```

> 在 Kubernetes 集群中只能有 * 一个操作符实例 * 。请勿创建多个部署的Asta三端操作员。

9. 验证 `trident-acp` 容器正在运行 `acpVersion` 为 `24.02.0` 状态为 `Installed`：

```
kubectl get torc -o yaml
```

响应：

```
status:
  acpVersion: 24.02.0
  currentInstallationParams:
    ...
    acpImage: <registry>/trident-acp:24.02.0
    enableACP: "true"
    ...
  ...
  status: Installed
```

Tridentctl

1. "下载Asta三端安装程序并解压缩"。
2. "如果您已有Asta Trident、请从托管它的集群中将其卸载"。
3. 在启用Asta Control配置程序的情况下安装Asta Trent (`--enable-acp=true`）：

```
./tridentctl -n trident install --enable-acp=true --acp
-image=mycustomregistry/trident-acp:24.02
```

4. 确认已启用Asta Control配置程序：

```
./tridentctl -n trident version
```

响应：

```
+----------------+----------------+-------------+ | SERVER VERSION |
CLIENT VERSION | ACP VERSION | +----------------+----------------
+-------------+ | 24.02.0 | 24.02.0 | 24.02.0. | +----------------
+----------------+-------------+
```

掌舵

1. 如果您安装了Astra Trident 23.07.1或更早版本、 "卸载" 操作员和其他组件。
2. 如果您的Kubornetes集群运行的是1.24或更早版本、请删除PSP：

```
kubectl delete psp tridentoperatorpod
```

3. 添加Astra Trident Helm存储库：

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-
chart
```

4. 更新Helm图表：

```
helm repo update netapp-trident
```

响应：

```
Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "netapp-trident" chart
repository
Update Complete. □Happy Helming!□
```

5. 列出图像：

```
./tridentctl images -n trident
```

响应：

```
| v1.28.0            | netapp/trident:24.02.0|
|                    | docker.io/netapp/trident-autosupport:24.02|
|                    | registry.k8s.io/sig-storage/csi-
provisioner:v4.0.0|
|                    | registry.k8s.io/sig-storage/csi-
attacher:v4.5.0|
|                    | registry.k8s.io/sig-storage/csi-
resizer:v1.9.3|
|                    | registry.k8s.io/sig-storage/csi-
snapshotter:v6.3.3|
|                    | registry.k8s.io/sig-storage/csi-node-driver-
registrar:v2.10.0 |
|                    | netapp/trident-operator:24.02.0 (optional)
```

6. 确保提供了三项运算符24.02.0：

```
helm search repo netapp-trident/trident-operator --versions
```

响应：

```
NAME                                CHART VERSION   APP VERSION
DESCRIPTION
netapp-trident/trident-operator 100.2402.0      24.02.0         A
```

7. 使用 ... `helm install` 并运行以下选项之一、其中包括这些设置：
   - 部署位置的名称
   - Astra三端版本
   - Asta Control配置程序映像的名称
   - 用于启用配置程序的标志
   - (可选)本地注册表路径。如果您使用的是本地注册表、则为 " {\f270 {\f151 {\f270} " 可以位于一个注册表或不同的注册表中、但所有CSI映像都必须位于同一注册表中。
   - 三端名称空间

选项

- 没有注册表的映像

```
helm install trident netapp-trident/trident-operator --version
100.2402.0 --set acpImage=cr.astra.netapp.io/astra/trident-acp:24.02.0
--set enableACP=true --set operatorImage=netapp/trident-
operator:24.02.0 --set
tridentAutosupportImage=docker.io/netapp/trident-autosupport:24.02
--set tridentImage=netapp/trident:24.02.0 --namespace trident
```

- 一个或多个注册表中的图像

```
helm install trident netapp-trident/trident-operator --version
100.2402.0 --set acpImage=<your-registry>:<acp image> --set
enableACP=true --set imageRegistry=<your-registry>/sig-storage --set
operatorImage=netapp/trident-operator:24.02.0 --set
tridentAutosupportImage=docker.io/netapp/trident-autosupport:24.02
--set tridentImage=netapp/trident:24.02.0 --namespace trident
```

您可以使用 `helm list` 查看安装详细信息、例如名称、命名空间、图表、状态、应用程序版本、和修订版号。

如果您在使用Helm部署TRIdent时遇到任何问题、请运行此命令以完全卸载Asta TRIdent：

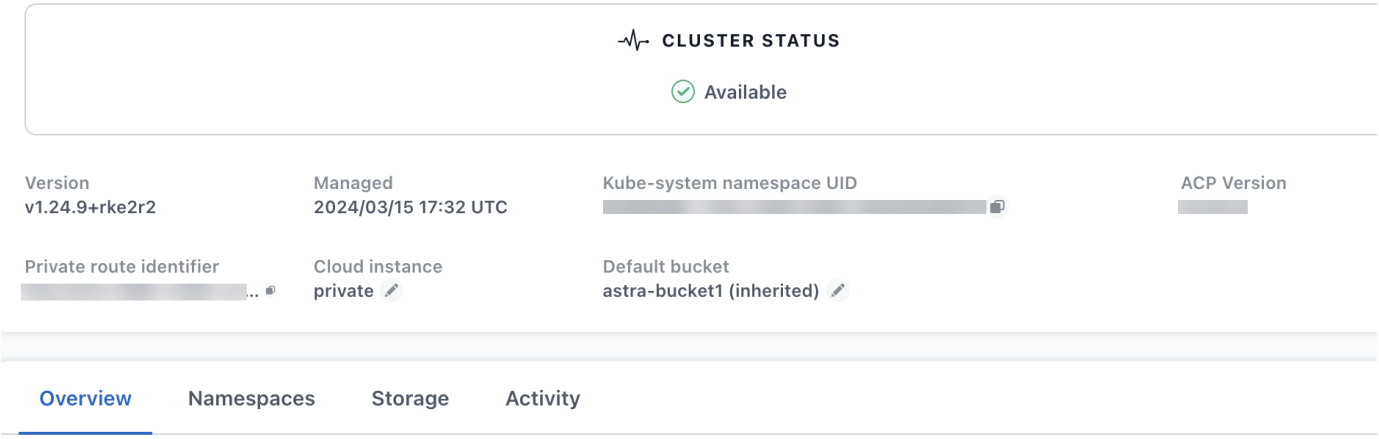
```
./tridentctl uninstall -n trident
```

请勿 "完全删除Asta Trdent CRD" 在尝试重新启用Astra Control配置程序之前、作为卸载的一部分。

结果

Asta Control配置程序功能已启用、您可以使用当前运行的版本可用的任何功能。

(仅适用于Asta Control Center用户)安装Asta Control配置程序后、在Asta Control Center UI中托管此配置程序的集群将显示 `ACP version` 而不是 `Trident version` 字段和当前安装的版本号。



有关详细信息 ...

- "Asta Trdent升级文档"

使用**Astra Control**准备用于集群管理的环境

在添加集群之前、应确保满足以下前提条件。此外、您还应运行资格检查、以确保您的集群已准备好添加到Astra Control Center、并根据需要创建kubeconfig"集群角色。

Astra Control允许您根据环境和首选项添加由自定义资源(Custom Resource、CR)或kubeconfig"管理的集群。

开始之前

- 满足环境前提条件：您的环境满足 "操作环境要求" A作用 控制中心。
- 配置工作节点：确保您 "配置工作节点" 在集群中使用适当的存储驱动程序、以便Pod可以与后端存储进行交互。
- 启用**PSA**限制：如果集群启用了POD安全准入强制(这是Kubernetes 1.25及更高版本集群的标准配置)、则需要对以下名称空间启用PSA限制：
  - `netapp-acc-operator` 命名空间：

    ```
    kubectl label --overwrite ns netapp-acc-operator pod-
    security.kubernetes.io/enforce=privileged
    ```

  - `netapp monitoring` 命名空间：

```
kubectl label --overwrite ns netapp-monitoring pod-
security.kubernetes.io/enforce=privileged
```

- * ONTAP 凭据*：您需要在备用ONTAP 系统上设置ONTAP 凭据以及超级用户和用户ID、以便使用Astra控制中心备份和还原应用程序。

  在ONTAP 命令行中运行以下命令：

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

- *kubeconfig-managed cluster requirement *：这些要求特定于由kubeconfig-managed的应用程序集群。
  - 使**kubeconfig***可访问：您可以访问 "默认集群kubeconfig" 那 "您在安装期间配置的"。
  - 证书颁发机构注意事项：如果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 "Asta Control配置程序高级存储功能"。在未来版本中、如果Asta Control配置程序未启用、则Asta Control将不支持Asta Trent。
  - 配置存储后端：必须至少有一个存储后端 "已在Asta Trdent中配置" 在集群上。
  - 配置存储类：必须至少有一个存储类 "已在Asta Trdent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Asta三端到功能。

运行资格检查

运行以下资格检查，以确保您的集群已准备好添加到 Astra 控制中心。

步骤

1. 确定您正在运行的Astra三项目标版本：

```
kubectl get tridentversion -n trident
```

如果存在Asta三项功能、您将看到类似于以下内容的输出：

```
NAME      VERSION
trident   24.02.0
```

如果Astra三端存储不存在、则会显示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Astra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 Trdent 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. 确保所有Pod (包括 `trident-acp`)正在运行：

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

4. 确定存储类是否正在使用受支持的Asta三端驱动程序。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
```

创建集群角色kubeconfig

对于使用kubeconfig"管理的集群、您可以选择为Astra Control Center创建有限权限或扩展权限管理员角色。这不是Astra控制中心设置所需的操作步骤、因为您已在中配置了kubeconfig "安装过程"。

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Asta 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：

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

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

- 要使用活动环境的集群管理员权限管理的集群的活动kubeconfig

步骤

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

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

```yaml
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>
```

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

# Get, List, Create, and Update all resources
# Necessary to backup and restore all resources in an app
- apiGroups:
  - '*'
  resources:
  - '*'
  verbs:
  - get
  - list
  - create
  - patch

# Delete Resources
# Necessary for in-place restore and AppMirror failover
- apiGroups:
  - ""
  - apps
  - autoscaling
  - batch
  - crd.projectcalico.org
  - extensions
  - networking.k8s.io
  - policy
  - rbac.authorization.k8s.io
  - snapshot.storage.k8s.io
  - trident.netapp.io
  resources:
  - configmaps
  - cronjobs
  - daemonsets
  - deployments
```

```
    - horizontalpodautoscalers
    - ingresses
    - jobs
    - namespaces
    - networkpolicies
    - persistentvolumeclaims
    - poddisruptionbudgets
    - pods
    - podtemplates
    - replicasets
    - replicationcontrollers
    - replicationcontrollers/scale
    - rolebindings
    - roles
    - secrets
    - serviceaccounts
    - services
    - statefulsets
    - tridentmirrorrelationships
    - tridentsnapshotinfos
    - volumesnapshots
    - volumesnapshotcontents
    verbs:
    - delete

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

  # Update resources
  - apiGroups:
    - ""
    - build.openshift.io
    - image.openshift.io
    resources:
    - builds/details
    - replicationcontrollers
    - replicationcontrollers/scale
    - imagestreams/layers
```

```
  - imagestreamtags
  - imagetags
  verbs:
  - update
```

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Kubbernetes示例。有关特定于您的环境的说明、请参见Kubennetes分发版的文档。

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

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

```yaml
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. 创建一个 `ClusterRoleBindingm` 文件，该文件名为 `astracontrol – clusterrolebind.YAML` 。

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

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

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，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-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=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
```

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
```

(技术预览)为受管集群安装Asta Connector

由Asta Control Center管理的集群使用Asta Connector在受管集群和Asta Control Center之间实现通信。您需要在要管理的所有集群上安装Astra Connector。

安装A作用 连接器

您可以使用Kubbernetes命令和自定义资源(Custom Resource、CR)文件安装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Asta 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Asta Control进行管理的集群上安装Asta 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/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Asta 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Asta Control中分配此集群的名称。
   - <ASTRA_CONTROL_URL>：Asta Control的Web UI URL。例如：

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

```yaml
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。

添加集群

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

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

开始之前

- 在添加集群之前，请查看并执行必要的操作 "前提条件任务"。
- 如果您使用的是ONTAP SAN驱动程序、请确保在所有Kubbernetes集群上启用了多路径。

步骤

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

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

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

   > 您应选择一个存储类、该存储类在Astra控件配置程序中进行配置、并由ONTAP存储提供支持。

6. 查看相关信息、如果一切正常、请选择*添加*。

结果

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

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

在ONTAP存储后端启用身份验证

Astra控制中心提供了两种对ONTAP 后端进行身份验证的模式：

- 基于凭据的身份验证：具有所需权限的ONTAP 用户的用户名和密码。您应使用预定义的安全登录角色(如admin或vsadmin)、以确保与ONTAP 版本的最大兼容性。
- 基于证书的身份验证：Astra控制中心还可以使用后端安装的证书与ONTAP 集群进行通信。您应使用客户端证书、密钥和可信CA证书(如果使用)(建议)。

您可以稍后更新现有后端、以便从一种身份验证类型迁移到另一种身份验证方法。一次仅支持一种身份验证方法。

启用基于凭据的身份验证

ASRA控制中心需要集群范围的凭据 `admin` 与ONTAP 后端通信。您应使用标准的预定义角色、例如 `admin`。这样可以确保与未来的ONTAP 版本向前兼容、这些版本可能会公开功能API、以供未来的Astra控制中心版本使用。

> 可以创建自定义安全登录角色并将其用于Astra Control Center、但不建议这样做。

示例后端定义如下所示：

```
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "username": "admin",
  "password": "secret"
}
```

后端定义是以纯文本格式存储凭据的唯一位置。创建或更新后端是唯一需要了解凭据的步骤。因此、这是一项仅由管理员执行的操作、由Kubornetes或存储管理员执行。

启用基于证书的身份验证

Astra控制中心可以使用证书与新的和现有的ONTAP 后端进行通信。您应在后端定义中输入以下信息。

- `clientCertificate`：客户端证书。
- `clientPrivateKey`:关联的私钥。
- `trustedCACertificate`：可信CA证书。如果使用可信 CA ，则必须提供此参数。如果不使用可信 CA ，则可以忽略此设置。

您可以使用以下类型的证书之一：

- 自签名证书
- 第三方证书

使用自签名证书启用身份验证

典型的工作流包括以下步骤。

步骤

1. 生成客户端证书和密钥。生成时、请将公用名(Common Name、CN)设置为ONTAP 用户、以进行身份验证。

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=<common-name>"
```

2. 安装类型为的客户端证书 `client-ca` 和键ONTAP。

```
security certificate install -type client-ca -cert-name <certificate-
name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

3. 确认ONTAP 安全登录角色支持证书身份验证方法。

```
security login create -user-or-group-name vsadmin -application ontapi
-authentication-method cert -vserver <vserver-name>
security login create -user-or-group-name vsadmin -application http
-authentication-method cert -vserver <vserver-name>
```

4. 使用生成的证书测试身份验证。将<SVM ManagementLIF> and <vserver name> 替换为管理LIF IP和ONTAP 名称。您必须确保LIF的服务策略设置为 `default-data-management`。

```
curl -X POST -Lk https://<ONTAP-Management-
LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key
--cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp
xmlns=http://www.netapp.com/filer/admin version="1.21" vfiler="<vserver-
name>"><vserver-get></vserver-get></netapp>
```

5. 使用上一步中获得的值、在Astra Control Center UI中添加存储后端。

使用第三方证书启用身份验证

如果您拥有第三方证书、则可以使用以下步骤设置基于证书的身份验证。

步骤

1. 生成私钥和CSR：

```
openssl req -new -newkey rsa:4096 -nodes -sha256 -subj "/" -outform pem
-out ontap_cert_request.csr -keyout ontap_cert_request.key -addext
"subjectAltName = DNS:<ONTAP_CLUSTER_FQDN_NAME>,IP:<ONTAP_MGMT_IP>”
```

2. 将CSR传递到Windows CA (第三方CA)、然后问题描述 签名证书。
3. 下载签名证书并将其命名为`ONTAP signed_cert.crt
4. 从Windows CA (第三方CA)导出根证书。
5. 为此文件命名 `ca_root.crt`

   现在、您已有以下三个文件：

   - 私钥： `ontap_signed_request.key` (这是ONTAP 中服务器证书对应的密钥。安装服务器证书时需要此证书。)
   - 签名证书： `ontap_signed_cert.crt` (在ONTAP 中也称为_server certificATE _。)
   - 根**CA**证书： `ca_root.crt` (在ONTAP 中也称为_server-ca certific存在_。)
6. 在ONTAP 中安装这些证书。生成并安装 `server` 和 `server-ca` ONTAP 上的证书。

展开**SAMPLE.YAML**

```
# Copy the contents of ca_root.crt and use it here.

security certificate install -type server-ca

Please enter Certificate: Press <Enter> when done

-----BEGIN CERTIFICATE-----
<certificate details>
-----END CERTIFICATE-----


You should keep a copy of the CA-signed digital certificate for
future reference.

The installed certificate's CA and serial number for reference:

CA:
serial:

The certificate's generated name for reference:


===

# Copy the contents of ontap_signed_cert.crt and use it here. For
key, use the contents of ontap_cert_request.key file.
security certificate install -type server
Please enter Certificate: Press <Enter> when done

-----BEGIN CERTIFICATE-----
<certificate details>
-----END CERTIFICATE-----

Please enter Private Key: Press <Enter> when done

-----BEGIN PRIVATE KEY-----
<private key details>
-----END PRIVATE KEY-----

Enter certificates of certification authorities (CA) which form the
certificate chain of the server certificate. This starts with the
issuing CA certificate of the server certificate and can range up to
the root CA certificate.
Do you want to continue entering root and/or intermediate
```

```
certificates {y|n}: n

The provided certificate does not have a common name in the subject
field.
Enter a valid common name to continue installation of the
certificate: <ONTAP_CLUSTER_FQDN_NAME>

You should keep a copy of the private key and the CA-signed digital
certificate for future reference.
The installed certificate's CA and serial number for reference:
CA:
serial:
The certificate's generated name for reference:


==
# Modify the vserver settings to enable SSL for the installed
certificate

ssl modify -vserver <vserver_name> -ca <CA>  -server-enabled true
-serial <serial number>       (security ssl modify)

==
# Verify if the certificate works fine:

openssl s_client -CAfile ca_root.crt -showcerts -servername server
-connect <ONTAP_CLUSTER_FQDN_NAME>:443
CONNECTED(00000005)
depth=1 DC = local, DC = umca, CN = <CA>
verify return:1
depth=0
verify return:1
write W BLOCK
---
Certificate chain
0 s:
   i:/DC=local/DC=umca/<CA>

-----BEGIN CERTIFICATE-----
<Certificate details>
```

7. 为同一主机创建客户端证书、以实现无密码通信。Asta控制中心使用此过程与ONTAP 进行通信。
8. 在ONTAP 上生成并安装客户端证书：

展开**SAMPLE.YAML**

```
# Use /CN=admin or use some other account which has privileges.
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout
ontap_test_client.key -out ontap_test_client.pem -subj "/CN=admin"

Copy the content of ontap_test_client.pem file and use it in the
below command:
security certificate install -type client-ca -vserver <vserver_name>

Please enter Certificate: Press <Enter> when done

-----BEGIN CERTIFICATE-----
<Certificate details>
-----END CERTIFICATE-----

You should keep a copy of the CA-signed digital certificate for
future reference.
The installed certificate's CA and serial number for reference:

CA:
serial:
The certificate's generated name for reference:


==

ssl modify -vserver <vserver_name> -client-enabled true
(security ssl modify)

# Setting permissions for certificates
security login create -user-or-group-name admin -application ontapi
-authentication-method cert -role admin -vserver <vserver_name>

security login create -user-or-group-name admin -application http
-authentication-method cert -role admin -vserver <vserver_name>

==

#Verify passwordless communication works fine with the use of only
certificates:

curl --cacert ontap_signed_cert.crt  --key ontap_test_client.key
--cert ontap_test_client.pem
https://<ONTAP_CLUSTER_FQDN_NAME>/api/storage/aggregates
{
```

```
"records": [
{
"uuid": "f84e0a9b-e72f-4431-88c4-4bf5378b41bd",
"name": "<aggr_name>",
"node": {
"uuid": "7835876c-3484-11ed-97bb-d039ea50375c",
"name": "<node_name>",
"_links": {
"self": {
"href": "/api/cluster/nodes/7835876c-3484-11ed-97bb-d039ea50375c"
}
}
},
"_links": {
"self": {
"href": "/api/storage/aggregates/f84e0a9b-e72f-4431-88c4-
4bf5378b41bd"
}
}
}
],
"num_records": 1,
"_links": {
"self": {
"href": "/api/storage/aggregates"
}
}
}%
```


9. 在Asta Control Center UI中添加存储后端、并提供以下值：
   - 客户端证书：ONATP_TEST_client.prom
   - 私钥：ontap_test_client.key
   - 可信**CA**证书：ONATP_signed_cert.crt

添加存储后端

设置凭据或证书身份验证信息后、您可以将现有ONTAP 存储后端添加到Astra控制中心以管理其资源。

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

如果启用了Astra Control配置程序、则在使用NetApp SnapMirror技术时、可以选择在Astra控制中心中添加和管理ONTAP存储后端。

步骤

1. 从左侧导航区域的信息板中、选择*后端*。
2. 选择 * 添加 * 。
3. 在添加存储后端页面的使用现有部分中，选择* ONTAP *。
4. 选择以下选项之一：
   - 使用管理员凭据：输入ONTAP 集群管理IP地址和管理员凭据。凭据必须是集群范围的凭据。

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

   - 使用证书：上传证书 `.pem` file、证书密钥 `.key` 文件、以及证书颁发机构文件(可选)。
5. 选择 * 下一步 * 。
6. 确认后端详细信息并选择 * 管理 * 。

结果

后端将显示在中 `online` 包含摘要信息的列表中的状态。

您可能需要刷新页面才能显示后端。

添加存储分段

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

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

开始之前

- 确保您有一个可从Astra Control Center管理的集群访问的存储分段。
- 确保您具有此存储分段的凭据。
- 确保存储分段为以下类型之一：
  - NetApp ONTAP S3
  - NetApp StorageGRID S3
  - Microsoft Azure
  - 通用 S3

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

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

步骤

1. 在左侧导航区域中，选择 * 桶 * 。
2. 选择 * 添加 * 。
3. 选择存储分段类型。

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

4. 输入现有存储分段名称和可选的问题描述。

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

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

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

概念

架构和组件

Asta Control是一款Kubennet应用程序数据生命周期管理解决方案、可简化有状态应用程序的操作、并帮助您在混合和多云环境之间存储、保护和移动Kubennet工作负载。

功能

Astra Control 为 Kubernetes 应用程序数据生命周期管理提供了关键功能：

存储：

- 为容器化工作负载动态配置存储
- 对从容器到永久性卷的数据进行传输中加密
- 跨区域、跨区域复制

保护：

- 自动发现整个应用程序及其数据并提供应用程序感知型保护
- 根据组织需求从任何Snapshot版本即时恢复应用程序
- 跨区域、区域和云提供商实现快速故障转移

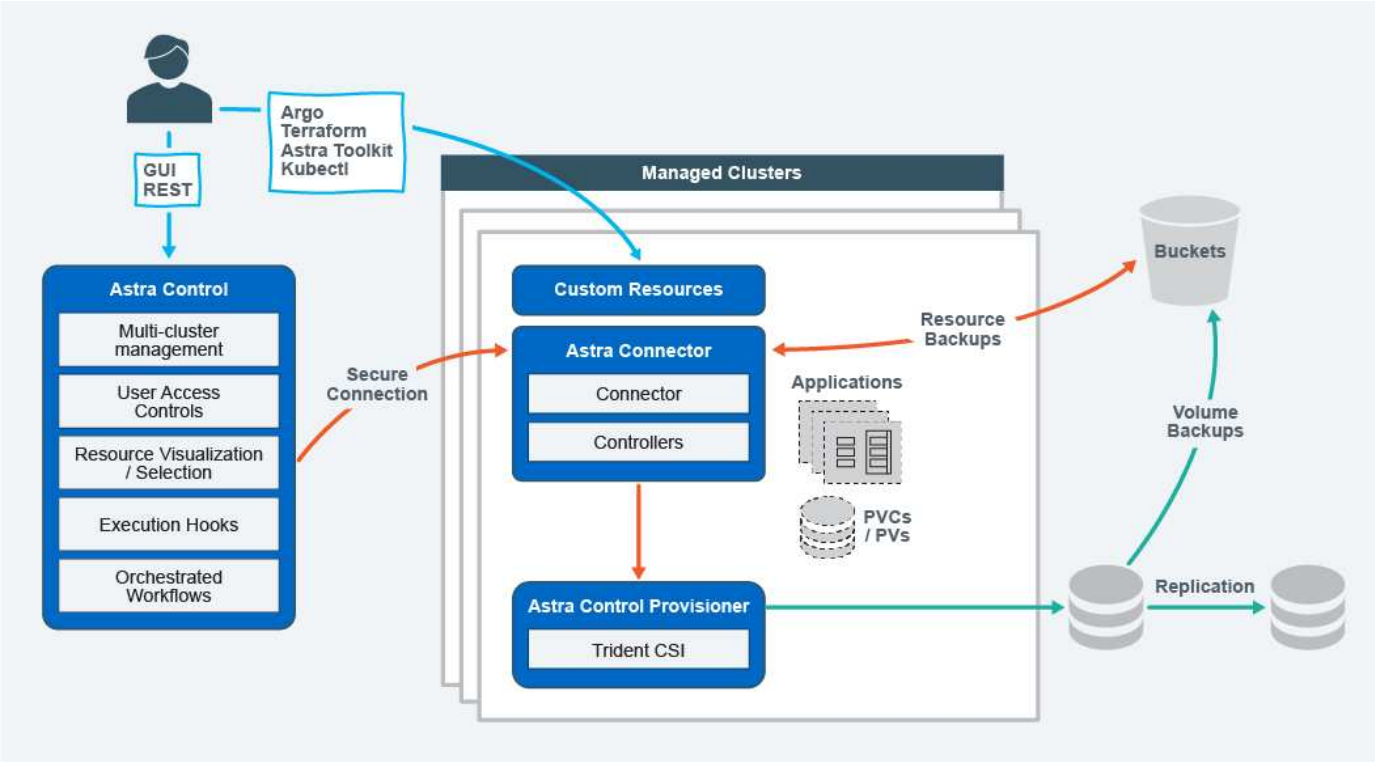
移动：

- 在Kubbernetes集群和云之间实现全面的应用程序和数据移动
- 即时克隆整个应用程序和数据
- 通过一致的Web UI和API一键迁移应用程序

架构

Asta Control的架构支持IT提供高级数据管理功能、以增强Kubbernetes应用程序的功能和可用性、简化容器化工作负载在公有云和内部环境之间的管理、保护和移动。 并通过REST API和SDK提供自动化功能、支持编程访问、以便与现有工作流无缝集成。

Astra Control是Kub联网 原生版本、支持利用自定义资源的数据保护工作流、同时保持与现有API和SDK的向后兼容性。Kubornetes原生数据保护具有显著优势；通过与Kubornetes API和资源无缝集成、数据保护可以通过组织的现有CI/CD和/或GitOps工具成为应用程序生命周期的固有组成部分。

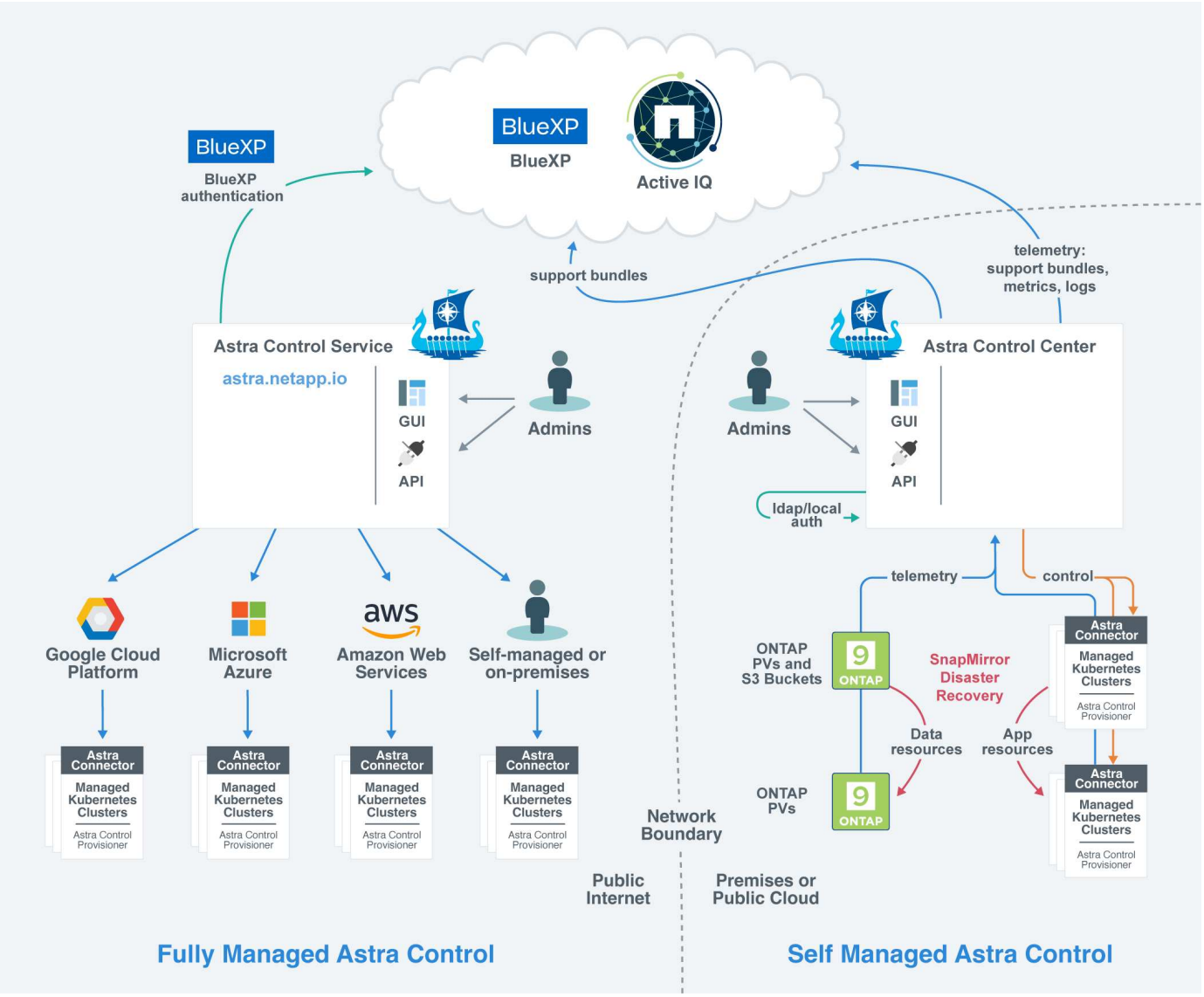


Astra Control基于四个互补组件构建：

- **Astra Control**：Astra Control是适用于所有托管集群的集中式管理服务，可提供协调的工作负载，以实现云和内部环境中的应用程序保护和移动性以及以下功能：
  - 多个集群和云的组合视图
  - 保护协调一致的工作流
  - 精细的资源可视化和选择
- **Astra Connecter**：Astra Connector与Astra Control相结合、可提供与每个受管集群的安全连接、无论连接状态如何、均可在本地执行计划的操作、并具有以下功能：
  - 本地执行计划的操作、而不管连接状态如何
  - 在集群之间分布和优化A作用 的系统资源使用的本地操作
  - 本地安装、允许对集群进行最低权限访问以提高安全性
- **Astra Control**置备程序：Astra Control置备程序提供核心CSI配置功能和高级存储管理功能，以增加安全性和灾难恢复配置，以及以下功能：
  - 为容器化工作负载动态配置存储
  - 高级存储管理：
    - 对从容器到PV的数据进行传输中加密
    - SnapMirror Cloud功能支持跨区域、跨区域复制
- **Astra**自定义资源：每个集群上使用的自定义资源提供了一种Kubernetes本机方法来在本地运行操作、简化了与其他Kubernetes友好工具和自动化的集成、并提供了以下功能：
  - 直接的生态系统工具集成和自动化工作流
  - 用于启用自定义工作流的较低级别的基本功能

部署模式

Asta Control有两种部署模式。



- * Astra Control Service*：NetApp管理的服务、可为多个云提供商环境中的Kubernetes集群以及自管理Kubernetes集群提供应用程序感知型数据管理。

  "Astra Control Service 文档"

- * Astra Control Center*：自管理软件，可为内部环境中运行的 Kubernetes 集群提供应用程序感知型数据管理。Astra控制中心还可以安装在多个云提供商环境中、并具有一个NetApp Cloud Volumes ONTAP 存储后端。

  "Astra 控制中心文档"

	Astra 控制服务	**Astra** 控制中心
如何提供?	作为 NetApp 提供的一项完全托管的云服务	作为可下载、安装和管理的软件

	Astra 控制服务	**Astra** 控制中心
它托管在何处?	基于 NetApp 选择的公有云	在您自己的Kubernetes集群上
如何更新?	由 NetApp 管理	您可以管理任何更新
支持哪些**Kubednetes**分发版?	• 云提供商 ◦ Amazon Web Services ▪ Amazon Elelic Kubelnetes Service (EKS) ◦ Google Cloud ▪ Google Kubernetes Engine （GKEE） ◦ Microsoft Azure ▪ Azure Kubernetes Service （AKS） • 自管理集群 ◦ Kubnetes (上游) ◦ Rancher Kubernetes Engine （RKE） ◦ Red Hat OpenShift 容器平台 • 内部集群 ◦ Red Hat OpenShift容器平台内部部署	• 基于Azure堆栈HCI的Azure Kubnetes Service • Google Anthos • Kubnetes (上游) • Rancher Kubernetes Engine （RKE） • Red Hat OpenShift 容器平台

	Astra 控制服务	Astra 控制中心
支持哪些存储后端?	• 云提供商 ◦ Amazon Web Services ▪ Amazon EBS ▪ 适用于 NetApp ONTAP 的 Amazon FSX ▪ "Cloud Volumes ONTAP" ◦ Google Cloud ▪ Google 持久磁盘 ▪ NetApp Cloud Volumes Service ▪ "Cloud Volumes ONTAP" ◦ Microsoft Azure ▪ Azure受管磁盘 ▪ Azure NetApp Files ▪ "Cloud Volumes ONTAP" • 自管理集群 ◦ Amazon EBS ◦ Azure受管磁盘 ◦ Google 持久磁盘 ◦ "Cloud Volumes ONTAP" ◦ NetApp MetroCluster ◦ "Longhorn" • 内部集群 ◦ NetApp MetroCluster ◦ NetApp ONTAP AFF 和 FAS 系统 ◦ NetApp ONTAP Select ◦ "Cloud Volumes ONTAP" ◦ "Longhorn"	• NetApp ONTAP AFF 和 FAS 系统 • NetApp ONTAP Select • "Cloud Volumes ONTAP" • "Longhorn"

有关详细信息 ...

- "Astra Control Service 文档"
- "Astra 控制中心文档"
- "Astra Trident 文档"
- "Astra Control API"
- "Cloud Insights 文档"

- "ONTAP 文档"

数据保护

了解 Astra 控制中心提供的数据保护类型，以及如何以最佳方式使用它们来保护您的应用程序。

快照，备份和保护策略

快照和备份均可保护以下类型的数据：

- 应用程序本身
- 与应用程序关联的任何永久性数据卷
- 属于应用程序的任何资源项目

snapshot 是应用程序的时间点副本，它与应用程序存储在同一个已配置卷上。通常速度较快。您可以使用本地快照将应用程序还原到较早的时间点。快照对于快速克隆很有用；快照包括应用程序的所有 Kubernetes 对象，包括配置文件。快照对于克隆或还原同一集群中的应用程序非常有用。

_backup_基于快照。它存储在外部对象存储中、因此、与本地快照相比、创建速度可能会较慢。您可以将应用程序备份还原到同一集群，也可以通过将应用程序备份还原到其他集群来迁移应用程序。您还可以选择较长的备份保留期限。由于备份存储在外部对象存储中，因此在发生服务器故障或数据丢失时，备份通常比快照提供更好的保护。

保护策略 _ 是一种通过根据您为应用程序定义的计划自动创建快照和 / 或备份来保护应用程序的方法。此外、您还可以通过保护策略选择要在计划中保留多少个快照和备份、并设置不同的计划粒度级别。使用保护策略自动执行备份和快照是确保每个应用程序根据组织的需求和服务级别协议(Service Level Agreement、SLA)要求进行保护的最佳方式。

> *You can't be Fully protected until you have a recent backup* 。这一点非常重要，因为备份存储在对象存储中，而不是永久性卷。如果发生故障或意外事件会擦除集群及其关联的永久性存储，则需要备份才能恢复。快照无法让您恢复。

不可配置的备份

不可变备份是指在指定时间段内无法更改或删除的备份。在创建不可更改的备份时、Astra Control会检查以确保您使用的存储分段是一次写入多次读取(Write on时 读取多次、WORM)存储分段、如果是、则会确保备份在Astra Control中不可更改。
Astra Control Center支持使用以下平台和存储分段类型创建不可配置的备份：

- Amazon Web Services使用配置了S3对象锁定的Amazon S3存储分段
- 使用配置了S3对象锁定的S3存储分段的NetApp StorageGRID

使用不可配置备份时、请注意以下事项：

- 如果备份到不受支持的平台中的WORM存储分段或备份到不受支持的存储分段类型、则可能会出现无法预测的结果、例如、即使已过保留时间、备份删除也会失败。
- Astra Control不支持数据生命周期管理策略、也不支持手动删除用于不可变备份的存储分段上的对象。确保存储后端未配置为管理Astra Control快照或备份数据的生命周期。

克隆

_cloner_是应用程序、其配置及其永久性数据卷的精确副本。您可以在同一个 Kubernetes 集群或另一个集群上手动创建克隆。如果需要将应用程序和存储从一个 Kubernetes 集群移动到另一个 Kubernetes 集群，则克隆应用程序非常有用。

在存储后端之间进行复制

使用Astra Control、您可以使用NetApp SnapMirror技术的异步复制功能、以低RPO (恢复点目标)和低RTO (恢复时间目标)为应用程序构建业务连续性。配置后、应用程序便可将数据和应用程序更改从一个存储后端复制到另一个存储后端、复制到同一集群上或复制到不同集群之间。

您可以在同一ONTAP集群或不同ONTAP集群上的两个ONTAP SVM之间进行复制。

Astra Control会将应用程序Snapshot副本异步复制到目标集群。复制过程包括SnapMirror复制的永久性卷中的数据以及受Astra Control保护的应用程序元数据。

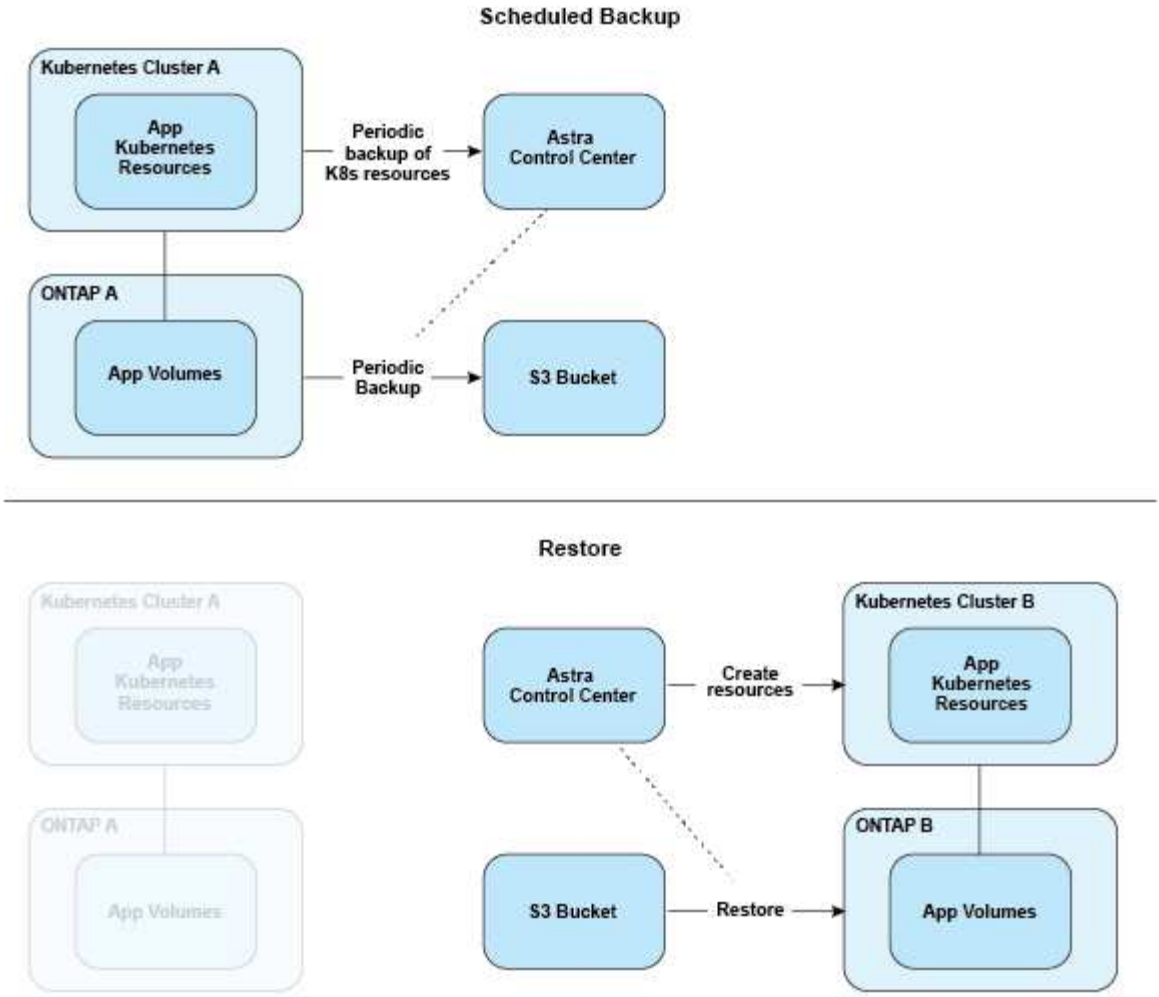
应用程序复制与应用程序备份和还原在以下方面有所不同：

- 应用程序复制：Astra Control要求源和目标Kubernetes集群(可以是同一集群)可用并进行管理、并将其各自的ONTAP存储后端配置为启用NetApp SnapMirror。Asta Control创建策略驱动型应用程序快照并将其复制到目标存储后端。NetApp SnapMirror技术用于复制永久性卷数据。要进行故障转移、Astra Control可以在目标Kubernetes集群上重新创建应用程序对象、并在目标ONTAP 集群上创建复制的卷、从而使复制的应用程序联机。由于目标ONTAP集群上已存在永久性卷数据、因此Astra Control可以为故障转移提供快速恢复时间。
- 应用程序备份和还原：备份应用程序时、Astra Control会创建应用程序数据的快照并将其存储在对象存储分段中。需要还原时、必须将存储分段中的数据复制到ONTAP 集群上的永久性卷。备份/还原操作不要求二级Kubernetes或ONTAP集群可用并进行管理、但额外的数据复制可能会导致还原时间较长。

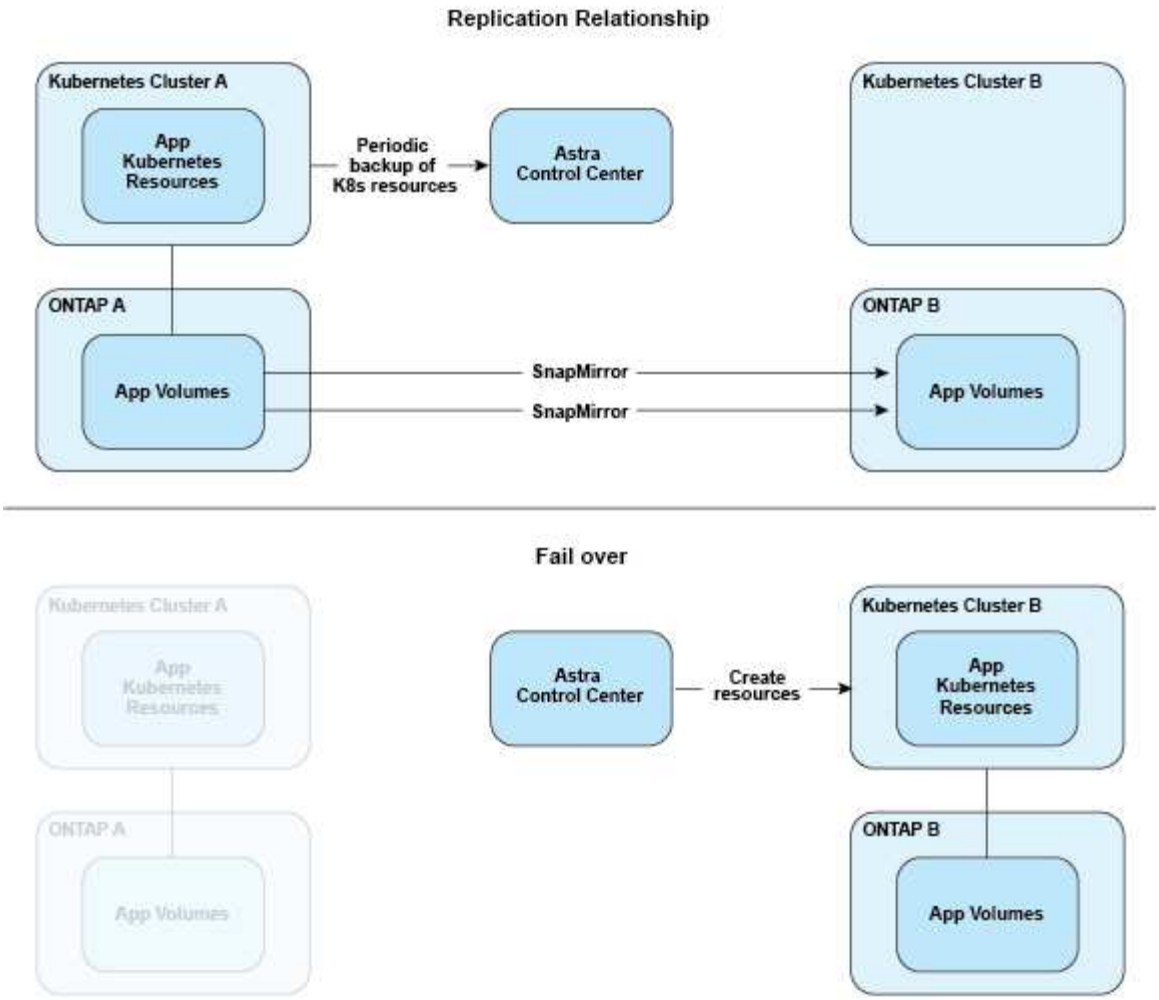
要了解如何复制应用程序、请参见 "使用SnapMirror技术将应用程序复制到远程系统"。

下图显示了计划的备份和还原过程与复制过程的对比情况。

备份过程会将数据复制到S3存储分段、并从S3存储分段进行还原：



另一方面、复制是通过复制到ONTAP来完成的、然后故障转移会创建Kubrenetes资源：



许可证已过期的备份、快照和克隆

如果许可证过期、只有当要添加或保护的应用程序是另一个Asta Control Center实例时、您才能添加新应用程序或执行应用程序保护操作(例如快照、备份、克隆和还原操作)。

许可

在部署Astra Control Center时、它会安装一个嵌入式90天评估版许可证、可用于4、800个CPU单元。如果您需要更多容量或更长的评估期、或者要升级到完整许可证、则可以从NetApp获得不同的评估许可证或完整许可证。

您可以通过以下方式之一获取许可证：

- 如果您正在评估Astra Control Center、并且需要与嵌入式评估许可证中包含的评估条款不同的评估条款、请与NetApp联系以申请不同的评估许可证文件。
- "如果您已购买Astra Control Center、请生成NetApp许可证文件(NLF)" 登录到NetApp 支持站点 并导航到"Systems"(系统)菜单下的软件许可证。

有关ONTAP 存储后端所需许可证的详细信息、请参见 "支持的存储后端"。

请确保您的许可证至少启用所需数量的CPU单元。如果Astra Control Center当前管理的CPU单元数超过所应用新许可证中的可用CPU单元数、您将无法应用新许可证。

评估版许可证和完全许可证

新安装的Astra Control Center会提供嵌入式评估许可证。评估版许可证可实现与完整许可证相同的功能和特性、有效期为90天。评估期结束后、需要完整许可证才能继续执行完整功能。

许可证到期

如果活动A作用 中的Astra Control Center许可证过期、则以下功能的UI和API功能将不可用：

- 手动创建本地快照和备份
- 计划本地快照和备份
- 从快照或备份还原
- 从快照或当前状态克隆
- 管理新应用程序
- 配置复制策略

如何计算许可证使用量

在将新集群添加到 Astra 控制中心时，只有在集群上运行的至少一个应用程序由 Astra 控制中心管理之后，该集群才会计入已用许可证。

开始管理集群上的应用程序时、该集群的所有CPU单元都会计入Astra Control Center许可证使用量中、但使用标签报告的Red Hat OpenShift集群节点CPU单元除外 `node-role.kubernetes.io/infra: ""`。

Red Hat OpenShift基础架构节点不使用Astra Control Center中的许可证。要将节点标记为基础架构节点、请应用此标签 `node-role.kubernetes.io/infra: ""` 连接到节点。

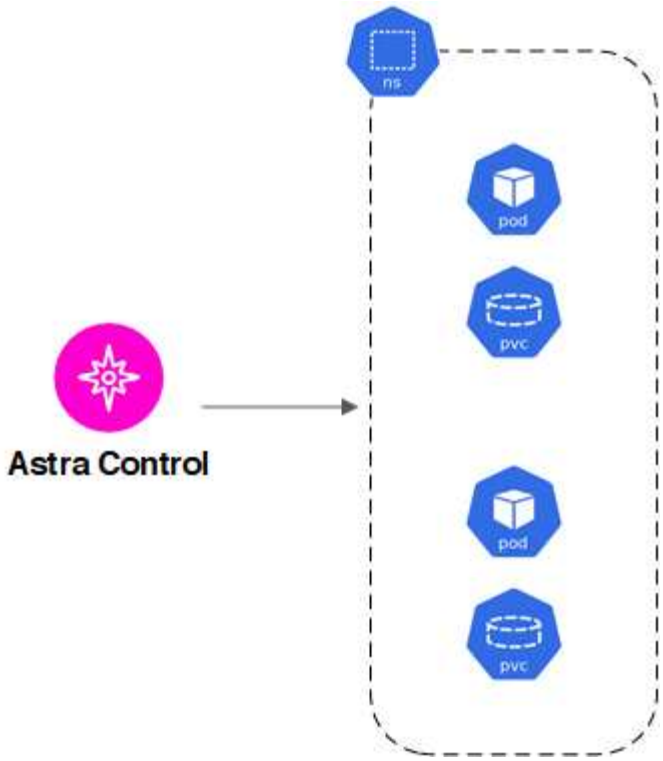
了解更多信息

- "首次设置Astra控制中心时添加许可证"
- "更新现有许可证"

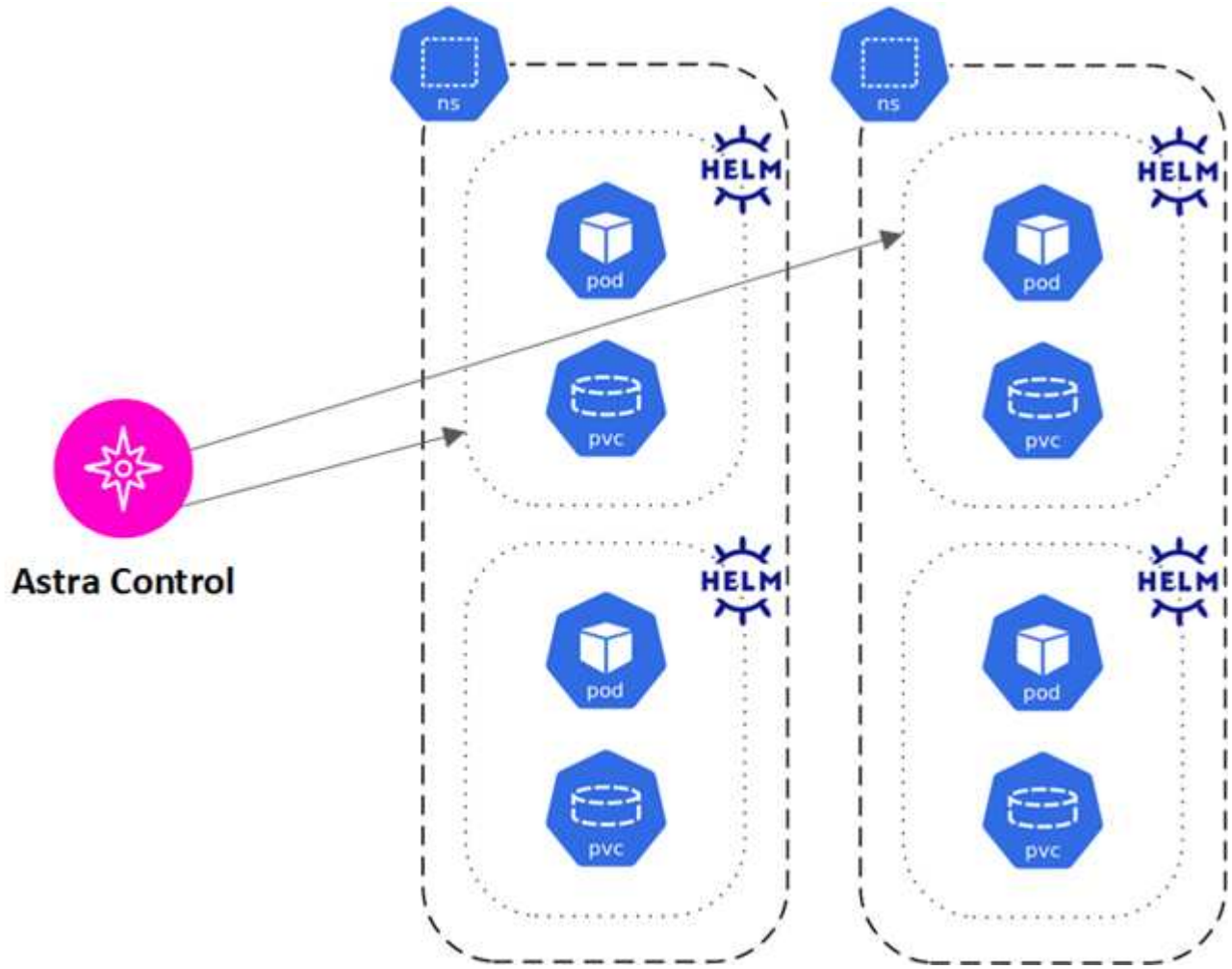
应用程序管理

当Astra Control发现集群时、这些集群上的应用程序将不受管理、直到您选择要如何管理它们为止。Astra Control 中的受管应用程序可以是以下任一项：

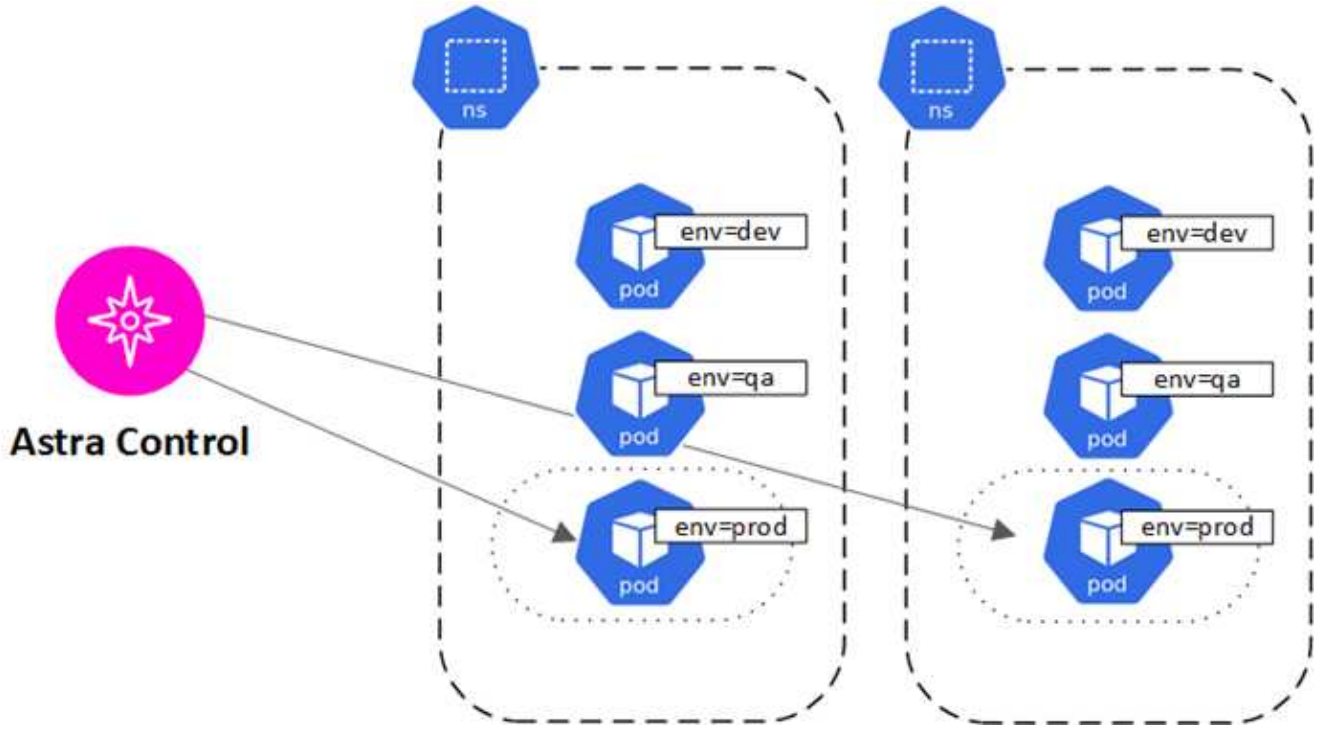
- 命名空间，包括该命名空间中的所有资源



- 部署在一个或多个命名空间中的单个应用程序(在此示例中使用了helm3)



- 一组通过一个或多个命名空间中的Kubernetes标签标识的资源



存储类和永久性卷大小

Astra控制中心支持NetApp ONTAP和Longhorn作为存储后端。

概述

Astra 控制中心支持以下功能：

- * ONTAP存储支持的存储类*：如果使用ONTAP后端、Astra控制中心可以导入ONTAP后端以报告监控信息。
- *Longhorn*支持的基于CSI的存储类：可以将Longhorn与Longhorn容器存储接口(CSI)驱动程序结合使用。

> 存储类应为 "已配置" 使用Astra Control配置程序。

存储类

将集群添加到Astra控制中心时、系统会提示您选择该集群上先前配置的一个存储类作为默认存储类。如果在永久性卷请求（ PVC ）中未指定存储类，则会使用此存储类。可以随时在 Astra 控制中心内更改默认存储类，也可以随时通过在 PVC 或 Helm 图表中指定存储类的名称来使用任何存储类。确保您仅为 Kubernetes 集群定义了一个默认存储类。

用户角色和命名空间

了解 Astra Control 中的用户角色和命名空间，以及如何使用它们控制对组织中资源的访问。

用户角色

您可以使用角色控制用户对 Astra Control 资源或功能的访问权限。以下是 Astra Control 中的用户角色：

- * 查看器 * 可以查看资源。
- " 成员 * " 具有 " 查看器 " 角色权限，可以管理应用程序和集群，取消管理应用程序以及删除快照和备份。
- * 管理员 * 具有成员角色权限，可以添加和删除除所有者之外的任何其他用户。
- * 所有者 * 具有管理员角色权限，可以添加和删除任何用户帐户。

您可以向 " 成员 " 或 " 查看器 " 用户添加限制，以将用户限制为一个或多个 [命名空间]。

命名空间

命名空间是指您可以分配给由 Astra Control 管理的集群中的特定资源的范围。将集群添加到 Astra Control 时，Astra Control 会发现集群的命名空间。发现后，可以将命名空间作为约束分配给用户。只有有权访问该命名空间的成员才能使用该资源。您可以使用命名空间来控制对资源的访问，方法是采用对您的组织有意义的模式；例如，按公司内的物理区域或部门进行访问。向用户添加约束时，您可以将该用户配置为可以访问所有命名空间或仅访问一组特定命名空间。您还可以使用命名空间标签分配命名空间约束。

了解更多信息

"管理本地用户和角色"

使用 Astra 控制中心

开始管理应用程序

你先请 "将集群添加到 Astra Control 管理中"、您可以在集群上安装应用程序(在Astra Control之外)、然后转到Astra Control中的应用程序页面来定义应用程序及其资源。

您可以定义和管理包含存储资源且运行Pod的应用程序、或者包含存储资源且未运行任何Pod的应用程序。没有运行Pod的应用程序称为纯数据应用程序。

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 Control Center管理应用程序，您需要嵌入式Astra Control Center评估许可证或完整许可证。
- 命名空间：可以使用Astra Control在单个集群上的一个或多个指定命名空间内定义应用程序。一个应用程序可以包含跨越同一集群中多个命名空间的资源。Astra Control不支持在多个集群之间定义应用程序。
- 存储类：如果您安装的应用程序明确设置了存储类、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存储类。将具有显式设置的存储类的应用程序克隆到没有相同存储类的集群将失败。
- * 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	ConfigMap
CronJob	CustomResourceDefinition	CustomResource
DaemonSet	DeploymentConfig	HorizontalPodAutoscaler
Ingress	MutatingWebhook	NetworkPolicy
PersistentVolumeClaim	Pod	PodDisruptionBudget
PodTemplate	ReplicaSet	Role
RoleBinding	Route	Secret
Service	ServiceAccount	StatefulSet
ValidatingWebhook		

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操作符安装的应用程序，这些操作符通常采用"传递值"而不是"传递参考"架构设计。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。

  以下是一些遵循这些模式的操作员应用程序：

  - "Apache K8ssandra"

    > 对于 K8ssandra ，支持原位还原操作。要对新命名空间或集群执行还原操作，需要关闭应用程序的原始实例。这是为了确保传输的对等组信息不会导致跨实例通信。不支持克隆应用程序。

  - "Jenkins CI"
  - "Percona XtraDB 集群"

  Astra Control可能无法克隆使用"按参考传递"架构设计的运算符(例如CockroachDB运算符)。在这些类型的克隆操作期间，克隆的操作员会尝试引用源操作员提供的 Kubernetes 机密，尽管在克隆过程中他们拥有自己的新机密。克隆操作可能会失败，因为 Astra Control 不知道源运算符中的 Kubernetes 密钥。

在集群上安装应用程序

你先请 "已添加集群" 对于Astra Control、您可以在集群上安装应用程序或管理现有应用程序。可以管理范围限定为一个或多个命名空间的任何应用程序。

定义应用程序

在Astra Control发现集群上的命名空间后、您可以定义要管理的应用程序。您可以选择 管理跨越一个或多个命名空间的应用程序 或 将整个命名空间作为一个应用程序进行管理。这一切都可以细化到数据保护操作所需的粒度级别。

虽然您可以使用Astra Control单独管理层次结构的两个级别(命名空间和该命名空间中的应用程序或跨命名空间)、但最佳做法是选择一个或另一个。如果在命名空间和应用程序级别同时执行操作，则在 Astra Control 中执行的操作可能会失败。

> 例如、您可能希望为"Maria"设置一个每周节奏的备份策略、但您可能需要比该策略更频繁地备份"MariaDB"(位于同一命名空间中)。根据这些需求、您需要单独管理这些应用程序、而不是作为单命名空间应用程序来管理。

开始之前

- 已将Kubernetes集群添加到Astra Control中。
- 集群上安装的一个或多个应用程序。 阅读有关支持的应用程序安装方法的更多信息。
- 已添加到Astra Control的Kubernetes集群上的现有命名空间。
- (可选) Any上的Kubernetes标签 "支持的Kubernetes资源"。

> 标签是一个键 / 值对，您可以将其分配给 Kubernetes 对象进行标识。通过标签，可以更轻松地对 Kubernetes 对象进行排序，组织和查找。要了解有关 Kubernetes 标签的更多信息， "请参见 Kubernetes 官方文档"。

关于此任务

- 开始之前、您还应了解相关信息 "管理标准命名空间和系统命名空间"。
- 如果您计划在Astra Control中对应用程序使用多个命名空间、 "修改具有命名空间限制的用户角色" 升级到支持多命名空间的Astra Control Center版本后。
- 有关如何使用 Astra Control API 管理应用程序的说明，请参见 "Astra Automation 和 API 信息"。

应用程序管理选项

- [定义要作为应用程序进行管理的资源]
- [定义要作为应用程序进行管理的命名空间]
- "(技术预览)使用Kubnetes自定义资源定义应用程序"

定义要作为应用程序进行管理的资源

您可以指定 "构成应用程序的Kubernetes资源" 要使用Astra Control进行管理的。通过定义应用程序、您可以将Kubernetes集群中的元素分组到一个应用程序中。此Kubernetes资源集合按命名空间和标签选择器标准进行组织。

通过定义应用程序、您可以更精细地控制要包含在Astra Control操作中的内容、包括克隆、快照和备份。

> 定义应用程序时、请确保不在具有保护策略的多个应用程序中包含Kubernetes资源。Kubernetes资源上重叠的保护策略可能会发生发生原因 数据冲突。 阅读示例中的更多内容。

展开以了解有关将集群范围的资源添加到应用程序的更多信息。

除了自动包含的Astra Control之外、您还可以导入与命名空间资源关联的集群资源。您可以添加一个规则、该规则将包含特定组的资源、种类、版本以及标签(可选)。如果存在Astra Control不会自动包含的资源、您可能需要执行此操作。

您不能排除Astra Control自动包含的任何集群范围的资源。

您可以添加以下内容 `apiVersions` (这些组与API版本结合使用)：

资源种类	**apiVersions (组+版本)**
`ClusterRole`	rbac.authorization.k8s.io/v1
`ClusterRoleBinding`	rbac.authorization.k8s.io/v1
`CustomResource`	apiextensions.k8s.io/v1、apiextensions.k8s.io/v1bea1
`CustomResourceDefinition`	apiextensions.k8s.io/v1、apiextensions.k8s.io/v1bea1
`MutatingWebhookConfiguration`	可批准registration.K8s.IO/v1
`ValidatingWebhookConfiguration`	可批准registration.K8s.IO/v1

步骤

1. 从应用程序页面中、选择*定义*。
2. 在*定义应用程序*窗口中、输入应用程序名称。
3. 在*集群*下拉列表中选择运行应用程序的集群。
4. 从*命名空间*下拉列表中为应用程序选择一个命名空间。

   可以使用Astra Control在单个集群上的一个或多个指定命名空间中定义应用程序。一个应用程序可以包含跨越同一集群中多个命名空间的资源。Astra Control不支持在多个集群之间定义应用程序。

5. (可选)为每个命名空间中的Kubernetes资源输入一个标签。您可以指定单个标签或标签选择器条件(查询)。

   要了解有关 Kubernetes 标签的更多信息， "请参见 Kubernetes 官方文档"。

6. (可选)通过选择*添加命名空间*并从下拉列表中选择命名空间来为应用程序添加其他命名空间。
7. (可选)为您添加的任何其他命名空间输入单个标签或标签选择器条件。
8. (可选)要在Astra Control自动包含的资源之外还包括集群范围的资源、请选中*包括其他集群范围的资源*并完成以下操作：
   a. 选择*添加包含规则*。
   b. 组：从下拉列表中、选择API资源组。
   c. 种类：从下拉列表中、选择对象架构的名称。

d. 版本：输入API版本。

e. 标签选择器：也可以包括要添加到规则中的标签。此标签仅用于检索与此标签匹配的资源。如果不提供标签、则Astra Control将收集为该集群指定的所有资源类型的实例。

f. 查看根据条目创建的规则。

g. 选择 * 添加 * 。

> 您可以根据需要创建任意数量的集群范围资源规则。这些规则将显示在"定义应用程序摘要"中。

9. 选择 * 定义 * 。
10. 选择*定义*后、根据需要对其他应用程序重复此过程。

定义完应用程序后、该应用程序将显示在中 `Healthy` 在应用程序页面上的应用程序列表中的状态。现在、您可以克隆它并创建备份和快照。

> 您刚刚添加的应用程序在 " 受保护 " 列下可能会显示一个警告图标，表示它尚未备份，并且尚未计划备份。

> 要查看特定应用程序的详细信息，请选择应用程序名称。

要查看添加到此应用程序的资源、请选择*资源*选项卡。在资源列中选择资源名称后面的数字、或者在搜索中输入资源名称、以查看包含的其他集群范围资源。

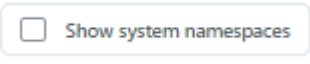
定义要作为应用程序进行管理的命名空间

您可以通过将命名空间的资源定义为应用程序来将命名空间中的所有Kubernetes资源添加到Astra Control管理中。如果您要以类似的方式并以通用间隔管理和保护特定命名空间中的所有资源、则此方法比单独定义应用程序更好。

步骤

1. 从集群页面中、选择一个集群。
2. 选择*命名空间*选项卡。
3. 选择包含要管理的应用程序资源的命名空间的"Actions"菜单、然后选择*定义为应用程序*。

> 如果要定义多个应用程序、请从命名空间列表中进行选择、然后选择左上角的*操作*按钮并选择*定义为应用程序*。这将在各个命名空间中定义多个单独的应用程序。有关多命名空间应用程序、请参见 [定义要作为应用程序进行管理的资源]。

> 选中*显示系统命名空间*复选框以显示默认情况下在应用程序管理中不使用的系统命名空间。
> 
>
> "阅读更多内容"。

此过程完成后、与此命名空间关联的应用程序将显示在`Associated applications`列中。

[技术预览]使用Kubernetes自定义资源定义应用程序

您可以通过使用自定义资源(CR)将要使用Asta Control管理的Kubbernetes资源定义为应用程序来指定这些资源。如果要单独管理某个命名空间中的资源、或者要以类似方式并按相同间隔管理和保护某个特定命名空间中的所有资源、则可以添加集群范围的资源。

步骤

1. 创建自定义资源(CR)文件并将其命名为(例如、 `astra_mysql_app.yaml`）。
2. 在中命名应用程序 `metadata.name`。
3. 定义要管理的应用程序资源：

spec.includedClusterScopedResources

除了Asta Control自动包含的资源类型之外、还包括集群范围的资源类型：

- * spec.includedClusterScopedResources*:_(可选)_要包含的集群范围资源类型列表。
  - **groupVersion Kind**：_(可选)_明确标识一种类型。
    - **group**：_(如果使用groupVersion Kind、则为必需项)_要包含的资源的API组。
    - 版本：_(如果使用groupVersionKind、则为必需项)_要包含的资源的API版本。
    - **kind**：_(如果使用groupVersion Kind、则为必填项)_要包含的资源种类。
  - **labelSelecter**：_(可选)_一组资源的标签查询。它仅用于检索与标签匹配的资源。如果不提供标签、则Astra Control将收集为该集群指定的所有资源类型的实例。将对“对等标签”和“对等显示”的结果进行AND运算。
    - 匹配标签：_(可选)_个｛key、value｝对的映射。在匹配标签映射中，单个{key,value}相当于m匹 配表达式的元素，该元素的键字段为“key”，运算符为“in”，值数组仅包含“value”。这些要求是AND。
    - 不符合要求：_(可选)_标签选择器要求列表。这些要求是AND。
      - **key**:_(如果使用了“对等表达式”，则为必需项)_与标签选择器关联的标签键。
      - 运算符：_(如果使用的是对等表达式则为必需项)_表示键与一组值的关系。有效运算符为 `In`， `NotIn`， `Exists` 和 `DoesNotExist`。
      - **values**：_(如果使用的是匹配备用)_字符串值数组。如果运算符为 `In` 或 `NotIn`，值数组必须为空。如果运算符为 `Exists` 或 `DoesNotExist`，值数组必须为空。

spec.includedNamespaces

在应用程序中的这些资源中包括名称和资源：

- 。**spec.includedNamespaces**:_(必需)_用于定义命名空间和可选的资源选择筛选器。
  - 命名空间：_(必需)_包含要使用Astra Control管理的应用程序资源的命名空间。
  - **labelSelecter**：_(可选)_一组资源的标签查询。它仅用于检索与标签匹配的资源。如果不提供标签、则Astra Control将收集为该集群指定的所有资源类型的实例。将对“对等标签”和“对等显示”的结果进行AND运算。
    - 匹配标签：_(可选)_个｛key、value｝对的映射。在匹配标签映射中，单个{key,value}相当于m匹 配表达式的元素，该元素的键字段为“key”，运算符为“in”，值数组仅包含“value”。这些要求是AND。
    - 不符合要求：_(可选)_标签选择器要求列表。 `key` 和 `operator` 为必填项。这些要求是AND。
      - **key**:_(如果使用了“对等表达式”，则为必需项)_与标签选择器关联的标签键。
      - 运算符：_(如果使用的是对等表达式则为必需项)_表示键与一组值的关系。有效运算符为 `In`， `NotIn`， `Exists` 和 `DoesNotExist`。
      - 值：_(如果使用匹配备则为必需项)_字符串值数组。如果运算符为 `In` 或 `NotIn`，值数组必须为空。如果运算符为 `Exists` 或 `DoesNotExist`，值数组必须为空。

YAML示例：

```yaml
apiVersion: astra.netapp.io/v1
kind: Application
metadata:
  name: astra_mysql_app
spec:
  includedNamespaces:
    - namespace: astra_mysql_app
      labelSelector:
        matchLabels:
          app: nginx
          env: production
        matchExpressions:
          - key: tier
            operator: In
            values:
              - frontend
              - backend
```

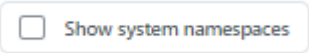
4. 在您填充之后 `astra_mysql_app.yaml` 使用正确值的文件、应用CR：

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

系统命名空间如何？

Astra Control还会发现Kubernetes集群上的系统命名空间。默认情况下、我们不会向您显示这些系统命名空间、因为您很少需要备份系统应用程序资源。

通过选中*显示系统命名空间*复选框、您可以从选定集群的命名空间选项卡中显示系统命名空间。



> Astra Control Center默认情况下不会显示为您可以管理的应用程序、但您可以使用另一个Astra Control Center实例备份和还原Astra Control Center实例。

示例：不同版本的单独保护策略

在此示例中、DevOps团队正在管理"金丝利"版本部署。该团队的集群中有三个Pod运行nginx。其中两个 Pod 专用于稳定版本。第三个 POD 适用于加那利版本。

DevOps 团队的 Kubernetes 管理员会将标签 `detion=stable` 添加到稳定版本 Pod 中。该团队会将标签 `deeption=Canary` 添加到 Canary 版本 POD 中。

该团队的稳定版本要求每小时创建一次快照，每天进行备份。金那利版本的发布时间较短，因此他们希望为任何标记为 `deeption=Canary` 的对象创建一个不太积极的短期保护策略。

为了避免可能发生的数据冲突、管理员将创建两个应用程序：一个用于"加那利"版本、一个用于"稳定"版本。这样就可以使两组 Kubernetes 对象的备份，快照和克隆操作分开。

了解更多信息

- "使用 Astra Control API"
- "取消管理应用程序"

保护应用程序

保护概述

您可以使用 Astra 控制中心为应用程序创建备份，克隆，快照和保护策略。备份应用程序可帮助您的服务和关联数据尽可能地可用；在灾难情形下，从备份还原可以确保应用程序及其关联数据的完全恢复，而不会造成任何中断。备份，克隆和快照有助于防止常见威胁，例如勒索软件，意外数据丢失和环境灾难。 "了解 Astra 控制中心提供的数据保护类型以及何时使用"。

此外、您还可以将应用程序复制到远程集群、以便为灾难恢复做好准备。

应用程序保护工作流

您可以使用以下示例工作流开始保护应用程序。

[一个] 保护所有应用程序

要确保您的应用程序立即受到保护， "为所有应用程序创建手动备份"。

[两个] 为每个应用程序配置一个保护策略

要自动执行未来备份和快照， "为每个应用程序配置一个保护策略"。例如，您可以从每周备份和每日快照开始，这两种备份均保留一个月。强烈建议使用保护策略自动执行备份和快照，而不是手动备份和快照。

[三个] 调整保护策略

随着应用程序及其使用模式的变化，根据需要调整保护策略以提供最佳保护。

[四个] 将应用程序复制到远程集群

"复制应用程序" 使用NetApp SnapMirror技术连接到远程集群。Astra Control可将快照复制到远程集群、从而提供异步灾难恢复功能。

[五个] 发生灾难时、请使用最新备份或复制功能将应用程序还原到远程系统

如果发生数据丢失，您可以通过进行恢复 "还原最新备份" 每个应用程序的第一个。然后，您可以还原最新的快照（如果可用）。或者、您也可以使用复制到远程系统。

通过快照和备份保护应用程序

通过使用自动保护策略或临时创建快照和备份来保护所有应用程序。您可以使用Astra控制中心UI或 "Astra Control API" 保护应用程序。

关于此任务

- * Helm部署的应用程序*：如果您使用Helm部署应用程序、则Astra控制中心需要Helm版本3。完全支持管理和克隆使用 Helm 3 部署的应用程序（或从 Helm 2 升级到 Helm 3 ）。不支持使用 Helm 2 部署的应用程序。
- **(**仅限**OpenShift**集群**)**添加策略：创建用于在OpenShift集群上托管应用程序的项目时、系统会为该项目(或Kubernetes命名空间)分配SecurityContext UID。要使 Astra 控制中心能够保护您的应用程序并将应用程序移动到 OpenShift 中的其他集群或项目，您需要添加策略，使应用程序能够作为任何 UID 运行。例如，以下 OpenShift 命令行界面命令会为 WordPress 应用程序授予相应的策略。

  ```
  oc new-project WordPress oc adm policy add-SCS-to-group anyuid system ：
  serviceaccounts ： WordPress oc adm policy add-SCS-to-user privileged -z
  default -n WordPress
  ```

您可以执行以下与保护应用程序数据相关的任务：

- [配置保护策略]
- [创建快照]
- [创建备份]
- 为ONTAP NAS经济型操作启用备份和还原
- [创建不可还原的备份]
- [查看快照和备份]
- [删除快照]
- [取消备份]
- [删除备份]

配置保护策略

保护策略通过按定义的计划创建快照，备份或这两者来保护应用程序。您可以选择每小时，每天，每周和每月创建快照和备份，并且可以指定要保留的副本数。您可以使用Astra Control Web UI或自定义资源(CR)文件定义保护策略。

如果您需要备份或快照的运行频率高于每小时一次，则可以 "使用 Astra Control REST API 创建快照和备份"。

> 如果要定义一个保护策略来创建不可变备份以写入一次多次读取(Write 1机会 读取、WORM)分段、请确保备份的保留时间不短于为分段配置的保留期限。

> 偏移备份和复制计划以避免计划重叠。例如、在每小时的前几个小时执行备份、并计划复制、以5分钟的偏移和10分钟的间隔开始。

使用**Web UI**配置保护策略

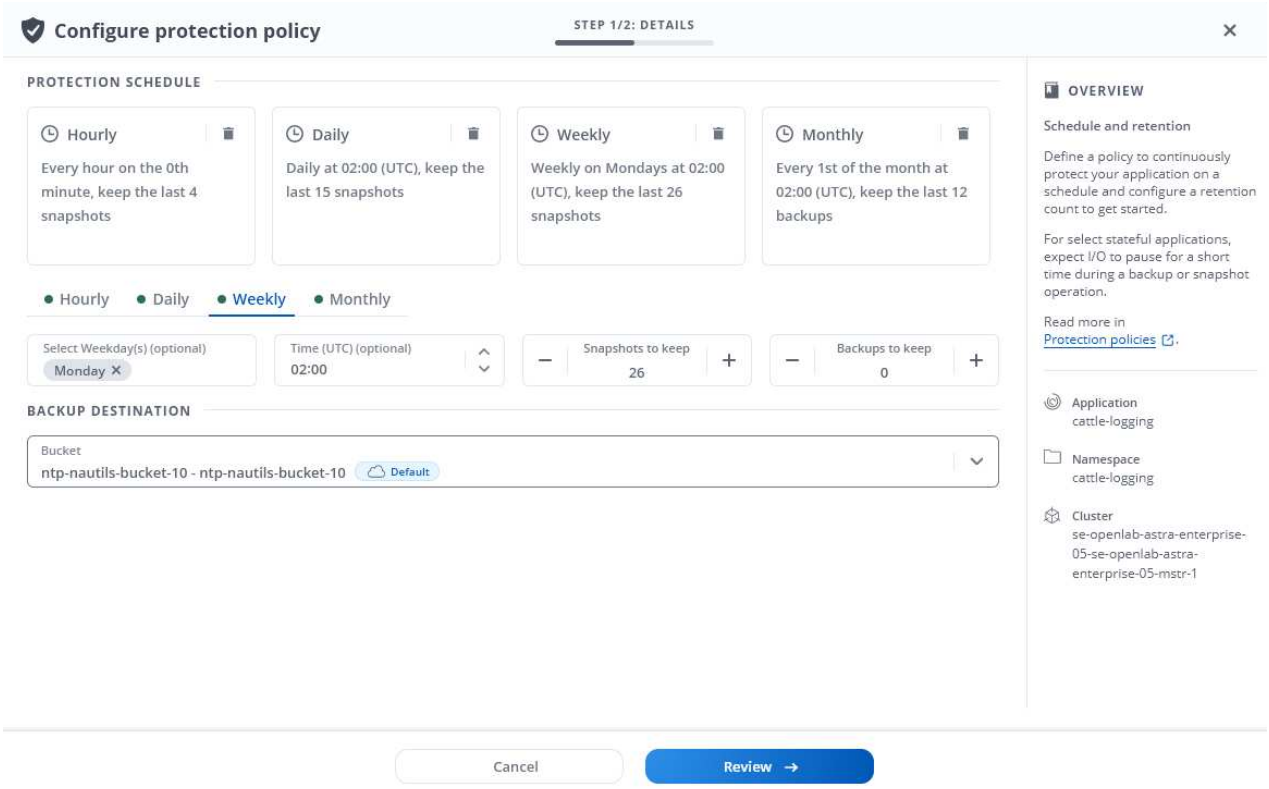
步骤

1. 选择 * 应用程序 * ，然后选择应用程序的名称。
2. 选择 * 数据保护 * 。
3. 选择 * 配置保护策略 * 。
4. 通过选择每小时，每天，每周和每月保留的快照和备份数量来定义保护计划。

   您可以同时定义每小时，每天，每周和每月计划。在设置保留级别之前，计划不会变为活动状态。

   在为备份设置保留级别时，您可以选择要将备份存储到的存储分段。

   以下示例将为快照和备份设置四个保护计划：每小时，每天，每周和每月。



5. [技术预览]从存储分段列表中为备份或快照选择目标分段。
6. 选择 * 审阅 * 。
7. 选择 * 设置保护策略。 *

[技术预览**]**使用**CR**配置保护策略

步骤

1. 创建自定义资源(CR)文件并将其命名为 `astra-control-schedule-cr.yaml`。更新方括号<>中的值、以匹配您的Astra Control环境、集群配置和数据保护需求：
   - <CR_NAME>：此自定义资源的名称；为您的环境选择一个唯一且合理的名称。
   - <APPLICATION_NAME>：要备份的应用程序的KubeNet名称。

- <APPVAULT_NAME>：应存储备份内容的AppVault的名称。
- <BACKUPS_RETAINED>：要保留的备份数。零表示不应创建任何备份。
- <SNAPSHOTS_RETAINED>：要保留的快照数量。零表示不应创建任何快照。
- <GRANULARITY>：计划应运行的频率。可能值以及必需的关联字段：
  - `hourly` (需要您指定 `spec.minute`)
  - `daily` (需要您指定 `spec.minute` 和 `spec.hour`)
  - `weekly` (需要您指定 `spec.minute`， `spec.hour`，和 `spec.dayOfWeek`)
  - `monthly` (需要您指定 `spec.minute`， `spec.hour`，和 `spec.dayOfMonth`)
- <DAY_OF_MONTH>：_(可选)_计划应运行的日期(1 - 31)。如果粒度设置为、则此字段为必填字段 `monthly`。
- <DAY_OF_WEEK>：_(可选)_计划应运行的日期(0 - 7)。值0或7表示星期日。如果粒度设置为、则此字段为必填字段 `weekly`。
- <HOUR_OF_DAY>：_(可选)_计划应运行的时间(0 - 23)。如果粒度设置为、则此字段为必填字段 `daily`， `weekly`或 `monthly`。
- <MINUTE_OF_HOUR>：_(可选)_计划应运行的分钟(0 - 59)。如果粒度设置为、则此字段为必填字段 `hourly`， `daily`， `weekly`或 `monthly`。

```yaml
apiVersion: astra.netapp.io/v1
kind: Schedule
metadata:
  namespace: astra-connector
  name: <CR_NAME>
spec:
  applicationRef: <APPLICATION_NAME>
  appVaultRef: <APPVAULT_NAME>
  backupRetention: "<BACKUPS_RETAINED>"
  snapshotRetention: "<SNAPSHOTS_RETAINED>"
  granularity: <GRANULARITY>
  dayOfMonth: "<DAY_OF_MONTH>"
  dayOfWeek: "<DAY_OF_WEEK>"
  hour: "<HOUR_OF_DAY>"
  minute: "<MINUTE_OF_HOUR>"
```

2. 在您填充之后 `astra-control-schedule-cr.yaml` 使用正确值的文件、应用CR：

```
kubectl apply -f astra-control-schedule-cr.yaml
```

结果

Astra Control 通过使用您定义的计划和保留策略创建和保留快照和备份来实施数据保护策略。

创建快照

您可以随时创建按需快照。

关于此任务

Astra Control支持使用以下驱动程序支持的存储类创建快照：

- `ontap-nas`
- `ontap-san`
- `ontap-san-economy`

> 如果您的应用使用由支持的存储类 `ontap-nas-economy` 驱动程序、无法创建快照。为快照使用备用存储类。

使用**Web UI**创建快照

步骤

1. 选择 * 应用程序 * 。
2. 从所需应用程序的 * 操作 * 列的选项菜单中，选择 * 快照 * 。
3. 自定义快照的名称、然后选择*下一步*。
4. [技术预览]从存储分段列表中选择快照的目标分段。
5. 查看快照摘要并选择 * 快照 * 。

[技术预览**]**使用**CR**创建快照

步骤

1. 创建自定义资源(CR)文件并将其命名为 `astra-control-snapshot-cr.yaml`。更新方括号<>中的值以匹配您的Astra Control环境和集群配置：
   - <CR_NAME>：此自定义资源的名称；为您的环境选择一个唯一且合理的名称。
   - <APPLICATION_NAME>：要创建快照的应用程序的KubeNet名称。
   - <APPVAULT_NAME>：应存储快照内容的AppVault的名称。
   - <RECLAIM_POLICY>：_(可选)_定义删除快照CR时快照会发生什么情况。有效选项：
     - `Retain`
     - `Delete` (默认)

```
apiVersion: astra.netapp.io/v1
kind: Snapshot
metadata:
  namespace: astra-connector
  name: <CR_NAME>
spec:
  applicationRef: <APPLICATION_NAME>
  appVaultRef: <APPVAULT_NAME>
  reclaimPolicy: <RECLAIM_POLICY>
```

2. 在您填充之后 `astra-control-snapshot-cr.yaml` 使用正确值的文件、应用CR：

```
kubectl apply -f astra-control-snapshot-cr.yaml
```

结果

快照过程开始。如果在*数据保护*>*快照*页面的*状态*列中、快照状态为*运行状况*、则快照将成功。

创建备份

您可以随时备份应用程序。

关于此任务

Astra Control中的存储分段不报告可用容量。在备份或克隆Asta Control管理的应用程序之前、请检查相应存储管理系统中的存储分段信息。

如果您的应用使用由支持的存储类 `ontap-nas-economy` 驱动程序、您需要这样做 启用备份和还原 功能。请确保您已定义 `backendType` 中的参数 "Kubbernetes存储对象" 值为 `ontap-nas-economy` 在执行任何保护操作之前。

> Astra Control支持使用以下驱动程序支持的存储类创建备份：
>
> - `ontap-nas`
> - `ontap-nas-economy`
> - `ontap-san`
> - `ontap-san-economy`

使用**Web UI**创建备份

步骤

1. 选择 * 应用程序 * 。
2. 从所需应用程序的*操作*列的选项菜单中、选择*备份*。
3. 自定义备份的名称。
4. 选择是否从现有快照备份应用程序。如果选择此选项，则可以从现有快照列表中进行选择。
5. [技术预览]从存储分段列表中选择备份的目标分段。
6. 选择 * 下一步 * 。
7. 查看备份摘要并选择*备份*。

[技术预览**]**使用**CR**创建备份

步骤

1. 创建自定义资源(CR)文件并将其命名为 `astra-control-backup-cr.yaml`。更新方括号<>中的值以匹配您的Astra Control环境和集群配置：
   - <CR_NAME>：此自定义资源的名称；为您的环境选择一个唯一且合理的名称。
   - <APPLICATION_NAME>：要备份的应用程序的KubeNet名称。
   - <APPVAULT_NAME>：应存储备份内容的AppVault的名称。

```
apiVersion: astra.netapp.io/v1
kind: Backup
metadata:
  namespace: astra-connector
  name: <CR_NAME>
spec:
  applicationRef: <APPLICATION_NAME>
  appVaultRef: <APPVAULT_NAME>
```

2. 在您填充之后 `astra-control-backup-cr.yaml` 使用正确值的文件、应用CR：

```
kubectl apply -f astra-control-backup-cr.yaml
```

结果

Astra Control 会创建应用程序的备份。

- 如果网络发生中断或异常缓慢，备份操作可能会超时。这会导致备份失败。
- 如果需要取消正在运行的备份、请按照中的说明进行操作 [取消备份]。要删除备份、请等待备份完成、然后按照中的说明进行操作 [删除备份]。
- 在执行数据保护操作（克隆，备份，还原）并随后调整永久性卷大小后，在 UI 中显示新卷大小之前，最长会有 20 分钟的延迟。数据保护操作将在几分钟内成功完成，您可以使用存储后端的管理软件确认卷大小的更改。

为**ONTAP NAS**经济型操作启用备份和还原

Asta Control配置程序提供了备份和还原功能、可为使用的存储后端启用这些功能 `ontap-nas-economy` 存储类。

开始之前

- 您已拥有 "已启用Asta Control配置程序"。
- 您已在Astra Control中定义了一个应用程序。在您完成此操作步骤之前、此应用程序的保护功能将受限。
- 您已拥有 `ontap-nas-economy` 已选择作为存储后端的默认存储类。

步骤

1. 在ONTAP存储后端执行以下操作：

   a. 查找托管的SVM `ontap-nas-economy`应用程序的基于卷。

   b. 登录到连接到创建卷的ONTAP的终端。

   c. 隐藏SVM的Snapshot目录：

   此更改会影响整个SVM。隐藏的目录将继续可访问。

```
nfs modify -vserver <svm name> -v3-hide-snapshot enabled
```

+

   验证ONTAP存储后端上的Snapshot目录是否已隐藏。如果未能隐藏此目录、可能会导致无法访问您的应用程序、尤其是在使用NFSv3的情况下。

2. 在Asta Control配置程序中执行以下操作：

   a. 为每个PV启用Snapshot目录 `ontap-nas-economy` 基于并与应用程序关联：

```
tridentctl update volume <pv name> --snapshot-dir=true --pool-level
=true -n trident
```

   b. 确认已为每个关联PV启用Snapshot目录：

```
tridentctl get volume <pv name> -n trident -o yaml | grep snapshotDir
```

响应：

```
snapshotDirectory: "true"
```

3. 在Astra Control中、启用所有关联的快照目录后刷新应用程序、以便Astra Control识别更改后的值。

结果

该应用程序已准备好使用Astra Control进行备份和还原。每个PVC还可供其他应用程序用于备份和恢复。

创建不可还原的备份

只要存储不可变备份的存储分段上的保留策略禁止、就无法修改、删除或覆盖该备份。您可以通过将应用程序备份到配置了保留策略的存储分段来创建不可配置的备份。请参见 "数据保护" 了解有关使用不可配置备份的重要信息。

开始之前

您需要使用保留策略配置目标存储分段。根据您使用的存储提供程序、执行此操作的方式会有所不同。有关详细信息、请参见存储提供程序文档：

- **Amazon Web Services**： "创建存储分段时启用S3对象锁定、并设置默认保留模式"g监管"和默认保留期限"。
- * NetApp StorageGRID *： "创建存储分段时启用S3对象锁定、并将默认保留模式设置为"Compliance (合规性)"和默认保留期限"。

> Astra Control中的存储分段不报告可用容量。在备份或克隆Asta Control管理的应用程序之前、请检查相应存储管理系统中的存储分段信息。

> 如果您的应用使用由支持的存储类 `ontap-nas-economy` 驱动程序、请确保您已定义 `backendType` 中的参数 "Kubbernetes存储对象" 值为 `ontap-nas-economy` 在执行任何保护操作之前。

步骤

1. 选择 * 应用程序 * 。
2. 从所需应用程序的*操作*列的选项菜单中、选择*备份*。
3. 自定义备份的名称。
4. 选择是否从现有快照备份应用程序。如果选择此选项，则可以从现有快照列表中进行选择。
5. 从存储分段列表中为备份选择一个目标分段。一次写入、多次读取(WORM)存储分段的状态在存储分段名称旁边显示为"已锁定"。

> 如果存储分段类型不受支持、则在将鼠标悬停在存储分段上或选择存储分段时会指示此情况。

6. 选择 * 下一步 * 。
7. 查看备份摘要并选择*备份*。

结果

Astra Control可为应用程序创建不可移动的备份。

- 如果网络发生中断或异常缓慢，备份操作可能会超时。这会导致备份失败。
- 如果您尝试同时为同一应用程序创建两个不可变备份到同一存储分段、Astra Control会阻止第二个备份启动。等待第一个备份完成、然后再启动另一个备份。
- 您无法取消正在运行的不可更改备份。
- 在执行数据保护操作（克隆，备份，还原）并随后调整永久性卷大小后，在 UI 中显示新卷大小之前，最长会有 20 分钟的延迟。数据保护操作将在几分钟内成功完成，您可以使用存储后端的管理软件确认卷大小的更改。

查看快照和备份

您可以从数据保护选项卡查看应用程序的快照和备份。

不可还原备份会在其所使用的存储分段旁边显示状态为"已锁定"。

步骤

1. 选择 * 应用程序 * ，然后选择应用程序的名称。
2. 选择 * 数据保护 * 。

   默认情况下会显示快照。

3. 选择 * 备份 * 可查看备份列表。

删除快照

删除不再需要的计划快照或按需快照。

您不能删除当前正在复制的快照。

步骤

1. 选择 * 应用程序 * ，然后选择受管应用程序的名称。
2. 选择 * 数据保护 * 。
3. 从选项菜单的 * 操作 * 列中为所需快照选择 * 删除快照 * 。
4. 键入单词 "delete" 确认删除，然后选择 * 是，删除 snapshot* 。

结果

Astra Control 会删除快照。

取消备份

您可以取消正在进行的备份。

> 要取消备份、备份必须位于中 `Running` 状态。您无法取消中的备份 `Pending` 状态。

> 您无法取消正在运行的不可更改备份。

步骤

1. 选择 * 应用程序 * ，然后选择应用程序的名称。
2. 选择 * 数据保护 * 。
3. 选择 * 备份 * 。
4. 从选项菜单中的*操作*列中为所需备份选择*取消*。
5. 键入单词"cancel"以确认操作、然后选择*是、取消备份*。

删除备份

删除不再需要的计划备份或按需备份。您不能删除对不可更改存储分段所做的备份、除非该存储分段的保留策略允许您这样做。

> 在保留期限到期之前、您不能删除不可更改的备份。

> 如果需要取消正在运行的备份、请按照中的说明进行操作 [取消备份]。要删除备份、请等待备份完成、然后按照以下说明进行操作。

步骤

1. 选择 * 应用程序 * ，然后选择应用程序的名称。
2. 选择 * 数据保护 * 。
3. 选择 * 备份 * 。
4. 从选项菜单的 * 操作 * 列中为所需备份选择 * 删除备份 * 。
5. 键入单词 "delete" 确认删除，然后选择 * 是，删除备份 * 。

结果

Astra Control 会删除备份。

[技术预览]保护整个集群

您可以为集群上的任何或所有非受管卷创建计划的自动备份。这些工作流由NetApp以Kubbernetes服务帐户、角色绑定和cron作业的形式提供、并使用Python脚本进行编排。

工作原理

在配置和安装完整集群备份工作流时、cron作业会定期运行、并保护尚未管理的任何命名空间、从而根据您在安

装期间选择的计划自动创建保护策略。

如果您不希望使用完整集群备份工作流保护集群上的每个非受管命名空间、则可以改用基于标签的备份工作流。基于标签的备份工作流也会使用cron任务、但它不会保护所有非受管命名库、而是通过您提供的标签来标识命名库、以根据铜牌、银牌或金牌备份策略保护命名库。

在所选工作流范围内创建新命名空间时、该命名空间会自动受到保护、无需任何管理员操作。这些工作流是按集群实施的、因此不同的集群可以根据集群的重要性使用任一工作流、并具有独特的保护级别。

示例：完全集群保护

例如、在配置和安装完整集群备份工作流时、任何命名空间中的任何应用程序都将定期进行管理和保护、而无需管理员进一步努力。安装工作流时、命名空间不必存在；如果将来添加命名空间、它将受到保护。

示例：基于标签的保护

要获得更精细的粒度、您可以使用基于标签的工作流。例如、您可以安装此工作流、并告诉用户根据所需的保护级别、将多个标签之一应用于要保护的任何命名区域。这样、用户就可以使用其中一个标签创建命名空间、而无需通知管理员。它们的新命名空间以及其中的所有应用程序都会自动受到保护。

为所有的名段创建计划备份

您可以使用完整集群备份工作流为集群上的所有名空间创建计划备份。

步骤

1. 将以下文件下载到可通过网络访问集群的计算机：
    - "Components.YAML CRD文件"
    - "protectCluster.py Python脚本"
2. 要配置和安装此工具包、请执行以下步骤： "按照附带的说明进行操作"。

为特定的名段创建计划备份

您可以使用基于标签的备份工作流按标签为特定命名库创建计划备份。

步骤

1. 将以下文件下载到可通过网络访问集群的计算机：
    - "Components.YAML CRD文件"
    - "protectCluster.py Python脚本"
2. 要配置和安装此工具包、请执行以下步骤： "按照附带的说明进行操作"。

还原应用程序

Astra Control 可以从快照或备份还原应用程序。将应用程序还原到同一集群时，从现有快照进行还原的速度会更快。您可以使用 Astra Control UI 或 "Astra Control API" 还原应用程序。

开始之前

- 首先保护您的应用程序：强烈建议您在恢复应用程序之前为其创建快照或备份。这样、如果还原失败、您就可以从快照或备份克隆。

- 检查目标卷：如果要还原到其他存储类、请确保该存储类使用相同的永久性卷访问模式(例如ReadWriteMany)。如果目标永久性卷访问模式不同，还原操作将失败。例如、如果源永久性卷使用rwx访问模式、请选择无法提供rwx的目标存储类、例如Azure托管磁盘、AWS EBS、Google持久磁盘或 `ontap-san`，发生原因则还原操作是否会失败。有关永久性卷访问模式的详细信息、请参阅 "Kubernetes" 文档。
- 规划空间需求：对使用NetApp ONTAP 存储的应用程序执行原位还原时、还原的应用程序使用的空间可能会增加一倍。执行原位还原后、从还原的应用程序中删除所有不需要的快照以释放存储空间。
- **(**仅限**Red Hat OpenShift**集群**)**添加策略：创建用于在OpenShift集群上托管应用程序的项目时、系统会为该项目(或Kubernetes命名空间)分配SecurityContext UID。要使 Astra 控制中心能够保护您的应用程序并将应用程序移动到 OpenShift 中的其他集群或项目，您需要添加策略，使应用程序能够作为任何 UID 运行。例如，以下 OpenShift 命令行界面命令会为 WordPress 应用程序授予相应的策略。

  ```
  oc new-project WordPress oc adm policy add-SCS-to-group anyuid system ：
  serviceaccounts ： WordPress oc adm policy add-SCS-to-user privileged -z
  default -n WordPress
  ```

- 支持的存储类驱动程序：Astra Control支持使用以下驱动程序支持的存储类还原备份：
  - `ontap-nas`
  - `ontap-nas-economy`
  - `ontap-san`
  - `ontap-san-economy`
- **(**仅限**ONTA-NAS**经济型驱动程序**)**备份和还原：备份或还原使用由备份的存储类的应用程序之前 `ontap-nas-economy` 驱动程序、请验证 "ONTAP存储后端上的Snapshot目录处于隐藏状态"。如果未能隐藏此目录、可能会导致无法访问您的应用程序、尤其是在使用NFSv3的情况下。
- * Helm部署的应用程序*：完全支持使用Helm 3部署的应用程序(或从Helm 2升级到Helm 3)。不支持使用Helm 2 部署的应用程序。

> 在与其他应用程序共享资源的应用程序上执行原位还原操作可能会产生意外结果。对其中一个应用程序执行原位还原时、这些应用程序之间共享的任何资源都会被替换。有关详细信息，请参见此示例。

根据要还原的归档类型、执行以下步骤：

使用**Web UI**从备份或快照还原数据

您可以使用Asta Control Web UI还原数据。

步骤

1. 选择 * 应用程序 * ，然后选择应用程序的名称。
2. 从“操作”列的“选项”菜单中，选择*Restore*。
3. 选择还原类型：
   - 还原到原始命名空间：使用此操作步骤 将应用程序原位还原到原始集群。

> 如果您的应用使用由支持的存储类 `ontap-nas-economy` 驱动程序、则必须使用原始存储类还原应用程序。如果要将应用程序还原到同一命名空间、则不能指定其他存储类。

i. 选择要用于原位还原应用程序的快照或备份、这会将应用程序还原到其自身的早期版本。

ii. 选择 * 下一步 * 。

> 如果还原到先前已删除的命名空间、则在还原过程中会创建一个同名的新命名空间。任何有权管理先前删除的命名空间中的应用程序的用户都需要手动还原对新重新创建的命名空间的权限。

- 还原到新命名空间：使用此操作步骤 将应用程序还原到另一个集群或使用与源不同的命名空间。

  i. 指定已还原应用程序的名称。

  ii. 为要还原的应用程序选择目标集群。

  iii. 为与应用程序关联的每个源命名空间输入目标命名空间。

  > 作为此还原选项的一部分、Astra Control会创建新的目标命名空间。指定的目标命名空间不能已存在于目标集群上。

  iv. 选择 * 下一步 * 。

  v. 选择用于还原应用程序的快照或备份。

  vi. 选择 * 下一步 * 。

  vii. 选择以下选项之一：

    - 使用原始存储类还原：除非目标集群上不存在、否则应用程序将使用最初关联的存储类。在这种情况下、将使用集群的默认存储类。
    - 使用其他存储类还原：选择目标集群上的存储类。在还原过程中、所有应用程序卷(无论其最初关联的存储类是什么)都将迁移到此不同的存储类。

  viii. 选择 * 下一步 * 。

4. 选择要筛选的任何资源：

   - 恢复所有资源：恢复与原始应用程序关联的所有资源。
   - 过滤资源：指定规则以还原原始应用程序资源的子集：

     i. 选择在已还原的应用程序中包括或排除资源。

     ii. 选择*添加包含规则*或*添加排除规则*，并配置规则以在应用程序恢复期间过滤正确的资源。您可以编辑或删除规则、然后重新创建规则、直到配置正确为止。

     > 要了解有关配置包含和排除规则的信息、请参见 [在应用程序还原期间筛选资源]。

5. 选择 * 下一步 * 。

6. 请仔细查看有关还原操作的详细信息，键入“restore”(如果出现提示)，然后选择*Restore*。

[技术预览]使用自定义资源从备份中恢复(CR)

您可以使用自定义资源(CR)文件将备份中的数据还原到其他命名空间或原始源命名空间。

使用**CR**从备份还原

步骤

1. 创建自定义资源(CR)文件并将其命名为 `astra-control-backup-restore-cr.yaml`。更新方括号<>中的值以匹配您的Astra Control环境和集群配置：
   - <CR_NAME>：此CR操作的名称；为您的环境选择一个合理的名称。
   - <APPVAULT_NAME>：存储备份内容的AppVault的名称。
   - <BACKUP_PATH>：AppVault中存储备份内容的路径。例如：

     ```
     ONTAP-S3_1343ff5e-4c41-46b5-af00/backups/schedule-
     20231213023800_94347756-9d9b-401d-a0c3
     ```

   - <SOURCE_NAMESPACE>：还原操作的源命名空间。
   - <DESTINATION_NAMESPACE>：还原操作的目标命名空间。

     ```
     apiVersion: astra.netapp.io/v1
     kind: BackupRestore
     metadata:
       name: <CR_NAME>
       namespace: astra-connector
     spec:
       appVaultRef: <APPVAULT_NAME>
       appArchivePath: <BACKUP_PATH>
       namespaceMapping: [{"source": "<SOURCE_NAMESPACE>",
     "destination": "<DESTINATION_NAMESPACE>"}]
     ```

2. (可选)如果只需要选择要还原的应用程序的某些资源、请添加筛选功能、其中包括或排除标记有特定标签的资源：
   - <INCLUDE-EXCLUDE>：_(筛选所需)_使用 `include` 或 `exclude` 包括或排除资源匹配备程序中定义的资源。添加以下resourceMatchers参数以定义要包括或排除的资源：
     - <GROUP>：_(可选)_要筛选的资源组。
     - <KIND>：_(可选)_要筛选的资源种类。
     - <VERSION>：要筛选的资源的_(可选)_版本。
     - <NAMES>：要筛选的资源的Kubenetes metadata.name字段中的_(可选)_个名称。
     - <NAMESPACES>：_(可选)_要筛选的资源的Kubenetes metadata.name字段中的命名区。
     - <SELECTORS>：中定义的资源的Kubelnetes metadata.name字段中的_(可选)_标签选择器字符串 "Kubernetes 文档"。示例 `"trident.netapp.io/os=linux"`。

       示例

```
spec:
    resourceFilter:
        resourceSelectionCriteria: "<INCLUDE-EXCLUDE>"
        resourceMatchers:
           group: <GROUP>
           kind: <KIND>
           version: <VERSION>
           names: <NAMES>
           namespaces: <NAMESPACES>
           labelSelectors: <SELECTORS>
```

3. 在您填充之后 `astra-control-backup-restore-cr.yaml` 使用正确值的文件、应用CR：

```
kubectl apply -f astra-control-backup-restore-cr.yaml
```

使用**CR**从备份还原到原始命名空间

步骤

1. 创建自定义资源(CR)文件并将其命名为 `astra-control-backup-ipr-cr.yaml`。更新方括号<>中的值以匹配您的Astra Control环境和集群配置：
   - <CR_NAME>：此CR操作的名称；为您的环境选择一个合理的名称。
   - <APPVAULT_NAME>：存储备份内容的AppVault的名称。
   - <BACKUP_PATH>：AppVault中存储备份内容的路径。例如：

     ```
     ONTAP-S3_1343ff5e-4c41-46b5-af00/backups/schedule-
     20231213023800_94347756-9d9b-401d-a0c3
     ```

     ```
     apiVersion: astra.netapp.io/v1
     kind: BackupInplaceRestore
     metadata:
       name: <CR_NAME>
       namespace: astra-connector
     spec:
       appVaultRef: <APPVAULT_NAME>
       appArchivePath: <BACKUP_PATH>
     ```

2. (可选)如果只需要选择要还原的应用程序的某些资源、请添加筛选功能、其中包括或排除标记有特定标签的资源：
   - <INCLUDE-EXCLUDE>：_(筛选所需)_使用 `include` 或 `exclude` 包括或排除资源匹配备程序中定义的资源。添加以下resourceMatchers参数以定义要包括或排除的资源：

- <GROUP>：_(可选)_要筛选的资源组。
- <KIND>：_(可选)_要筛选的资源种类。
- <VERSION>：要筛选的资源的_(可选)_版本。
- <NAMES>：要筛选的资源的Kubenetes metadata.name字段中的_(可选)_个名称。
- <NAMESPACES>：_(可选)_要筛选的资源的Kubenetes metadata.name字段中的命名区。
- <SELECTORS>：中定义的资源的Kubelnetes metadata.name字段中的_(可选)_标签选择器字符串 "Kubernetes 文档"。示例 `"trident.netapp.io/os=linux"`。

  示例

```
spec:
    resourceFilter:
        resourceSelectionCriteria: "<INCLUDE-EXCLUDE>"
        resourceMatchers:
           group: <GROUP>
           kind: <KIND>
           version: <VERSION>
           names: <NAMES>
           namespaces: <NAMESPACES>
           labelSelectors: <SELECTORS>
```

3. 在您填充之后 `astra-control-backup-ipr-cr.yaml` 使用正确值的文件、应用CR：

```
kubectl apply -f astra-control-backup-ipr-cr.yaml
```

[技术预览]使用自定义资源从快照恢复(CR)

您可以使用自定义资源(CR)文件从快照将数据还原到其他命名空间或原始源命名空间。

使用**CR**从快照还原

步骤

1. 创建自定义资源(CR)文件并将其命名为 `astra-control-snapshot-restore-cr.yaml`。更新方括号<>中的值以匹配您的Astra Control环境和集群配置：
   - <CR_NAME>：此CR操作的名称；为您的环境选择一个合理的名称。
   - <APPVAULT_NAME>：存储备份内容的AppVault的名称。
   - <BACKUP_PATH>：AppVault中存储备份内容的路径。例如：

     ```
     ONTAP-S3_1343ff5e-4c41-46b5-af00/backups/schedule-
     20231213023800_94347756-9d9b-401d-a0c3
     ```

   - <SOURCE_NAMESPACE>：还原操作的源命名空间。
   - <DESTINATION_NAMESPACE>：还原操作的目标命名空间。

     ```
     apiVersion: astra.netapp.io/v1
     kind: SnapshotRestore
     metadata:
       name: <CR_NAME>
       namespace: astra-connector
     spec:
       appArchivePath: <BACKUP_PATH>
       appVaultRef: <APPVAULT_NAME>
       namespaceMapping: [{"source": "<SOURCE_NAMESPACE>",
     "destination": "<DESTINATION_NAMESPACE>"}]
     ```

2. (可选)如果只需要选择要还原的应用程序的某些资源、请添加筛选功能、其中包括或排除标记有特定标签的资源：
   - <INCLUDE-EXCLUDE>：_(筛选所需)_使用 `include` 或 `exclude` 包括或排除资源匹配备程序中定义的资源。添加以下resourceMatchers参数以定义要包括或排除的资源：
     - <GROUP>：_(可选)_要筛选的资源组。
     - <KIND>：_(可选)_要筛选的资源种类。
     - <VERSION>：要筛选的资源的_(可选)_版本。
     - <NAMES>：要筛选的资源的Kubenetes metadata.name字段中的_(可选)_个名称。
     - <NAMESPACES>：_(可选)_要筛选的资源的Kubenetes metadata.name字段中的命名区。
     - <SELECTORS>：中定义的资源的Kubelnetes metadata.name字段中的_(可选)_标签选择器字符串 "Kubernetes 文档"。示例 `"trident.netapp.io/os=linux"`。

       示例

```
spec:
    resourceFilter:
        resourceSelectionCriteria: "<INCLUDE-EXCLUDE>"
        resourceMatchers:
           group: <GROUP>
           kind: <KIND>
           version: <VERSION>
           names: <NAMES>
           namespaces: <NAMESPACES>
           labelSelectors: <SELECTORS>
```

3. 在您填充之后 `astra-control-snapshot-restore-cr.yaml` 使用正确值的文件、应用CR：

```
kubectl apply -f astra-control-snapshot-restore-cr.yaml
```

使用**CR**从快照还原到原始命名空间

步骤

1. 创建自定义资源(CR)文件并将其命名为 `astra-control-snapshot-ipr-cr.yaml`。更新方括号<>中的值以匹配您的Astra Control环境和集群配置：
   - <CR_NAME>：此CR操作的名称；为您的环境选择一个合理的名称。
   - <APPVAULT_NAME>：存储备份内容的AppVault的名称。
   - <BACKUP_PATH>：AppVault中存储备份内容的路径。例如：

     ```
     ONTAP-S3_1343ff5e-4c41-46b5-af00/backups/schedule-
     20231213023800_94347756-9d9b-401d-a0c3
     ```

     ```
     apiVersion: astra.netapp.io/v1
     kind: SnapshotInplaceRestore
     metadata:
       name: <CR_NAME>
       namespace: astra-connector
     spec:
       appArchivePath: <BACKUP_PATH>
       appVaultRef: <APPVAULT_NAME>
     ```

2. (可选)如果只需要选择要还原的应用程序的某些资源、请添加筛选功能、其中包括或排除标记有特定标签的资源：
   - <INCLUDE-EXCLUDE>：_(筛选所需)_使用 `include` 或 `exclude` 包括或排除资源匹配备程序中定义的资源。添加以下resourceMatchers参数以定义要包括或排除的资源：

- <GROUP>：_(可选)_要筛选的资源组。
- <KIND>：_(可选)_要筛选的资源种类。
- <VERSION>：要筛选的资源的_(可选)_版本。
- <NAMES>：要筛选的资源的Kubenetes metadata.name字段中的_(可选)_个名称。
- <NAMESPACES>：_(可选)_要筛选的资源的Kubenetes metadata.name字段中的命名区。
- <SELECTORS>：中定义的资源的Kubelnetes metadata.name字段中的_(可选)_标签选择器字符串 "Kubernetes 文档"。示例 `"trident.netapp.io/os=linux"`。

  示例

```
spec:
    resourceFilter:
        resourceSelectionCriteria: "<INCLUDE-EXCLUDE>"
        resourceMatchers:
           group: <GROUP>
           kind: <KIND>
           version: <VERSION>
           names: <NAMES>
           namespaces: <NAMESPACES>
           labelSelectors: <SELECTORS>
```

3. 在您填充之后 `astra-control-snapshot-ipr-cr.yaml` 使用正确值的文件、应用CR：

```
kubectl apply -f astra-control-snapshot-ipr-cr.yaml
```

结果

Astra Control 会根据您提供的信息还原应用程序。如果您已原位还原应用程序、则现有永久性卷的内容将替换为已还原应用程序中的永久性卷的内容。

> 在执行数据保护操作(克隆、备份或还原)并随后调整永久性卷大小后、在Web UI中显示新卷大小之前、最多会有20分钟的延迟。数据保护操作将在几分钟内成功完成，您可以使用存储后端的管理软件确认卷大小的更改。

> 任何按命名空间名称/ID或命名空间标签限制命名空间的成员用户都可以将应用程序克隆或还原到同一集群上的新命名空间或其组织帐户中的任何其他集群。但是，同一用户无法访问新命名空间中的克隆或还原应用程序。克隆或还原操作创建新命名空间后、帐户管理员/所有者可以编辑成员用户帐户并更新受影响用户的角色约束、以授予对新命名空间的访问权限。

在应用程序还原期间筛选资源

您可以向添加筛选器规则 "还原" 此操作将指定要从还原的应用程序中包括或排除的现有应用程序资源。您可以根据指定的命名空间、标签或GVK (GroupVersion Kind)包括或排除资源。

展开以了解有关包括和排除方案的更多信息

- 选择包含原始命名空间的规则(就地还原)：您在规则中定义的现有应用程序资源将被删除，并替换为用于还原的选定快照或备份中的应用程序资源。未在包含规则中指定的任何资源将保持不变。
- 选择包含新名称空间的规则：使用此规则在还原的应用程序中选择所需的特定资源。未在包含规则中指定的任何资源将不会包含在已还原的应用程序中。
- 选择包含原始名称空间的排除规则(就地恢复)：您指定要排除的资源将不会还原、并且保持不变。未指定排除的资源将从快照或备份中还原。如果筛选的资源中包含相应的状态集、则永久性卷上的所有数据都将被删除并重新创建。
- 选择包含新名称空间的排除规则：使用此规则可选择要从还原的应用程序中删除的特定资源。未指定排除的资源将从快照或备份中还原。

规则可以是包含类型、也可以是排除类型。不提供组合使用资源包含和排除的规则。

步骤

1. 选择筛选资源并在恢复应用程序向导中选择包含或排除选项后，选择*添加包含规则*或*添加排除规则*。

   您不能排除Asta Control自动包含的任何集群范围的资源。

2. 配置筛选器规则：

   必须至少指定一个命名空间、标签或GVK。确保在应用筛选器规则后保留的任何资源足以使已还原的应用程序保持运行状况良好。

   a. 为规则选择特定命名空间。如果不进行选择、则会在筛选器中使用所有名称空间。

      如果您的应用程序最初包含多个名称空间、而您将其还原到新的名称空间、则会创建所有名称空间、即使它们不包含资源也是如此。

   b. (可选)输入资源名称。

   c. (可选)标签选择器：包括A "标签选择器" 以添加到规则中。标签选择器用于仅筛选与选定标签匹配的资源。

   d. (可选)选择*使用GVK (GroupVersion Kind)设置来筛选资源*以获取其他筛选选项。

      如果使用GVK筛选器、则必须指定版本和种类。

      i. (可选)组：从下拉列表中选择Kubernetes API组。

      ii. **KND**：从下拉列表中选择要在筛选器中使用的Kubernetes资源类型的对象模式。

      iii. 版本：选择Kubernetes API版本。

3. 查看根据条目创建的规则。

4. 选择 * 添加 * 。

   您可以根据需要创建任意数量的资源包含和排除规则。这些规则将显示在启动操作之前的还原应用程序摘要中。

如果某个应用程序与其他应用程序共享资源、则就地恢复会变得非常复杂

您可以对与其他应用共享资源并产生意外结果的应用程序执行原位还原操作。对其中一个应用程序执行原位还原时、这些应用程序之间共享的任何资源都会被替换。

以下示例情形会在使用NetApp SnapMirror复制进行还原时产生不希望出现的情况：

1. 您可以定义应用程序 `app1` 使用命名空间 `ns1`。
2. 您可以为配置复制关系 `app1`。
3. 您可以定义应用程序 `app2` (在同一集群上)使用命名空间 `ns1` 和 `ns2`。
4. 您可以为配置复制关系 `app2`。
5. 反向复制 `app2`。这将导致 `app1` 要停用的源集群上的应用程序。

使用**SnapMirror**技术在存储后端之间复制应用程序

使用Astra Control、您可以使用NetApp SnapMirror技术的异步复制功能、以低RPO (恢复点目标)和低RTO (恢复时间目标)为应用程序构建业务连续性。配置后、应用程序便可将数据和应用程序更改从一个存储后端复制到另一个存储后端、复制到同一集群上或复制到不同集群之间。

有关备份/还原与复制之间的比较、请参见 "数据保护概念"。

您可以在不同情形下复制应用程序、例如以下仅限内部部署、混合和多云情形：

- 内部站点A到内部站点A
- 内部站点A到内部站点B
- 借助Cloud Volumes ONTAP从内部环境迁移到云
- 通过Cloud Volumes ONTAP迁移到内部环境
- 采用Cloud Volumes ONTAP 的云到云(在同一云提供商的不同区域之间或不同云提供商之间)

Astra Control可以跨内部集群、内部到云(使用Cloud Volumes ONTAP)或云之间(Cloud Volumes ONTAP到Cloud Volumes ONTAP)复制应用程序。

> 您可以同时按相反方向复制不同的应用程序。例如、应用程序A、B、C可以从数据中心1复制到数据中心2；应用程序X、Y、Z可以从数据中心2复制到数据中心1。

使用Astra Control、您可以执行以下与复制应用程序相关的任务：

- [设置复制关系]
- [在目标集群上使复制的应用程序联机(故障转移)]
- [重新同步故障转移复制]
- [反向复制应用程序]
- [将应用程序故障恢复到原始源集群]
- [删除应用程序复制关系]

复制前提条件

Astra Control应用程序复制要求在开始之前满足以下前提条件：

ONTAP 集群

- **Astra**控件配置程序或**Astra**三端：使用ONTAP作为后端的源和目标Kubernetes集群上必须同时存在Astra控件配置程序或Astra三端。Astra Control支持使用以下驱动程序支持的存储类通过NetApp SnapMirror技术进行复制：
  - `ontap-nas`
  - `ontap-san`
- 许可证：必须在源和目标ONTAP集群上启用使用数据保护包的ONTAP SnapMirror异步许可证。请参见 "ONTAP 中的SnapMirror许可概述" 有关详细信息 …

对等

- 集群和**SVM**：ONTAP存储后端必须建立对等状态。请参见 "集群和 SVM 对等概述" 有关详细信息 …

  确保两个ONTAP集群之间的复制关系中使用的SVM名称是唯一的。

- **Astra Control**置备程序或**Astra**三端和**SVM**：对等远程SVM必须可供目标集群上的Astra Control置备程序或Astra三端。

  Astra 控制中心

  "部署Asta Control Center" 在第三个故障域或二级站点中、以实现无缝灾难恢复。

- 托管后端：您需要在Astra控制中心中添加和管理ONTAP存储后端、才能创建复制关系。

  如果启用了Astra Control配置程序、则可以选择在Astra控制中心中添加和管理ONTAP存储后端。

- 受管集群：使用Asta Control添加和管理以下集群、最好是在不同的故障域或站点上：
  - 源Kubbernetes集群
  - 目标Kubbernetes集群
  - 关联的ONTAP集群
- 用户帐户：将ONTAP存储后端添加到Astra控制中心时、请应用具有"admin"角色的用户凭据。此角色具有访问方法 `http` 和 `ontapi` 已在ONTAP 源集群和目标集群上启用。请参见 "管理ONTAP 文档中的用户帐户" 有关详细信息 …

  使用Astra Control配置程序功能、您无需专门定义"管理员"角色即可在Astra Control Center中管理集群、因为Astra Control Center不需要这些凭据。

对于使用基于TCP协议的NVMe的存储后端、Astra控制中心不支持NetApp SnapMirror复制。

Astra Trident / ONTAP 配置

Astra Control Center要求您至少配置一个存储后端、以便为源集群和目标集群同时支持复制。如果源集群和目标集群相同、则目标应用程序应使用与源应用程序不同的存储后端、以获得最佳故障恢复能力。

> Astra Control复制支持使用单个存储类的应用程序。将应用程序添加到命名空间时、请确保该应用程序与命名空间中的其他应用程序具有相同的存储类。向复制的应用程序添加PVC时、请确保新PVC与命名空间中的其他PVC具有相同的存储类。

设置复制关系

设置复制关系涉及以下方面：

- 选择您希望Astra Control创建应用程序快照的频率(包括应用程序的Kubbernetes资源以及应用程序每个卷的卷快照)
- 选择复制计划(包括Kubernetes资源以及永久性卷数据)
- 设置创建快照的时间

步骤

1. 从Astra Control左侧导航栏中、选择*应用程序*。
2. 选择*数据保护*>*复制*选项卡。
3. 选择*配置复制策略*。或者、从应用程序保护框中、选择操作选项并选择*配置复制策略*。
4. 输入或选择以下信息：
   - 目标集群：输入目标集群(可以与源集群相同)。
   - 目标存储类：选择或输入在目标ONTAP集群上使用对等SVM的存储类。作为最佳实践、目标存储类应指向与源存储类不同的存储后端。
   - 复制类型： `Asynchronous` 是当前唯一可用的复制类型。
   - 目标命名空间：为目标集群输入新的或现有的目标命名空间。
   - (可选)通过选择*添加命名空间*并从下拉列表中选择命名空间来添加其他命名空间。
   - 复制频率：设置您希望Astra Control创建快照并将其复制到目标的频率。
   - **Offset**：设置从Astra Control创建快照的小时数开始的分钟数。您可能希望使用偏移量、以便它不会与其他计划的操作保持一致。

     > 偏移备份和复制计划以避免计划重叠。例如、在每小时的前几个小时执行备份、并计划复制、以5分钟的偏移和10分钟的间隔开始。

5. 选择*下一步*、查看摘要、然后选择*保存*。

   > 首先、在执行第一个计划之前、状态将显示"app-mirror"。

   Asta Control创建用于复制的应用程序快照。

6. 要查看应用程序快照状态，请选择*Applications*>*Snapshot选项卡。

   快照名称使用的格式 `replication-schedule-<string>`。Asta Control会保留用于复制的最后一个快照。成功完成复制后、所有较早的复制快照都会被删除。

结果

这将创建复制关系。

建立关系后、Astra Control将完成以下操作：

- 在目标上创建命名空间(如果不存在)
- 在目标命名空间上创建与源应用程序的PVC对应的PVC。
- 创建应用程序一致的初始快照。
- 使用初始快照为永久性卷建立SnapMirror关系。

"数据保护"页面显示复制关系的状态：
<Health status>|<Relationship life cycle state>

例如：normal | established.

在本主题末尾了解有关复制状态和状态的更多信息。

在目标集群上使复制的应用程序联机(故障转移)

使用Astra Control、您可以将复制的应用程序故障转移到目标集群。此操作步骤 将停止复制关系并使应用程序在目标集群上联机。如果应用程序正常运行、则此操作步骤 不会停止源集群上的应用程序。

步骤

1. 从Astra Control左侧导航栏中、选择*应用程序*。
2. 选择*数据保护*>*复制*选项卡。
3. 从操作菜单中，选择*故障转移*。
4. 在故障转移页面中、查看相关信息并选择*故障转移*。

结果

故障转移操作步骤后会执行以下操作：

- 此时将根据最新复制的快照启动目标应用程序。
- 源集群和应用程序(如果运行正常)不会停止、并且将继续运行。
- 复制状态将更改为"故障转移"、然后在完成后更改为"故障转移"。
- 根据故障转移时源应用程序上的计划、源应用程序的保护策略将复制到目标应用程序。
- 如果源应用程序启用了一个或多个还原后执行挂钩、则会为目标应用程序运行这些执行挂钩。
- Astra Control会在源集群和目标集群上显示应用程序及其各自的运行状况。

重新同步故障转移复制

重新同步操作将重新建立复制关系。您可以选择关系的源、以便在源或目标集群上保留数据。此操作将重新建立SnapMirror关系、以便按所选方向启动卷复制。

此过程会在重新建立复制之前停止新目标集群上的应用程序。

> 在重新同步过程中、生命周期状态将显示为"正在建立"。

步骤

1. 从Astra Control左侧导航栏中、选择*应用程序*。
2. 选择*数据保护*>*复制*选项卡。
3. 从操作菜单中，选择*Resync*。
4. 在重新同步页面中、选择包含要保留的数据的源或目标应用程序实例。

> 请仔细选择重新同步源、因为目标上的数据将被覆盖。

5. 选择*重新同步*以继续。
6. 键入"resync-"进行确认。
7. 选择*是、重新同步*以完成。

结果

- 复制页面将显示"正在建立"作为复制状态。
- Astra Control将停止新目标集群上的应用程序。
- Astra Control使用SnapMirror重新同步功能按选定方向重新建立永久性卷复制。
- 复制页面将显示已更新的关系。

反向复制应用程序

这是一项计划内操作、可将应用程序移至目标存储后端、同时继续复制回原始源存储后端。Asta Control会停止源应用程序并将数据复制到目标、然后再故障转移到目标应用程序。

在这种情况下、您将交换源和目标。

步骤

1. 从Astra Control左侧导航栏中、选择*应用程序*。
2. 选择*数据保护*>*复制*选项卡。
3. 从操作菜单中，选择*反向复制*。
4. 在反向复制页面中、查看相关信息并选择*反向复制*以继续。

结果

反向复制会执行以下操作：

- 系统会为原始源应用程序的Kubbernetes资源创建一个快照。
- 通过删除原始源应用程序的Kubernetes资源(保留PVC和PV)、可以正常停止原始源应用程序的Pod。
- 关闭Pod后、将为应用程序的卷创建快照并进行复制。
- SnapMirror关系将中断、从而使目标卷做好读/写准备。
- 此应用程序的Kubornetes资源将使用在初始源应用程序关闭后复制的卷数据从关闭前的快照中还原。
- 反向重新建立复制。

将应用程序故障恢复到原始源集群

使用Astra Control、您可以通过以下操作序列在故障转移操作后实现"故障恢复"。在此工作流中、Astra Control会在反转复制方向之前、将所有应用程序更改复制(重新同步)回原始源应用程序。

此过程从已完成故障转移到目标的关系开始、涉及以下步骤：

- 从故障转移状态开始。
- 重新同步此关系。
- 反转复制。

步骤

1. 从Astra Control左侧导航栏中、选择*应用程序*。
2. 选择*数据保护*>*复制*选项卡。
3. 从操作菜单中，选择*Resync*。
4. 对于故障恢复操作、请选择故障转移应用程序作为重新同步操作的源(在故障转移后保留写入的任何数据)。
5. 键入"resync-"进行确认。
6. 选择*是、重新同步*以完成。
7. 重新同步完成后、在"Data Protection">"Replication"选项卡中、从"Actions"菜单中选择*反向复制*。
8. 在反向复制页面中、查看相关信息并选择*反向复制*。

结果

这将合并"重新同步"和"反向关系"操作的结果、以便在复制恢复到原始目标集群的情况下使应用程序在原始源集群上联机。

删除应用程序复制关系

删除此关系会导致出现两个独立的应用程序、它们之间没有任何关系。

步骤

1. 从Astra Control左侧导航栏中、选择*应用程序*。
2. 选择*数据保护*>*复制*选项卡。
3. 从应用程序保护框或关系图中，选择*删除复制关系*。

结果

删除复制关系后会执行以下操作：

- 如果已建立此关系、但此应用程序尚未在目标集群上联机(故障转移)、则Astra Control将保留初始化期间创建的PVC、在目标集群上保留一个"空"受管应用程序、并保留目标应用程序以保留可能已创建的任何备份。
- 如果应用程序已在目标集群上联机(故障转移)、则Astra Control会保留PVC和目标应用程序。源应用程序和目标应用程序现在被视为独立的应用程序。备份计划会同时保留在两个应用程序上、但不会彼此关联。

复制关系运行状况和关系生命周期状态

Astra Control显示关系的运行状况以及复制关系的生命周期状态。

复制关系运行状况

以下状态指示复制关系的运行状况：

- 正常：关系正在建立或已建立、并且最近的快照已成功传输。
- 警告：此关系正在进行故障转移或已进行故障转移(因此不再保护源应用程序)。
- * 严重 *
  - 此关系正在建立或故障转移、上次协调尝试失败。
  - 已建立此关系、上次尝试协调添加新PVC失败。
  - 已建立此关系(因此已成功复制快照、并且可以进行故障转移)、但最近的快照失败或无法复制。

复制生命周期状态

以下状态反映了复制生命周期的不同阶段：

- 正在建立：正在创建新的复制关系。Astra Control会根据需要创建命名空间、在目标集群上的新卷上创建永久性卷声明(PVC)、并创建SnapMirror关系。此状态还可以指示复制正在重新同步或反转复制。
- 已建立：存在复制关系。Astra Control会定期检查PVC是否可用、检查复制关系、定期创建应用程序快照并确定应用程序中的任何新源PVC。如果是、则Astra Control会创建资源以将其包括在复制中。
- 故障转移：Asta Control会中断SnapMirror关系、并从上次成功复制的应用程序快照还原应用程序的Kubernetes资源。
- 故障转移：Asta Control停止从源集群复制、使用目标上最新(成功)复制的应用程序快照、并还原Kubernetes资源。
- 正在重新同步：Astra Control使用SnapMirror重新同步将重新同步源上的新数据重新同步到重新同步目标。此操作可能会根据同步方向覆盖目标上的某些数据。Astra Control会停止在目标命名空间上运行的应用程序、并删除Kubernetes应用程序。在重新同步过程中、状态将显示为正在建立。
- 正在反转：是指在继续复制回原始源集群的同时将应用程序移动到目标集群的计划操作。Astra Control会停止源集群上的应用程序、将数据复制到目标、然后将应用程序故障转移到目标集群。在反向复制期间、状态显示为"正在 建立"。
- 正在删除：
  - 如果已建立复制关系、但尚未进行故障转移、则Astra Control会删除复制期间创建的PVC、并删除目标受管应用程序。
  - 如果复制已失败、则Astra Control会保留PVC和目标应用程序。

克隆和迁移应用程序

您可以克隆现有应用程序、以便在同一个Kubernetes集群或另一个集群上创建重复的应用程序。当 Astra Control 克隆应用程序时，它会为您的应用程序配置和永久性存储创建一个克隆。

如果您需要将应用程序和存储从一个 Kubernetes 集群移动到另一个集群，则克隆可以助您一臂之力。例如，您可能希望通过 CI/CD 管道以及在 Kubernetes 命名空间之间移动工作负载。您可以使用Astra控制中心UI或 "Astra Control API" 克隆和迁移应用程序。

开始之前

- 检查目标卷：如果克隆到其他存储类、请确保该存储类使用相同的永久性卷访问模式(例如ReadWriteMany)。如果目标永久性卷访问模式不同、则克隆操作将失败。例如、如果源永久性卷使用rwx访问模式、请选择无法提供rwx的目标存储类、例如Azure托管磁盘、AWS EBS、Google持久磁盘或 `ontap-san`，发生原因将使克隆操作失败。有关永久性卷访问模式的详细信息、请参阅 "Kubernetes" 文档。
- 要将应用程序克隆到其他集群、您需要确保包含源集群和目标集群(如果不相同)的云实例具有默认分段。您需要为每个云实例分配一个默认分段。
- 在克隆操作期间、需要IngressClass资源或webhooks才能正常运行的应用程序不能在目标集群上定义这些资源。

> 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`

克隆限制

- 显式存储类：如果部署的应用程序已明确设置存储类、并且需要克隆此应用程序、则目标集群必须具有最初指定的存储类。将具有显式设置的存储类的应用程序克隆到没有相同存储类的集群将失败。
- **UNONTAP NAS**经济型应用程序：如果应用程序的存储类由提供支持、则无法使用克隆操作 `ontap-nas-economy` 驱动程序。但是、您可以 "为ONTAP NAS经济型操作启用备份和还原"。
- 克隆和用户约束：任何按命名空间名称/ID或命名空间标签限制命名空间的成员用户都可以将应用程序克隆或还原到同一集群上的新命名空间或其组织帐户中的任何其他集群。但是，同一用户无法访问新命名空间中的克隆或还原应用程序。克隆或还原操作创建新命名空间后、帐户管理员/所有者可以编辑成员用户帐户并更新受影响用户的角色约束、以授予对新命名空间的访问权限。
- 克隆使用默认分段：在应用程序备份或应用程序还原期间、您可以选择指定分段ID。但是，应用程序克隆操作始终使用已定义的默认分段。没有选项可用于更改克隆的分段。如果要控制使用哪个存储分段，您可以选择 "更改存储分段默认值" 或者执行 "backup" 后跟 A "还原" 请单独使用。
- 使用**Jenkins CI**：如果克隆操作员部署的Jenkins CI实例、则需要手动还原持久数据。这是应用程序部署模式的一个限制。
- 对于**S3**存储分段：Astra控制中心中的S3存储分段不报告可用容量。在备份或克隆由 Astra 控制中心管理的应用程序之前，请检查 ONTAP 或 StorageGRID 管理系统中的存储分段信息。
- 使用特定版本的**PostgreSQL**时：同一集群中的应用程序克隆始终失败，并显示BitNami PostgreSQL 11.5.0图表。要成功克隆，请使用图表的早期或更高版本。

OpenShift 注意事项

- 集群和**OpenShift**版本：如果在集群之间克隆应用程序、则源集群和目标集群必须是OpenShift的相同分发版本。例如，如果从 OpenShift 4.7 集群克隆应用程序，请使用同时也是 OpenShift 4.7 的目标集群。
- 项目和**UID**：在OpenShift集群上创建用于托管应用程序的项目时、系统会为该项目(或Kubernetes命名空间)分配一个SecurityContext UID。要使 Astra 控制中心能够保护您的应用程序并将应用程序移动到 OpenShift 中的其他集群或项目，您需要添加策略，使应用程序能够作为任何 UID 运行。例如，以下 OpenShift 命令行界面命令会为 WordPress 应用程序授予相应的策略。

```
oc new-project WordPress oc adm policy add-SCS-to-group anyuid system ：
```

```
serviceaccounts ： WordPress oc adm policy add-SCS-to-user privileged -z
default -n WordPress
```

步骤

1. 选择 * 应用程序 * 。
2. 执行以下操作之一：
   - 在 * 操作 * 列中选择所需应用程序的选项菜单。
   - 选择所需应用程序的名称，然后选择页面右上角的状态下拉列表。
3. 选择 * 克隆 * 。
4. 指定克隆的详细信息：
   - 输入名称。
   - 选择克隆的目标集群。
   - 输入克隆的目标命名空间。与应用程序关联的每个源命名空间都会映射到您定义的目标命名空间。

     在克隆操作中、Astra Control会创建新的目标命名空间。指定的目标命名空间不能已存在于目标集群上。

   - 选择 * 下一步 * 。
   - 选择将原始存储类与应用程序保持关联、或者选择其他存储类。

     您可以将应用程序的存储类迁移到本机云提供商存储类或其他受支持的存储类、也可以将应用程序从支持的存储类迁移 `ontap-nas-economy` 存储类 `ontap-nas` 在同一集群上、或者将应用程序复制到存储类由支持的另一集群 `ontap-nas-economy` 驱动程序。

   如果您选择了其他存储类、但在还原时此存储类不存在、则会返回错误。

5. 选择 * 下一步 * 。
6. 查看有关克隆的信息、然后选择*克隆*。

结果

Astra Control会根据您提供的信息克隆应用程序。当新应用程序克隆处于中时、克隆操作成功 `Healthy` 状态。

克隆或还原操作创建新命名空间后、帐户管理员/所有者可以编辑成员用户帐户并更新受影响用户的角色约束、以授予对新命名空间的访问权限。

在执行数据保护操作(克隆、备份或还原)并随后调整永久性卷大小后、在UI中显示新卷大小之前、最多会有20分钟的延迟。数据保护操作将在几分钟内成功完成，您可以使用存储后端的管理软件确认卷大小的更改。

管理应用程序执行挂钩

执行挂钩是一种自定义操作、您可以将其配置为与受管应用程序的数据保护操作结合运行。例如、如果您有一个数据库应用程序、则可以使用执行挂钩在快照之前暂停所有数据

库事务、并在快照完成后恢复事务。这样可以确保应用程序一致的快照。

执行挂钩的类型

Astra Control Center支持以下类型的执行挂钩、具体取决于它们可以运行的时间：

- 预快照
- 快照后
- 预备份
- 备份后
- 还原后
- 故障转移后

执行钩筛选器

向应用程序添加或编辑执行挂钩时，可以向执行挂钩添加过滤器，以管理挂钩将匹配的容器。对于在所有容器上使用相同容器映像的应用程序、筛选器非常有用、但可能会将每个映像用于不同的用途(例如Elasticsearch)。通过筛选器、您可以创建执行挂钩在某些容器上运行的方案、但不一定是所有相同的容器上运行的方案。如果为单个执行钩创建多个筛选器、则这些筛选器将与逻辑运算符和运算符结合使用。每个执行连接最多可以有10个活动筛选器。

添加到执行挂钩中的每个筛选器都会使用一个正则表达式来匹配集群中的容器。当某个挂钩与某个容器匹配时、该挂钩将在该容器上运行其关联脚本。筛选器的正则表达式使用正则表达式2 (RE2)语法、不支持创建从匹配列表中排除容器的筛选器。有关Astra Control在执行挂钩筛选器中支持正则表达式语法的信息、请参见 "正则表达式2 (RE2)语法支持"。

> 如果将命名空间筛选器添加到在还原或克隆操作之后运行的执行挂钩、并且还原或克隆源和目标位于不同的命名空间中、则命名空间筛选器仅会应用于目标命名空间。

有关自定义执行挂钩的重要注意事项

在为应用程序规划执行挂钩时，请考虑以下几点。

> 由于执行挂钩通常会减少或完全禁用其运行的应用程序的功能，因此您应始终尽量缩短自定义执行挂钩运行所需的时间。
> 如果使用关联的执行挂钩启动备份或快照操作、但随后将其取消、则在备份或快照操作已开始时、仍允许运行这些挂钩。这意味着、备份后执行挂钩中使用的逻辑不能假定备份已完成。

- 默认情况下、对于新的Astra Control部署、执行挂钩功能处于禁用状态。
  - 您需要先启用执行挂钩功能、然后才能使用执行挂钩。
  - 所有者或管理员用户可以为当前Astra Control帐户中定义的所有用户启用或禁用执行挂钩功能。请参见 [启用执行挂钩功能] 和 [禁用执行挂钩功能] 有关说明，请参见。
  - 在Astra Control升级期间、功能启用状态会保留下来。
- 执行挂钩必须使用脚本执行操作。许多执行挂钩可以引用同一个脚本。
- Astra Control要求执行挂钩使用的脚本以可执行Shell脚本的格式写入。

- 脚本大小限制为96 KB。
- Astra Control使用执行挂钩设置和任何匹配条件来确定哪些挂钩适用于快照、备份或还原操作。
- 所有执行挂钩故障均为软故障；即使某个挂钩发生故障、仍会尝试执行其他挂钩和数据保护操作。但是，如果挂机发生故障，则会在 * 活动 * 页面事件日志中记录一个警告事件。
- 要创建，编辑或删除执行挂钩，您必须是具有所有者，管理员或成员权限的用户。
- 如果执行挂机运行时间超过 25 分钟，则此挂机将失败，从而创建返回代码为不适用的事件日志条目。任何受影响的快照都将超时并标记为失败，并会生成一个事件日志条目，用于记录超时情况。
- 对于按需数据保护操作，所有挂机事件都会生成并保存在*Activity*页面事件日志中。但是、对于计划的数据保护操作、事件日志中仅会记录挂钩故障事件(计划的数据保护操作本身生成的事件仍会记录下来)。
- 如果Astra Control Center将复制的源应用程序故障转移到目标应用程序、则在故障转移完成后、为源应用程序启用的任何故障转移后执行挂钩都会在目标应用程序上运行。

> 如果您已经在Astra Control Center 23.04中运行还原后挂钩、并将Astra Control Center升级到23.07或更高版本、则在故障转移复制之后、将不再执行还原后执行挂钩。您需要为应用程序创建新的故障转移后执行挂钩。或者、您也可以将用于故障转移的现有还原后挂钩的操作类型从"还原后"更改为"故障转移后"。

执行顺序

运行数据保护操作时、执行钩事件按以下顺序发生：

1. 任何适用的自定义操作前执行挂钩都会在相应的容器上运行。您可以根据需要创建和运行任意数量的自定义操作前挂钩、但操作前这些挂钩的执行顺序既不能保证也不可配置。
2. 执行数据保护操作。
3. 任何适用的自定义操作后执行挂钩都会在相应的容器上运行。您可以根据需要创建和运行任意数量的自定义操作后挂机、但这些挂机在操作后的执行顺序既不能保证也不可配置。

如果创建多个相同类型的执行挂钩(例如、预快照)、则无法保证这些挂钩的执行顺序。但是、可以保证不同类型的挂钩的执行顺序。例如、具有所有不同类型挂钩的配置的执行顺序如下所示：

1. 已执行备份前的挂钩
2. 已执行预快照挂钩
3. 已执行后快照挂钩
4. 已执行备份后挂钩
5. 已执行还原后挂机

您可以从中的表中的第2种情形中查看此配置的示例 [确定挂钩是否会运行]。

> 在生产环境中启用执行钩脚本之前，应始终对其进行测试。您可以使用 "kubectl exec" 命令方便地测试脚本。在生产环境中启用执行挂钩后、请测试生成的快照和备份、以确保它们一致。为此、您可以将应用程序克隆到临时命名空间、还原快照或备份、然后测试应用程序。

确定挂钩是否会运行

使用下表帮助确定是否会为您的应用程序运行自定义执行挂钩。

请注意、所有高级应用程序操作都包括运行快照、备份或还原的基本操作之一。根据具体情况、克隆操作可能由这些操作的各种组合组成、因此克隆操作运行时的执行挂钩将会有所不同。

原位还原操作需要现有快照或备份、因此这些操作不会运行快照或备份挂钩。

> 如果启动并取消包含快照的备份、并且存在关联的执行挂钩、则某些挂钩可能会运行、而其他挂钩则可能不会运行。这意味着、备份后执行挂钩不能假定备份已完成。对于已取消的备份以及关联的执行挂钩、请记住以下几点：
>
> - 备份前和备份后的挂钩始终处于运行状态。
> - 如果备份包含新快照且快照已启动、则会运行预快照和后快照挂钩。
> - 如果在快照启动之前取消了备份、则不会运行预快照和后快照挂钩。

场景	操作	现有快照	现有备份	命名空间	集群	快照挂钩运行	备份挂钩运行	**Restore Hooks run**	故障转移挂钩运行
1.	克隆	不包括	不包括	新增	相同	Y	不包括	Y	不包括
2.	克隆	不包括	不包括	新增	不同	Y	Y	Y	不包括
3.	克隆或还原	Y	不包括	新增	相同	不包括	不包括	Y	不包括
4.	克隆或还原	不包括	Y	新增	相同	不包括	不包括	Y	不包括
5.	克隆或还原	Y	不包括	新增	不同	不包括	不包括	Y	不包括
6.	克隆或还原	不包括	Y	新增	不同	不包括	不包括	Y	不包括
7.	还原	Y	不包括	现有	相同	不包括	不包括	Y	不包括
8.	还原	不包括	Y	现有	相同	不包括	不包括	Y	不包括
9	Snapshot	不适用	不适用	不适用	不适用	Y	不适用	不适用	不包括
10	备份	不包括	不适用	不适用	不适用	Y	Y	不适用	不包括
11.	备份	Y	不适用	不适用	不适用	不包括	不包括	不适用	不包括
12.	故障转移	Y	不适用	由复制创建	不同	不包括	不包括	不包括	Y
13.	故障转移	Y	不适用	由复制创建	相同	不包括	不包括	不包括	Y

执行钩示例

请访问 "NetApp Verda GitHub项目" 为Apache Cassandra和Elasticsearch等常见应用程序下载真正的执行挂钩。您还可以查看示例并了解如何构建自己的自定义执行挂钩。

启用执行挂钩功能

如果您是所有者或管理员用户、则可以启用执行挂钩功能。启用此功能时、此Astra Control帐户中定义的所有用

户都可以使用执行挂钩并查看现有执行挂钩和挂钩脚本。

步骤

1. 转到 * 应用程序 * ，然后选择受管应用程序的名称。
2. 选择 * 执行挂钩 * 选项卡。
3. 选择*启用执行挂钩*。

   出现*Account*>*Feature settings (功能设置)*选项卡。

4. 在*执行挂钩*窗格中，选择设置菜单。
5. 选择 * 启用 * 。
6. 注意出现的安全警告。
7. 选择*是，启用执行挂钩*。

禁用执行挂钩功能

如果您是所有者或管理员用户、则可以对此Astra Control帐户中定义的所有用户禁用执行挂钩功能。您必须先删除所有现有的执行挂钩、然后才能禁用执行挂钩功能。请参见 [删除执行挂钩] 有关删除现有执行挂钩的说明。

步骤

1. 进入*Account*，然后选择*Feature settings (功能设置)*选项卡。
2. 选择 * 执行挂钩 * 选项卡。
3. 在*执行挂钩*窗格中，选择设置菜单。
4. 选择 * 禁用 * 。
5. 注意出现的警告。
6. Type `disable` 确认要为所有用户禁用此功能。
7. 选择*是，禁用*。

查看现有执行挂钩

您可以查看应用程序的现有自定义执行挂钩。

步骤

1. 转到 * 应用程序 * ，然后选择受管应用程序的名称。
2. 选择 * 执行挂钩 * 选项卡。

   您可以在显示的列表中查看所有已启用或已禁用的执行挂钩。您可以查看挂钩的状态、匹配的容器数量、创建时间以及运行时间(操作前或操作后)。您可以选择 + 此挂机名称旁边的图标可展开要运行它的容器列表。要查看与此应用程序的执行挂钩相关的事件日志、请转到*活动*选项卡。

查看现有脚本

您可以查看已上传的现有脚本。您还可以在此页面上查看正在使用哪些脚本以及正在使用哪些挂钩。

步骤

1. 转到*帐户*。
2. 选择*脚本*选项卡。

   您可以在此页面上查看已上传的现有脚本列表。*使用者*列显示了使用每个脚本的执行挂钩。

添加脚本

每个执行挂钩都必须使用脚本执行操作。您可以添加一个或多个可供执行挂钩引用的脚本。许多执行挂钩可以引用同一个脚本；这样、您只需更改一个脚本、即可更新多个执行挂钩。

步骤

1. 确保执行钩子功能为 enabled。
2. 转到*帐户*。
3. 选择*脚本*选项卡。
4. 选择 * 添加 * 。
5. 执行以下操作之一：
   - 上传自定义脚本。
     i. 选择 * 上传文件 * 选项。
     ii. 浏览到文件并上传。
     iii. 为脚本指定一个唯一名称。
     iv. （可选）输入其他管理员应了解的有关该脚本的任何注释。
     v. 选择*保存脚本*。
   - 从剪贴板粘贴到自定义脚本中。
     i. 选择*粘贴或类型*选项。
     ii. 选择文本字段并将脚本文本粘贴到字段中。
     iii. 为脚本指定一个唯一名称。
     iv. （可选）输入其他管理员应了解的有关该脚本的任何注释。
6. 选择*保存脚本*。

结果

新脚本将显示在*脚本*选项卡的列表中。

删除脚本

如果不再需要某个脚本、并且任何执行挂钩都不使用该脚本、则可以将其从系统中删除。

步骤

1. 转到*帐户*。
2. 选择*脚本*选项卡。
3. 选择要删除的脚本、然后在*操作*列中选择菜单。

4. 选择 * 删除 * 。

> 如果该脚本与一个或多个执行挂钩关联、则*删除*操作将不可用。要删除此脚本、请先编辑关联的执行挂钩、然后将其与其他脚本关联。

创建自定义执行挂钩

您可以为应用程序创建自定义执行挂钩、并将其添加到Astra Control中。请参见 [执行钩示例] 有关挂机示例。要创建执行挂钩，您需要拥有所有者，管理员或成员权限。

> 创建用作执行挂钩的自定义Shell脚本时、请务必在文件开头指定适当的Shell、除非您正在运行特定命令或提供可执行文件的完整路径。

步骤

1. 确保执行钩子功能为 enabled。
2. 选择 * 应用程序 * ，然后选择受管应用程序的名称。
3. 选择 * 执行挂钩 * 选项卡。
4. 选择 * 添加 * 。
5. 在*挂机详细信息*区域中：
   a. 从*操作*下拉菜单中选择操作类型、以确定何时应运行挂钩。
   b. 输入此挂钩的唯一名称。
   c. （可选）输入执行期间传递到挂机的任何参数，在输入的每个参数之后按 Enter 键以记录每个参数。
6. (可选)在*挂机筛选器详细信息*区域中、您可以添加筛选器来控制执行挂机运行在哪些容器上：
   a. 选择*添加筛选器*。
   b. 在*挂机筛选器类型*列中、从下拉菜单中选择要筛选的属性。
   c. 在*正则表达式*列中、输入要用作筛选器的正则表达式。Astra Control使用 "正则表达式2 (RE2)正则表达式语法"。

      > 如果在正则表达式字段中筛选某个属性的确切名称(例如Pod名称)而不包含其他文本、则会执行子字符串匹配。要匹配确切的名称以及仅匹配该名称、请使用精确的字符串匹配语法(例如、 `^exact_podname$`）。

   d. 要添加更多筛选器、请选择*添加筛选器*。

      > 一个执行钩的多个筛选器与一个逻辑运算符和运算符结合使用。每个执行连接最多可以有10个活动筛选器。

7. 完成后、选择*下一步*。
8. 在 * 脚本 * 区域中，执行以下操作之一：
   - 添加新脚本。
     i. 选择 * 添加 * 。
     ii. 执行以下操作之一：

        - 上传自定义脚本。
            I. 选择 * 上传文件 * 选项。
            II. 浏览到文件并上传。
            III. 为脚本指定一个唯一名称。
            IV. （可选）输入其他管理员应了解的有关该脚本的任何注释。
            V. 选择*保存脚本*。
        - 从剪贴板粘贴到自定义脚本中。
            I. 选择*粘贴或类型*选项。
            II. 选择文本字段并将脚本文本粘贴到字段中。
            III. 为脚本指定一个唯一名称。
            IV. （可选）输入其他管理员应了解的有关该脚本的任何注释。
    - 从列表中选择一个现有脚本。

      这将指示执行挂钩使用此脚本。

9. 选择 * 下一步 * 。
10. 查看执行钩配置。
11. 选择 * 添加 * 。

检查执行挂钩的状态

在快照、备份或还原操作运行完毕后、您可以检查在该操作中运行的执行挂钩的状态。您可以使用此状态信息来确定是要保持执行状态、修改执行状态还是删除执行状态。

步骤

1. 选择 * 应用程序 * ，然后选择受管应用程序的名称。
2. 选择*数据保护*选项卡。
3. 选择*快照*可查看正在运行的快照、选择*备份*可查看正在运行的备份。

   *挂机状态*显示操作完成后执行挂机运行的状态。有关详细信息、可以将鼠标悬停在状态上。例如、如果在快照期间发生执行挂机故障、则将鼠标悬停在该快照的挂机状态上可显示失败的执行挂机列表。要查看每次失败的原因、您可以查看左侧导航区域中的*活动*页面。

查看脚本使用情况

您可以在Astra Control Web UI中查看哪些执行挂钩使用特定脚本。

步骤

1. 选择 * 帐户 * 。
2. 选择*脚本*选项卡。

   脚本列表中的*使用者*列包含有关列表中每个脚本使用哪些挂钩的详细信息。

3. 在*使用者*列中选择您感兴趣的脚本的信息。

   此时将显示一个更详细的列表、其中包含正在使用此脚本的挂钩的名称以及这些挂钩配置为运行的操作类型。

编辑执行挂钩

如果要更改执行挂钩的属性、筛选器或所使用的脚本、您可以编辑该执行挂钩。要编辑执行挂钩、您需要拥有所有者、管理员或成员权限。

步骤

1. 选择 * 应用程序 * ，然后选择受管应用程序的名称。
2. 选择 * 执行挂钩 * 选项卡。
3. 在*操作*列中选择要编辑的挂钩的选项菜单。
4. 选择 * 编辑 * 。
5. 完成每个部分后、选择*下一步*进行所需的更改。
6. 选择 * 保存 * 。

禁用执行挂钩

如果要暂时阻止执行挂钩在应用程序快照之前或之后运行，可以禁用执行挂钩。要禁用执行挂钩，您需要拥有所有者，管理员或成员权限。

步骤

1. 选择 * 应用程序 * ，然后选择受管应用程序的名称。
2. 选择 * 执行挂钩 * 选项卡。
3. 在 * 操作 * 列中选择要禁用的挂机的选项菜单。
4. 选择 * 禁用 * 。

删除执行挂钩

如果您不再需要执行挂钩，则可以将其完全移除。要删除执行挂钩，您需要拥有所有者，管理员或成员权限。

步骤

1. 选择 * 应用程序 * ，然后选择受管应用程序的名称。
2. 选择 * 执行挂钩 * 选项卡。
3. 在 * 操作 * 列中选择要删除的挂机的选项菜单。
4. 选择 * 删除 * 。
5. 在显示的对话框中、键入"delete"进行确认。
6. 选择*是、删除执行钩*。

有关详细信息 ...

- "NetApp Verda GitHub项目"

使用**Astra Control Center**保护**Astra Control Center**

要更好地确保在运行Astra Control Center的Kuburenet集群上针对致命错误的故障恢复能力、请保护Astra Control Center应用程序本身。您可以使用二级Asta控制中心实例备份和还原Asta控制中心、或者如果底层存储使用ONTAP、则可以使用Asta复制。

在这些情况下、Asta Control Center的第二个实例部署和配置在不同的容错域中、并在与主Asta Control Center实例不同的第二个Kubnetes集群上运行。第二个Asta Control实例用于备份主Asta Control Center实例并可能还原该实例。还原或复制的Astra Control Center实例将继续为应用程序集群应用程序提供应用程序数据管理、并恢复对这些应用程序备份和快照的访问。

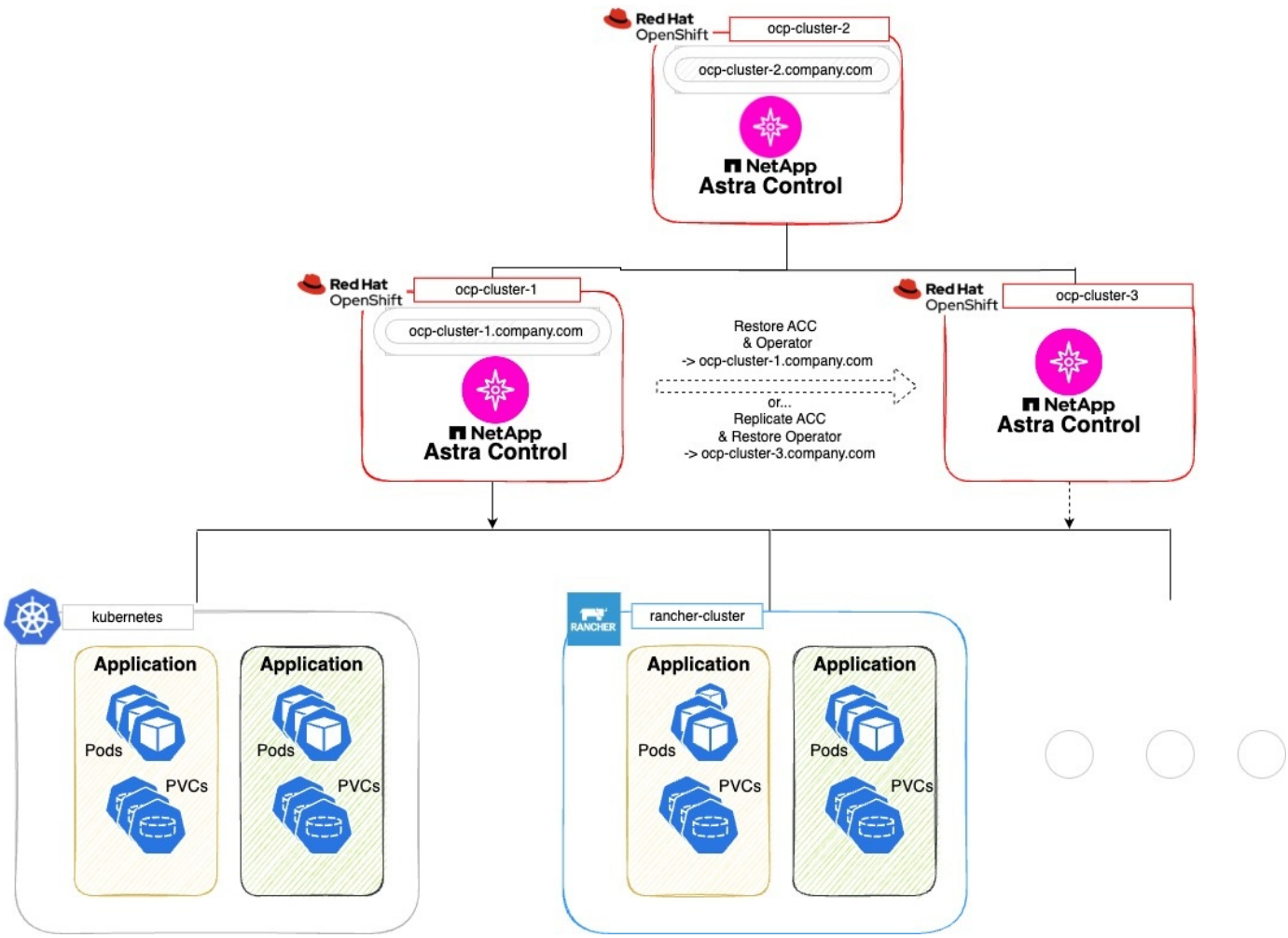
开始之前

在为Astra Control Center设置保护方案之前、请确保您具备以下条件：

- *运行主Asta Control Center实例*的Kubernetes集群：此集群托管主Asta Control Center实例、用于管理应用程序集群。
- 运行辅助**Asta Control Center**实例的主**Kubernetes**分发类型相同的第二个**Kubernetes**集群：此集群托管管理主Asta Control Center实例的Asta Control Center实例。
- *与主*具有相同Kubernetes分发类型的第三个Kubernetes集群：此集群将托管已还原或复制的Astra Control Center实例。它必须具有当前部署在主系统上的相同可用Asta Control Center命名空间。例如、如果Asta Control Center部署在命名空间中 `netapp-acc` 源集群上的命名空间 `netapp-acc` 必须可用且未被目标Kubnetes集群上的任何应用程序使用。
- **S3**兼容存储分段:每个Astra Control Center实例都有一个可访问的S3兼容对象存储分段。
- 已配置的负载平衡器：负载平衡器为Astra提供IP地址、并且必须与应用程序集群和两个S3分段建立网络连接。
- 集群满足**Astra Control Center**要求：Astra Control Center保护中使用的每个集群都满足要求 "Asta Control Center的一般要求"。

关于此任务

这些过程介绍了使用将Asta Control Center还原到新集群所需的步骤 备份和还原 或 复制。步骤基于以下示例配置：



在此示例配置中、显示了以下内容：

- 运行主**Asta Control Center**实例的**Kubernetes**集群：
  - OpenShift集群： `ocp-cluster-1`
  - Asta Control Center主实例： `ocp-cluster-1.company.com`
  - 此集群用于管理应用程序集群。
- 与运行辅助**Astra Control Center**实例的主集群具有相同**Kubernetes**分发类型的第二个**Kubernetes**集群：
  - OpenShift集群： `ocp-cluster-2`
  - Asta Control Center二级实例： `ocp-cluster-2.company.com`
  - 此集群将用于备份主Asta Control Center实例或配置复制到其他集群(在此示例中为 `ocp-cluster-3` 集群)。
- 将用于还原操作的与主**Kubernetes**分发类型相同的第三个**Kubernetes**集群：
  - OpenShift集群： `ocp-cluster-3`
  - Asta Control Center第三实例： `ocp-cluster-3.company.com`
  - 此集群将用于Asta Control Center还原或复制故障转移。

> 理想情况下、应用程序集群应位于上图中Kubernetes和randcher集群所示的三个Astra Control Center集群之外。

图中未显示：

- 所有集群都安装了ONTAP后端以及Asta三叉型或Asta控制配置程序。
- 在此配置中、OpenShift集群使用MetalLB作为负载平衡器。
- Snapshot控制器和卷SnapshotClass也会安装在所有集群上、如中所述 "前提条件"。

步骤**1**选项：备份和还原**Astra Control Center**

本操作步骤介绍了使用备份和还原将Astra控制中心还原到新集群所需的步骤。

在此示例中、Astra Control Center始终安装在下 `netapp-acc` 命名空间和操作符安装在下 `netapp-acc-operator` 命名空间。

> Asta Control Center operator也可以部署在与Asta CR相同的命名空间中、但未进行说明。

开始之前

- 您已在集群上安装主Asta Control Center。
- 您已在另一个集群上安装辅助Asta Control Center。

步骤

1. 从二级Asta Control Center实例(运行于上)管理主Asta Control Center应用程序和目标集群 `ocp-cluster-2` 集群)：
   a. 登录到辅助Asta Control Center实例。
   b. "添加主Asta Control Center集群" (`ocp-cluster-1`）。
   c. "添加目标第三个集群" (`ocp-cluster-3`)。
2. 在辅助Asta Control Center上管理Asta Control Center和Asta Control Center操作员：
   a. 从应用程序页面中、选择*定义*。
   b. 在*Define application*窗口中，输入新的应用程序名称 (`netapp-acc`）。
   c. 选择运行主Asta Control Center的集群 (`ocp-cluster-1`)。
   d. 选择 `netapp-acc` Astra Control Center的命名空间。
   e. 在Cluster Resources页面上，选中*include additional cluster-scope ResResResees*。
   f. 选择*添加包含规则*。
   g. 选择这些条目，然后选择*Add*：
      - 标签选择器：<label name>
      - 组：i扩展.k8s.io
      - 版本：V1
      - 种类：CustomResourceDefinition

h. 确认应用程序信息。

i. 选择 * 定义 * 。

选择*defin*后，对运算符重复“定义应用程序”过程 `netapp-acc-operator`)、然后选择 `netapp-acc-operator` 命名空间。

3. 备份Asta控制中心和操作员：

a. 在辅助Asta控制中心上、选择应用程序选项卡以导航到应用程序页面。

b. "备份" Astra Control Center应用程序 (`netapp-acc`）。

c. "备份" 运算符 (`netapp-acc-operator`）。

4. 备份Astra Control Center和操作员之后、使用模拟灾难恢复(DR)场景 "正在卸载Astra Control Center" 从主集群。

> 您需要将Astra控制中心还原到新集群(此操作步骤中所述的第三个Kubbernetes集群)、并对新安装的Astra控制中心使用与主集群相同的DNS。

5. 使用辅助Asta控制中心、 "还原" Asta Control Center应用程序的主实例从其备份中：

a. 选择*Applications*，然后选择Astra Control Center应用程序的名称。

b. 从“操作”列的“选项”菜单中，选择*Restore*。

c. 选择*还原到新的空间*作为还原类型。

d. 输入还原名称 (`netapp-acc`）。

e. 选择目标第三个集群 (`ocp-cluster-3`）。

f. 更新目标命名空间、使其与原始命名空间相同。

g. 在还原源页面上、选择要用作还原源的应用程序备份。

h. 选择*使用原始存储类还原*。

i. 选择*恢复所有资源*。

j. 查看还原信息，然后选择*Restore*以启动将Astra Control Center还原到目标集群的还原过程 (`ocp-cluster-3`）。应用程序进入后、还原完成 `available` 状态。

6. 在目标集群上配置Astra Control Center：

a. 打开终端并使用kubeconfig"连接到目标集群 (`ocp-cluster-3`)、其中包含已恢复的Astra控制中心。

b. 确认 `ADDRESS` Astra Control Center配置中的列引用了主系统的DNS名称：

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

响应：

```
NAME  UUID                                   VERSION    ADDRESS
READY
astra 89f4fd47-0cf0-4c7a-a44e-43353dc96ba8 24.02.0-69 ocp-cluster-
1.company.com           True
```

a. 如果 `ADDRESS` 上述响应中的字段不具有主Asta Control Center实例的FQDN、请更新此配置以引用Asta Control Center DNS：

```
kubectl edit acc -n netapp-acc
```

   i. 更改 `astraAddress` 下 `spec`：到FQDN (`ocp-cluster-1.company.com` 在本示例中)。

   ii. 保存配置。

   iii. 确认地址已更新：

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

b. 转至 恢复A作用 控制中心操作员 以完成还原过程。

第**1**步选项：使用复制保护**Astra Control Center**

本操作步骤介绍了配置所需的步骤 "Asta Control Center复制" 保护主Asta Control Center实例。

在此示例中、Astra Control Center始终安装在下 `netapp-acc` 命名空间和操作符安装在下 `netapp-acc-operator` 命名空间。

开始之前

- 您已在集群上安装主Asta Control Center。
- 您已在另一个集群上安装辅助Asta Control Center。

步骤

1. 从二级Asta Control Center实例管理主Asta Control Center应用程序和目标集群：
   a. 登录到辅助Asta Control Center实例。
   b. "添加主Asta Control Center集群" (`ocp-cluster-1`）。
   c. "添加目标第三个集群" (`ocp-cluster-3`)。
2. 在辅助Asta Control Center上管理Asta Control Center和Asta Control Center操作员：
   a. 选择*群集*，然后选择包含主Astra Control Center的群集 (`ocp-cluster-1`）。
   b. 选择*命名空间*选项卡。
   c. 选择 … `netapp-acc` 和 `netapp-acc-operator` "。
   d. 选择操作菜单，然后选择*Define as applications*。

e. 选择*在应用程序中查看*以查看定义的应用程序。

3. 配置用于复制的后端：

> 复制要求主Asta Control Center集群和目标集群 (`ocp-cluster-3`)使用不同的对等ONTAP存储后端。
> 在每个后端建立对等并添加到Astra Control后，后端将显示在“后端”页面的“发现”选项卡中。

a. "添加对等后端" 到主集群上的Astra Control Center。

b. "添加对等后端" 到目标集群上的Astra Control Center。

4. 配置复制：

a. 在应用程序屏幕上、选择 `netapp-acc` 应用程序。

b. 选择*配置复制策略*。

c. 选择 ... `ocp-cluster-3` 作为目标集群。

d. 选择存储类。

e. 输入 ... `netapp-acc` 作为目标命名空间。

f. 根据需要更改复制频率。

g. 选择 * 下一步 * 。

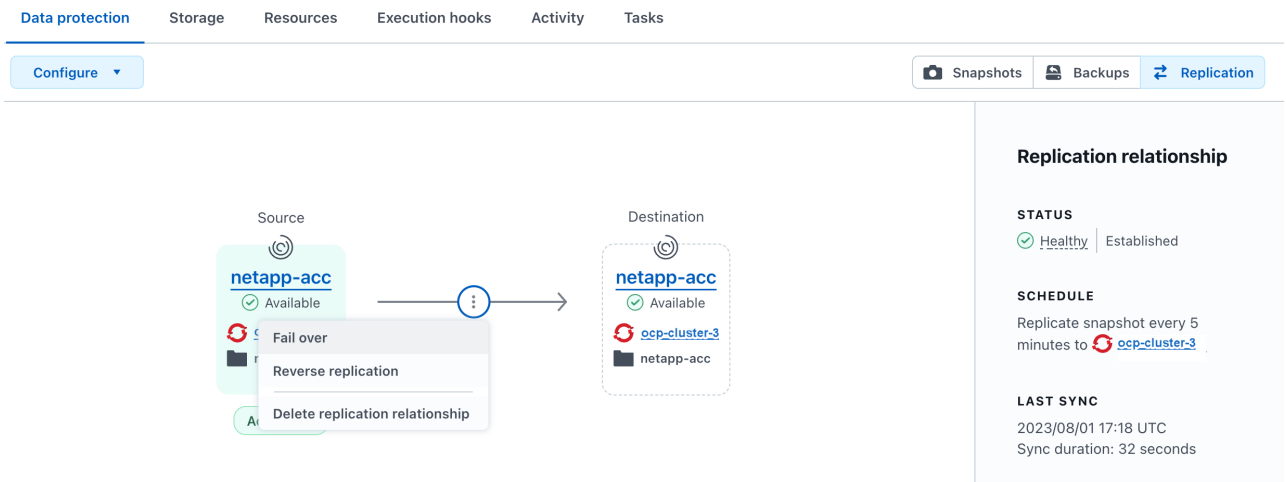
h. 确认配置正确，然后选择*Save*。

复制关系将从过渡 `Establishing` to `Established`。处于活动状态时、此复制将每五分钟进行一次、直到删除复制配置为止。

5. 如果主系统损坏或无法再访问、请将复制故障转移到另一个集群：

> 确保目标集群未安装Astra Control Center、以确保成功进行故障转移。

a. 选择垂直省略号图标并选择*故障转移*。



b. 确认详细信息并选择*故障转移*以开始故障转移过程。

复制关系状态将更改为 `Failing over` 然后 `Failed over` 完成后。

6. 完成故障转移配置：

   a. 打开终端并使用第三个集群的kubeconfig (`ocp-cluster-3`）。此集群现在已安装Asta Control Center。

   b. 确定第三个集群上的Asta Control Center FQDN (`ocp-cluster-3`）。

   c. 更新配置以引用Astra Control Center DNS：

```
kubectl edit acc -n netapp-acc
```

      i. 更改 `astraAddress` 下 `spec`：使用FQDN (`ocp-cluster-3.company.com`)。

      ii. 保存配置。

      iii. 确认地址已更新：

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

   d. 确认所有必需的traefik CRD都存在：

```
kubectl get crds | grep traefik
```

   所需的traefik CRD：

```
ingressroutes.traefik.containo.us
ingressroutes.traefik.io
ingressroutetcps.traefik.containo.us
ingressroutetcps.traefik.io
ingressrouteudps.traefik.containo.us
ingressrouteudps.traefik.io
middlewares.traefik.containo.us
middlewares.traefik.io
middlewaretcps.traefik.containo.us
middlewaretcps.traefik.io
serverstransports.traefik.containo.us
serverstransports.traefik.io
tlsoptions.traefik.containo.us
tlsoptions.traefik.io
tIsstores.traefik.containo.us
tIsstores.traefik.io
traefikservices.traefik.containo.us
traefikservices.traefik.io
```

   a. 如果上述部分CRD缺失：

i. 转至 "Traefik文档"。

ii. 将"定义"区域复制到文件中。

iii. 应用更改：

```
kubectl apply -f <file name>
```

iv. 重新启动traefik：

```
kubectl get pods -n netapp-acc | grep -e "traefik" | awk '{print
$1}' | xargs kubectl delete pod -n netapp-acc
```

b. 转至 恢复A作用 控制中心操作员 以完成还原过程。

第**2**步：恢复**Asta Control Center**操作员

使用辅助Asta控制中心、从备份中还原主Asta控制中心操作员。目标命名空间必须与源命名空间相同。如果从主源集群中删除了Astra Control Center、则仍会存在备份以执行相同的还原步骤。

步骤

1. 选择*应用程序*，然后选择运营商应用程序的名称 (`netapp-acc-operator`）。
2. 从“操作”列的“选项”菜单中，选择*Restore*
3. 选择*还原到新的空间*作为还原类型。
4. 选择目标第三个集群 (`ocp-cluster-3`）。
5. 将命名空间更改为与主源集群关联的命名空间相同 (`netapp-acc-operator`）。
6. 选择先前创建的备份作为还原源。
7. 选择*使用原始存储类还原*。
8. 选择*恢复所有资源*。
9. 查看详细信息，然后单击*Restore*开始还原过程。

   "应用程序"页面显示了正在还原到目标第三个集群的A作用 力控制中心操作员 (`ocp-cluster-3`）。此过程完成后、此状态将显示为 `Available`。十分钟内、页面上的DNS地址应已解析。

结果

现在、目标第三个集群上可以使用Astra Control Center、其注册的集群以及具有其快照和备份的受管应用程序 (`ocp-cluster-3`）。您在原始实例上设置的任何保护策略也会位于新实例上。您可以继续创建计划内或按需备份和快照。

故障排除

确定系统运行状况以及保护过程是否成功。

- **Pod**未运行：确认所有Pod均已启动且正在运行：

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

如果中有一些Pod `CrashLookBackOff` 陈述并重新启动它们、它们应过渡到 `Running` 状态。

- 确认系统状态：确认Astra Control Center系统处于 `ready` 状态：

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

响应：

```
NAME  UUID                                   VERSION    ADDRESS
READY
astra 89f4fd47-0cf0-4c7a-a44e-43353dc96ba8 24.02.0-69 ocp-cluster-
1.company.com           True
```

- 确认部署状态：显示Astra Control Center部署信息以确认 `Deployment State` 为 `Deployed`。

```
kubectl describe acc astra -n netapp-acc
```

- **Restored Asta Control Center UI**返回**404**错误:如果在选择时发生这种情况 `AccTraefik` 作为入口选项、选中 Traefik CRD 以确保所有这些组件均已安装。

监控应用程序和集群运行状况

查看应用程序和集群运行状况摘要

选择 * 信息板 * 可查看应用程序，集群，存储后端及其运行状况的高级视图。

这些数字或状态不仅仅是静态数字或状态，您可以逐层查看。例如，如果应用程序未得到完全保护，您可以将鼠标悬停在图标上以确定哪些应用程序未得到完全保护，这包括原因。

应用程序区块

"* 应用程序 * " 图块可帮助您确定以下内容：

- 您当前使用 Astra 管理的应用程序数量。
- 这些受管应用程序是否运行正常。
- 应用程序是否受到完全保护（如果有最新备份可用，则会对其进行保护）。
- 已发现但尚未管理的应用程序的数量。

  理想情况下，此数字为零，因为您可能会在发现应用程序后对其进行管理或忽略。然后，您将监控信息板上发现的应用程序的数量，以确定开发人员何时向集群添加新应用程序。

集群图块

"* 集群 * " 图块提供了有关使用 Astra 控制中心管理的集群运行状况的类似详细信息，您可以像使用应用程序一样深入查看以获取更多详细信息。

存储后端图块

"Storage Backends* " 图块提供的信息可帮助您确定存储后端的运行状况，其中包括：

- 管理的存储后端数量
- 这些受管后端是否运行正常
- 后端是否受到完全保护
- 已发现但尚未管理的后端数量。

查看集群运行状况并管理存储类

添加要由 Astra 控制中心管理的集群后，您可以查看有关集群的详细信息，例如集群的位置，工作节点，永久性卷和存储类。您还可以更改受管集群的默认存储类。

查看集群运行状况和详细信息

您可以查看有关集群的详细信息、例如集群的位置、工作节点、永久性卷和存储类。

步骤

1. 在 Astra 控制中心 UI 中，选择 * 集群 * 。
2. 在 * 集群 * 页面上，选择要查看其详细信息的集群。

> 如果集群位于中 `removed` 状态虽然集群和网络连接运行状况良好(外部尝试使用Kubernetes API访问集群成功)、但您提供给Astra Control的kubeconfig可能不再有效。这可能是由于集群上的证书轮换或到期造成的。要更正此问题描述，请使用在 Astra Control 中更新与集群关联的凭据 "Astra Control API"。

3. 查看 * 概述 * ， * 存储 * 和 * 活动 * 选项卡上的信息，找到您要查找的信息。
   - * 概述 * ： 有关工作节点的详细信息，包括其状态。
   - * 存储 * ： 与计算关联的永久性卷，包括存储类和状态。
   - * 活动 * ： 显示与集群相关的活动。

> 您还可以从 Astra 控制中心 * 信息板 * 开始查看集群信息。在 * 资源摘要 * 下的 * 集群 * 选项卡上，您可以选择受管集群，此操作将转到 * 集群 * 页面。进入 * 集群 * 页面后，请按照上述步骤进行操作。

更改默认存储类

您可以更改集群的默认存储类。当Astra Control管理集群时、它会跟踪集群的默认存储类。

请勿使用kubectl命令更改存储类。请改用此操作步骤。如果使用kubectl进行更改、则Astra Control将还原这些更改。

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*。

查看应用程序的运行状况和详细信息

开始管理应用程序后、Astra Control会提供有关该应用程序的详细信息、您可以通过这些信息确定其通信状态(Astra Control是否可以与该应用程序通信)、保护状态(是否在发生故障时受到全面保护)、Pod、永久性存储等。

步骤

1. 在 Astra 控制中心 UI 中，选择 * 应用程序 * ，然后选择应用程序的名称。
2. 查看相关信息。

   应用程序状态

   提供反映Asta Control是否可以与应用程序通信的状态。

   - 应用程序保护状态：提供应用程序的保护程度状态：
     - * 完全保护 *：应用程序具有一个活动备份计划，并且备份成功完成不到一周
     - * 部分保护 *：应用程序具有活动备份计划，活动快照计划或成功备份或快照
     - * 未受保护 *：既不受完全保护也不受部分保护的应用程序。

       You can't be Fully protected until you have a recent backup 。这一点非常重要，因为备份存储在对象存储中，而不是永久性卷。如果发生故障或意外事件会擦除集群及其永久性存储，则需要备份才能恢复。快照无法让您恢复。
   - 概述：有关与应用程序关联的Pod的状态的信息。
   - 数据保护：用于配置数据保护策略以及查看现有快照和备份。
   - 存储：显示应用程序级别的永久性卷。从 Kubernetes 集群的角度来看，永久性卷的状态。
   - 资源：用于验证正在备份和管理哪些资源。
   - 活动：显示与应用程序相关的活动。

   您还可以从 Astra 控制中心 * 信息板 * 开始查看应用程序信息。在 * 资源摘要 * 下的 * 应用程序 * 选项卡上，您可以选择受管应用程序，此操作将转到 * 应用程序 * 页面。进入 * 应用程序 * 页面后，请按照上述步骤进行操作。

管理您的帐户

管理本地用户和角色

您可以使用Astra Control UI添加、删除和编辑Astra Control Center安装的用户。您可以使用 Astra Control UI 或 "Astra Control API" 以管理用户。

您还可以使用LDAP对选定用户执行身份验证。

使用 **LDAP**

LDAP是一种用于访问分布式目录信息的行业标准协议、也是企业身份验证的常见选择。您可以将Astra控制中心连接到LDAP服务器、以便对选定的Astra控制用户执行身份验证。从较高层面来看、该配置涉及将Astra与LDAP集成、并定义与LDAP定义对应的Astra Control用户和组。您可以使用Astra Control API或Web UI配置LDAP身份验证以及LDAP用户和组。有关详细信息、请参见以下文档：

- "使用Astra Control API管理远程身份验证和用户"
- "使用Astra Control UI管理远程用户和组"
- "使用Astra Control UI管理远程身份验证"

添加用户

帐户所有者和管理员可以向 Astra 控制中心安装添加更多用户。

步骤

1. 在 * 管理帐户 * 导航区域中，选择 * 帐户 * 。
2. 选择 * 用户 * 选项卡。
3. 选择 * 添加用户 * 。
4. 输入用户的名称，电子邮件地址和临时密码。

   用户需要在首次登录时更改密码。

5. 选择具有适当系统权限的用户角色。

   每个角色都提供以下权限：

   - * 查看器 * 可以查看资源。
   - " 成员 * " 具有 " 查看器 " 角色权限，可以管理应用程序和集群，取消管理应用程序以及删除快照和备份。
   - * 管理员 * 具有成员角色权限，可以添加和删除除所有者之外的任何其他用户。
   - * 所有者 * 具有管理员角色权限，可以添加和删除任何用户帐户。

6. 要为具有成员或查看器角色的用户添加约束，请启用 * 将角色限制为约束条件 * 复选框。

   有关添加约束的详细信息、请参见 "管理本地用户和角色"。

7. 选择 * 添加 * 。

管理密码

您可以在 Astra 控制中心管理用户帐户的密码。

更改密码

您可以随时更改用户帐户的密码。

步骤

1. 选择屏幕右上角的用户图标。
2. 选择 * 配置文件 * 。
3. 从选项菜单的 * 操作 * 列中选择 * 更改密码 * 。
4. 输入符合密码要求的密码。
5. 再次输入密码进行确认。
6. 选择 * 更改密码 * 。

重置其他用户的密码

如果您的帐户具有管理员或所有者角色权限，则可以重置其他用户帐户以及您自己的帐户的密码。重置密码时，您需要分配一个临时密码，用户必须在登录时更改此密码。

步骤

1. 在 * 管理帐户 * 导航区域中，选择 * 帐户 * 。
2. 选择 * 操作 * 下拉列表。
3. 选择 * 重置密码 * 。
4. 输入符合密码要求的临时密码。
5. 再次输入密码进行确认。

   用户下次登录时，系统将提示用户更改密码。

6. 选择 * 重置密码 * 。

删除用户

具有所有者或管理员角色的用户可以随时从帐户中删除其他用户。

步骤

1. 在 * 管理帐户 * 导航区域中，选择 * 帐户 * 。
2. 在 * 用户 * 选项卡中，选中要删除的每个用户所在行中的复选框。
3. 从选项菜单的 * 操作 * 列中，选择 * 删除用户 / 秒 * 。
4. 出现提示时，键入单词 "remove" 并选择 * 是，删除用户 * 以确认删除。

结果

Astra 控制中心从帐户中删除用户。

管理角色

您可以通过添加命名空间限制并将用户角色限制为这些限制来管理角色。这样，您就可以控制对组织内资源的访问。您可以使用 Astra Control UI 或 "Astra Control API" 以管理角色。

向角色添加命名空间限制

管理员或所有者用户可以向成员或查看器角色添加命名空间限制。

步骤

1. 在 * 管理帐户 * 导航区域中，选择 * 帐户 * 。
2. 选择 * 用户 * 选项卡。
3. 在 * 操作 * 列中，为具有成员或查看器角色的用户选择菜单按钮。
4. 选择 * 编辑角色 * 。
5. 启用 * 将角色限制为约束条件 * 复选框。

   此复选框仅适用于 " 成员 " 或 " 查看器 " 角色。您可以从 * 角色 * 下拉列表中选择其他角色。

6. 选择 * 添加约束 * 。

   您可以按命名空间或命名空间标签查看可用约束的列表。

7. 在 * 约束类型 * 下拉列表中，根据命名空间的配置方式选择 * Kubernetes 命名空间 * 或 * Kubernetes 命名空间标签 * 。
8. 从列表中选择一个或多个命名空间或标签，以构成一个限制，将角色限制为这些命名空间。
9. 选择 * 确认 * 。

   "* 编辑角色 * " 页面将显示您为此角色选择的约束列表。

10. 选择 * 确认 * 。

    在 * 帐户 * 页面上，您可以在 * 角色 * 列中查看任何成员或查看器角色的限制。

> 如果为某个角色启用了限制并选择了 * 确认 * 而未添加任何限制，则该角色将被视为具有完全限制（该角色将被拒绝访问分配给命名空间的任何资源）。

从角色中删除命名空间限制

管理员或所有者用户可以从角色中删除命名空间限制。

步骤

1. 在 * 管理帐户 * 导航区域中，选择 * 帐户 * 。
2. 选择 * 用户 * 选项卡。
3. 在 * 操作 * 列中，为具有成员或查看器角色且具有活动约束的用户选择菜单按钮。
4. 选择 * 编辑角色 * 。

"* 编辑角色 " 对话框显示角色的活动约束。

5. 选择需要删除的约束右侧的 * X * 。
6. 选择 * 确认 * 。

有关详细信息 ...

- "用户角色和命名空间"

管理远程身份验证

LDAP是一种用于访问分布式目录信息的行业标准协议、也是企业身份验证的常见选择。您可以将Astra控制中心连接到LDAP服务器、以便对选定的Astra控制用户执行身份验证。

从较高层面来看、该配置涉及将Astra与LDAP集成、并定义与LDAP定义对应的Astra Control用户和组。您可以使用Astra Control API或Web UI配置LDAP身份验证以及LDAP用户和组。

> Astra Control Center使用用户登录属性(在启用远程身份验证时配置)来搜索和跟踪远程用户。对于您希望在Astra Control Center中显示的任何远程用户、此字段中必须存在电子邮件地址("mail")或用户主体名称("userPrincipalName")的属性。此属性在Astra Control Center中用作用户名以进行身份验证，并在搜索远程用户时使用。

添加用于**LDAPS**身份验证的证书

为LDAP服务器添加专用TLS证书、以便在使用LDAPS连接时、Astra控制中心可以向LDAP服务器进行身份验证。您只需要执行一次此操作、或者在安装的证书过期时执行此操作。

步骤

1. 转到*帐户*。
2. 选择*证书*选项卡。
3. 选择 * 添加 * 。
4. 上传 `.pem` 将文件内容归档或粘贴到剪贴板中。
5. 选中*可信*复选框。
6. 选择*添加证书*。

启用远程身份验证

您可以启用LDAP身份验证并配置Astra Control与远程LDAP服务器之间的连接。

开始之前

如果您计划使用LDAPS、请确保将LDAP服务器的专用TLS证书安装在Astra控制中心中、以便Astra控制中心能够向LDAP服务器进行身份验证。请参见 添加用于LDAPS身份验证的证书 有关说明，请参见。

步骤

1. 转至*帐户>连接*。
2. 在*远程身份验证*窗格中、选择配置菜单。

3. 选择 * 连接 * 。
4. 输入服务器IP地址、端口和首选连接协议(LDAP或LDAPS)。

> 作为最佳实践、请在与LDAP服务器连接时使用LDAPS。在连接到LDAPS之前、您需要在Astra控制中心安装LDAP服务器的专用TLS证书。

5. 以电子邮件格式输入服务帐户凭据(administrator@example.com)。在与LDAP服务器连接时、Astra Control将使用这些凭据。
6. 在*用户匹配*部分，执行以下操作：
   a. 输入基本DN和相应的用户搜索筛选器、以便从LDAP服务器检索用户信息时使用。
   b. (可选)如果目录使用用户登录属性 `userPrincipalName` 而不是 `mail`、输入 `userPrincipalName` 在“用户登录属性”字段的正确属性中。
7. 在*组匹配*部分中、输入组搜索基础DN和相应的自定义组搜索筛选器。

> 请务必对*用户匹配*和*组匹配*使用正确的基本可分辨名称(DN)和适当的搜索筛选器。基础DN用于指示Astra Control在目录树的哪个级别开始搜索、而搜索筛选器用于限制目录树Astra Control搜索的各个部分。

8. 选择 * 提交 * 。

结果

与LDAP服务器建立连接后、远程身份验证*窗格状态将移至*待定、然后移至*已连接*。

禁用远程身份验证

您可以暂时禁用与LDAP服务器的活动连接。

> 禁用与LDAP服务器的连接时、将保存所有设置、并保留从该LDAP服务器添加到Astra Control中的所有远程用户和组。您可以随时重新连接到此LDAP服务器。

步骤

1. 转至*帐户>连接*。
2. 在*远程身份验证*窗格中、选择配置菜单。
3. 选择 * 禁用 * 。

结果

"远程身份验证"窗格状态将移至"*已禁用"。所有远程身份验证设置、远程用户和远程组都会保留下来、您可以随时重新启用连接。

编辑远程身份验证设置

如果禁用了与LDAP服务器的连接或*远程身份验证*窗格处于"连接错误"状态、则可以编辑配置设置。

> 如果*远程身份验证*窗格处于"已禁用"状态、则无法编辑LDAP服务器URL或IP地址。您需要 [断开远程身份验证] 第一个。

步骤

1. 转至*帐户>连接*。
2. 在*远程身份验证*窗格中、选择配置菜单。
3. 选择 * 编辑 * 。
4. 进行必要的更改、然后选择*编辑*。

断开远程身份验证

您可以从LDAP服务器断开连接、并从Astra Control中删除配置设置。

> 如果您是LDAP用户并断开连接、则会话将立即结束断开与LDAP服务器的连接后、该LDAP服务器的所有配置设置以及从该LDAP服务器添加的任何远程用户和组都会从Astra Control中删除。

步骤

1. 转至*帐户>连接*。
2. 在*远程身份验证*窗格中、选择配置菜单。
3. 选择*断开连接*。

结果

"远程身份验证"窗格状态将移至"*已断开连接"。远程身份验证设置、远程用户和远程组将从Astra Control中删除。

管理远程用户和组

如果您已在Astra Control系统上启用LDAP身份验证、则可以搜索LDAP用户和组、并将其包含在系统的已批准用户中。

添加远程用户

帐户所有者和管理员可以向Astra Control添加远程用户。Astra Control Center支持多达10、000个LDAP远程用户。

> Astra Control Center使用用户登录属性(在启用远程身份验证时配置)来搜索和跟踪远程用户。对于您希望在Astra Control Center中显示的任何远程用户、此字段中必须存在电子邮件地址("mail")或用户主体名称("userPrincipalName")的属性。此属性在Astra Control Center中用作用户名以进行身份验证，并在搜索远程用户时使用。

> 如果系统中已存在具有相同电子邮件地址(基于"mail"或"user主体名称"属性)的本地用户、则无法添加远程用户。要将此用户添加为远程用户、请先从系统中删除此本地用户。

步骤

1. 转到*帐户*区域。
2. 选择*用户和组*选项卡。
3. 在页面最右侧、选择*远程用户*。
4. 选择 * 添加 * 。

5. 或者、也可以通过在*按电子邮件筛选*字段中输入用户的电子邮件地址来搜索LDAP用户。
6. 从列表中选择一个或多个用户。
7. 为用户分配角色。

> 如果您为用户和用户组分配不同的角色、则优先使用较为宽松的角色。

8. (可选)为此用户分配一个或多个命名空间约束、然后选择*将角色限制为约束条件*以强制实施这些限制。您可以通过选择*添加约束*来添加新的命名空间约束。

> 如果通过LDAP组成员资格为用户分配了多个角色、则只有最宽松角色中的限制才会生效。例如、如果具有本地查看器角色的用户加入了绑定到成员角色的三个组、则成员角色的约束之和将生效、而查看器角色的任何约束将被忽略。

9. 选择 * 添加 * 。

结果

新用户将显示在远程用户列表中。在此列表中、您可以查看用户的活动约束、并从*操作*菜单管理用户。

添加远程组

要一次性添加多个远程用户、帐户所有者和管理员可以向Astra Control添加远程组。添加远程组时、该组中的所有远程用户均可登录到Astra Control、并继承与该组相同的角色。

Astra Control Center最多支持5、000个LDAP远程组。

步骤

1. 转到*帐户*区域。
2. 选择*用户和组*选项卡。
3. 在页面最右侧、选择*远程组*。
4. 选择 * 添加 * 。

   在此窗口中、您可以看到Astra Control从目录中检索到的LDAP组的公用名和可分辨名称列表。

5. 或者、也可以在*按公用名筛选*字段中输入组的公用名来搜索LDAP组。
6. 从列表中选择一个或多个组。
7. 为组分配角色。

> 您选择的角色将分配给此组中的所有用户。如果您为用户和用户组分配不同的角色、则优先使用较为宽松的角色。

8. (可选)为此组分配一个或多个命名空间约束、然后选择*将角色限制为约束条件*以强制实施这些限制。您可以通过选择*添加约束*来添加新的命名空间约束。

- 如果要访问的资源属于安装了最新**Astra Connector**的集群：通过LDAP组成员资格为用户分配多个角色时，这些角色的约束条件将合并在一起。例如、如果具有本地查看器角色的用户加入绑定到成员角色的三个组、则该用户现在可以以查看器角色访问原始资源、并以成员角色访问通过组成员资格获得的资源。
- 如果要访问的资源属于未安装**Astra Connector**的集群：通过LDAP组成员资格为用户分配多个角色时，只有最宽松角色的限制才会生效。

9. 选择 * 添加 * 。

结果

新组将显示在远程组列表中。此组中的远程用户不会显示在远程用户列表中、直到每个远程用户都登录为止。在此列表中、您可以查看有关该组的详细信息、并从*操作*菜单管理该组。

查看和管理通知

操作完成或失败时， Astra 会向您发出通知。例如，如果应用程序的备份成功完成，您将看到通知。

您可以从界面右上角管理这些通知：

步骤

1. 选择右上角的未读通知数量。
2. 查看通知，然后选择 * 标记为已读 * 或 * 显示所有通知 * 。

   如果选择 * 显示所有通知 * ，则会加载通知页面。

3. 在 * 通知 * 页面上，查看通知，选择要标记为已读的通知，选择 * 操作 * 并选择 * 标记为已读 * 。

添加和删除凭据

随时从您的帐户中添加和删除本地私有云提供商的凭据，例如 ONTAP S3 ，使用 OpenShift 管理的 Kubernetes 集群或非受管 Kubernetes 集群。Astra 控制中心使用这些凭据来发现 Kubernetes 集群和集群上的应用程序，并代表您配置资源。

请注意， Astra 控制中心中的所有用户都共享相同的凭据集。

添加凭据

您可以在管理集群时向 Astra 控制中心添加凭据。要通过添加新集群来添加凭据、请参见 "添加 Kubernetes 集群"。

如果创建自己的kubeconfig,则只能在其中定义*on*上下文元素。请参见 "Kubernetes 文档" 有关创建kubeconfig.文件的信息、请参见。

删除凭据

随时从帐户中删除凭据。您只能在之后删除凭据 "取消管理所有关联集群"。

> 您添加到 Astra 控制中心的第一组凭据始终在使用中，因为 Astra 控制中心使用这些凭据向备份存储分段进行身份验证。最好不要删除这些凭据。

步骤

1. 选择 * 帐户 * 。
2. 选择 * 凭据 * 选项卡。
3. 在 * 状态 * 列中选择要删除的凭据的选项菜单。
4. 选择 * 删除 * 。
5. 键入单词 "remove" 确认删除，然后选择 * 是，删除凭据 * 。

结果

Astra 控制中心将从帐户中删除凭据。

监控帐户活动

您可以在 Astra Control 帐户中查看有关活动的详细信息。例如，邀请新用户时，添加集群时或创建快照时。您还可以将帐户活动导出到 CSV 文件。

在 **Astra Control** 中查看所有帐户活动

1. 选择 * 活动 * 。
2. 使用筛选器缩小活动列表的范围，或者使用搜索框准确查找所需内容。
3. 选择 * 导出到 CSV* 将您的帐户活动下载到 CSV 文件。

查看特定应用程序的帐户活动

1. 选择 * 应用程序 * ，然后选择应用程序的名称。
2. 选择 * 活动 * 。

查看集群的帐户活动

1. 选择 * 集群 * ，然后选择集群的名称。
2. 选择 * 活动 * 。

采取措施解决需要关注的事件

1. 选择 * 活动 * 。
2. 选择需要关注的事件。
3. 选择 * 执行操作 * 下拉选项。

   从此列表中，您可以查看可能采取的更正操作，查看与问题描述 相关的文档，并获得支持以帮助解决问题描述 。

更新现有许可证

您可以将评估版许可证转换为完整许可证，也可以使用新许可证更新现有评估版许可证或完整许可证。如果您没有完整的许可证，请与 NetApp 销售联系人联系以获取完整的许可证和序列号。您可以使用Astra控制中心UI或 "Astra Control API" 更新现有许可证。

步骤

1. 登录到 "NetApp 支持站点"。
2. 访问 Astra 控制中心下载页面，输入序列号，然后下载完整的 NetApp 许可证文件（ NLF ）。
3. 登录到 Astra 控制中心 UI 。
4. 从左侧导航栏中，选择 * 帐户 * > * 许可证 * 。
5. 在 * 帐户 * > * 许可证 * 页面中，选择现有许可证的状态下拉菜单，然后选择 * 替换 * 。
6. 浏览到您下载的许可证文件。
7. 选择 * 添加 * 。
    - 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。

有关详细信息 ...

- "Astra 控制中心许可"

管理存储分段

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，则对象存储分段提供程序至关重要。使用 Astra 控制中心，添加一个对象存储提供程序作为应用程序的集群外备份目标。

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 Amazon Simple Storage Service （ S3 ）存储分段提供商之一：

- NetApp ONTAP S3
- NetApp StorageGRID S3
- Microsoft Azure
- 通用 S3

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

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

存储分段可以处于以下状态之一：

- Pending ：存储分段已计划进行发现。

- Available ： 存储分段可供使用。
- Removed：当前无法访问存储分段。

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

您可以执行以下与管理存储分段相关的任务：

- "添加存储分段"
- [编辑存储分段]
- [设置默认分段]
- [轮换或删除存储分段凭据]
- [删除存储分段]
- "[技术预览使用自定义资源管理存储分段"]

> Astra 控制中心中的 S3 存储分段不会报告可用容量。在备份或克隆由 Astra 控制中心管理的应用程序之前，请检查 ONTAP 或 StorageGRID 管理系统中的存储分段信息。

编辑存储分段

您可以更改存储分段的访问凭据信息，并更改选定存储分段是否为默认存储分段。

> 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。请参见 "发行说明"。

步骤

1. 从左侧导航栏中、选择*分段*。
2. 从菜单的*操作*列中、选择*编辑*。
3. 更改存储分段类型以外的任何信息。

   > 您无法修改存储分段类型。

4. 选择 * 更新 * 。

设置默认分段

在集群间执行克隆时、Astra Control需要一个默认分段。按照以下步骤为所有集群设置默认存储分段。

步骤

1. 转至*云实例*。
2. 在列表中的*操作*列中为云实例选择菜单。
3. 选择 * 编辑 * 。
4. 在*分段*列表中、选择要用作默认分段的分段。

5. 选择 * 保存 * 。

轮换或删除存储分段凭据

Astra Control使用存储分段凭据获取访问权限、并为S3存储分段提供机密密钥、以便Astra控制中心可以与存储分段进行通信。

轮换存储分段凭据

如果要轮换凭据、请在维护窗口中没有正在进行的备份(计划备份或按需备份)时轮换凭据。

编辑和轮换凭据的步骤

1. 从左侧导航栏中、选择*分段*。
2. 从选项菜单的 * 操作 * 列中，选择 * 编辑 * 。
3. 创建新凭据。
4. 选择 * 更新 * 。

删除存储分段凭据

只有在已将新凭据应用于存储分段或存储分段不再处于活动状态时、才应删除存储分段凭据。

> 添加到 Astra Control 的第一组凭据始终处于使用状态，因为 Astra Control 使用这些凭据对备份存储分段进行身份验证。如果存储分段正在使用中、请勿删除这些凭据、因为这会导致备份失败和备份不可用。

> 如果删除了活动存储分段凭据、请参见 "对删除存储分段凭据进行故障排除"。

有关如何使用Astra Control API删除S3凭据的说明、请参见 "Astra Automation 和 API 信息"。

删除存储分段

您可以删除不再使用或运行状况不佳的存储分段。您可能需要执行此操作以使对象存储配置简单且最新。

> - 您不能删除默认存储分段。如果要删除此存储分段，请先选择另一个存储分段作为默认存储。
> - 在"一次写入、多次读取"(WORM)分段的云提供程序保留期限到期之前、您不能删除该分段。WORm分段名称旁用"已锁定"表示。

- 您不能删除默认存储分段。如果要删除此存储分段，请先选择另一个存储分段作为默认存储。

开始之前

- 开始之前，应检查以确保此存储分段没有正在运行或已完成的备份。
- 您应进行检查，以确保存储分段未在任何活动保护策略中使用。

如果存在、您将无法继续。

步骤

1. 从左侧导航栏中，选择 * 分段器 * 。
2. 从 * 操作 * 菜单中，选择 * 删除 * 。

> Astra Control 可首先确保没有使用存储分段进行备份的计划策略，并且要删除的存储分段中没有活动备份。

3. 键入 "remove" 确认此操作。
4. 选择 * 是，删除存储分段 * 。

[技术预览]使用自定义资源管理存储分段

您可以使用应用程序集群上的Astra Control自定义资源(CR)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。如果使用的是自定义资源方法、则应用程序快照功能需要一个存储分段。

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

Asta Control的存储分段自定义资源称为AppVault。此CR包含在保护操作中使用存储分段所需的配置。

开始之前

- 确保您有一个可从Astra Control Center管理的集群访问的存储分段。
- 确保您具有此存储分段的凭据。
- 确保存储分段为以下类型之一：
  - NetApp ONTAP S3
  - NetApp StorageGRID S3
  - Microsoft Azure
  - 通用 S3

> Amazon Web Services (AWS)使用通用S3存储分段类型。

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

步骤

1. 创建自定义资源(CR)文件并将其命名为(例如、 `astra-appvault.yaml`）。
2. 配置以下属性：
   - * metadata.name*:_(必需)_ AppVault自定义资源的名称。
   - **spec.prefix**：_(可选)_一个路径、该路径前缀为存储在AppVault中的所有实体的名称。
   - 。**spec.providerConfig**:_(必需)_用于存储使用指定提供程序访问AppVault所需的配置。
   - * spec.providerCredentials*:_(必需)_存储使用指定提供程序访问AppVault所需的任何凭据的引用。
     - * spec.providerCredentials.valueFromSecret*:_(可选)_表示凭据值应来自机密。
       - **key**:_(如果使用了valueFroMSecret)密钥的有效密钥_。

    - **name**:_(如果使用valueF物品 密钥，则为必需项)_包含此字段值的机密的名称。必须位于同一命名空间中。
  - * spec.providerType*:_(必需)_用于确定提供备份的内容；例如、NetApp ONTAP S3或Microsoft Azure。

    YAML示例：

```yaml
apiVersion: astra.netapp.io/v1
kind: AppVault
metadata:
  name: astra-appvault
spec:
  providerType: generic-s3
  providerConfig:
    path: testpath
    endpoint: 192.168.1.100:80
    bucketName: bucket1
    secure: "false"
  providerCredentials:
    accessKeyID:
      valueFromSecret:
        name: s3-creds
        key: accessKeyID
    secretAccessKey:
      valueFromSecret:
        name: s3-creds
        key: secretAccessKey
```

3. 在您填充之后 `astra-appvault.yaml` 使用正确值的文件、应用CR：

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

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

了解更多信息

- "使用 Astra Control API"

管理存储后端

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。

有关如何使用 Astra Control API 管理存储后端的说明，请参见 "Astra Automation 和 API 信息"。

您可以完成以下与管理存储后端相关的任务：

- "添加存储后端"
- [查看存储后端详细信息]
- [编辑存储后端身份验证详细信息]
- [管理已发现的存储后端]
- [取消管理存储后端]
- [删除存储后端]

查看存储后端详细信息

您可以从信息板或后端选项查看存储后端信息。

从信息板查看存储后端详细信息

步骤

1. 从左侧导航栏中选择 * 信息板 * 。
2. 查看信息板中显示状态的存储后端面板：
    - * 运行状况不正常 * ： 存储未处于最佳状态。这可能是由于延迟问题描述或应用程序因容器问题描述等原因而降级。
    - * 所有运行状况均正常 * ： 存储已进行管理并处于最佳状态。
    - * 已发现 * ： 存储已被发现，但未由 Astra Control 管理。

从后端选项查看存储后端详细信息

查看有关后端运行状况，容量和性能（ IOPS 吞吐量和 / 或延迟）的信息。

您可以查看Kubernetes应用程序正在使用的卷、这些卷存储在选定的存储后端。

步骤

1. 在左侧导航区域中，选择 * 后端 * 。
2. 选择存储后端。

编辑存储后端身份验证详细信息

Astra控制中心提供了两种对ONTAP 后端进行身份验证的模式。

- 基于凭据的身份验证：具有所需权限的ONTAP 用户的用户名和密码。您应使用预定义的安全登录角色(如admin)、以确保与ONTAP 版本的最大兼容性。
- 基于证书的身份验证：Astra控制中心还可以使用后端安装的证书与ONTAP 集群进行通信。您应使用客户端证书、密钥和可信CA证书(如果使用)(建议)。

您可以更新现有后端、以便从一种身份验证类型迁移到另一种身份验证方法。一次仅支持一种身份验证方法。

有关启用基于证书的身份验证的详细信息、请参见 "在ONTAP 存储后端启用身份验证"。

步骤

1. 从左侧导航栏中，选择 * 后端 * 。
2. 选择存储后端。
3. 在“凭据”字段中，选择*Edit*图标。
4. 在编辑页面中、选择以下选项之一。
   - 使用管理员凭据：输入ONTAP 集群管理IP地址和管理员凭据。凭据必须是集群范围的凭据。

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

   - 使用证书：上传证书 `.pem` file、证书密钥 `.key` 文件、以及证书颁发机构文件(可选)。
5. 选择 * 保存 * 。

管理已发现的存储后端

您可以选择管理未受管理但已发现的存储后端。管理存储后端时、Astra Control会指示用于身份验证的证书是否已过期。

步骤

1. 从左侧导航栏中，选择 * 后端 * 。
2. 选择*已发现*选项。
3. 选择存储后端。
4. 从“选项”菜单的“操作”列中，选择“管理”。
5. 进行更改。
6. 选择 * 保存 * 。

取消管理存储后端

您可以取消管理后端。

步骤

1. 从左侧导航栏中，选择 * 后端 * 。
2. 选择存储后端。
3. 从选项菜单的 * 操作 * 列中，选择 * 取消管理 * 。
4. 键入 "unmanage" 确认此操作。
5. 选择 * 是，取消管理存储后端 * 。

删除存储后端

您可以删除不再使用的存储后端。您可能需要执行此操作，以使您的配置简单且最新。

开始之前

- 确保存储后端未受管。
- 确保存储后端没有与集群关联的任何卷。

步骤

1. 从左侧导航栏中，选择 * 后端 * 。
2. 如果管理后端，请取消管理它。
   a. 选择 * 受管 * 。
   b. 选择存储后端。
   c. 从 * 操作 * 选项中，选择 * 取消管理 * 。
   d. 键入 "unmanage" 确认此操作。
   e. 选择 * 是，取消管理存储后端 * 。
3. 选择 * 已发现 * 。
   a. 选择存储后端。
   b. 从 * 操作 * 选项中，选择 * 删除 * 。
   c. 键入 "remove" 确认此操作。
   d. 选择 * 是，删除存储后端 * 。

了解更多信息

- "使用 Astra Control API"

监控正在运行的任务

您可以在Astra Control中查看有关过去24小时内已完成、失败或已取消的正在运行的任务和任务的详细信息。例如、您可以查看正在运行的备份、还原或克隆操作的状态、并查看完成百分比和估计剩余时间等详细信息。您可以查看已运行的已计划操作或手动启动的操作的状态。

查看正在运行或已完成的任务时、您可以展开任务详细信息以查看每个子任务的状态。对于正在进行的或已完成的任务、任务进度条为绿色、对于已取消的任务、任务进度条为蓝色、对于因错误而失败的任务、任务进度条为红色。

> 对于克隆操作、任务子任务由快照和快照还原操作组成。

要查看有关失败任务的详细信息、请参见 "监控帐户活动"。

步骤

1. 在任务运行期间、转到*应用程序*。
2. 从列表中选择应用程序的名称。
3. 在应用程序的详细信息中、选择*任务*选项卡。

   您可以查看当前或过去任务的详细信息、并按任务状态进行筛选。

> 任务将在*任务*列表中保留长达24小时。您可以使用配置此限制以及其他任务监控器设置 "Astra Control API"。

[技术预览]使用CRS管理Astra Control应用程序

使用Kubbernetes自定义资源(CR)管理Astra Control应用程序。可以使用以下选项：

- "使用Kubbernetes自定义资源定义应用程序"
- "使用自定义资源管理存储分段"

通过Prometheus或Fluentd连接监控基础架构

您可以配置多种可选设置来增强您的 Astra 控制中心体验。要监控并深入了解您的整个基础架构、请配置Prometheus或添加Fluentd连接。

如果运行Astra控制中心的网络需要使用代理连接到Internet (以便将支持包上传到NetApp 支持站点)、则应在Astra控制中心中配置代理服务器。

- 连接到Prometheus
- 连接到 Fluentd

添加用于连接到NetApp 支持站点 的代理服务器

如果运行Astra控制中心的网络需要使用代理连接到Internet (以便将支持包上传到NetApp 支持站点)、则应在Astra控制中心中配置代理服务器。

> Astra 控制中心不会验证您为代理服务器输入的详细信息。请确保输入正确的值。

步骤

1. 使用具有 * 管理员 / 所有者 * 权限的帐户登录到 Astra 控制中心。
2. 选择 * 帐户 * > * 连接 * 。
3. 从下拉列表中选择 * 连接 * 以添加代理服务器。

4. 输入代理服务器名称或 IP 地址以及代理端口号。
5. 如果代理服务器需要身份验证，请选中此复选框，然后输入用户名和密码。
6. 选择 * 连接 * 。

结果

如果您输入的代理信息已保存，则 * 帐户 * > * 连接 * 页面的 * HTTP 代理 * 部分将指示它已连接，并显示服务器名称。

编辑代理服务器设置

您可以编辑代理服务器设置。

步骤

1. 使用具有 * 管理员 / 所有者 * 权限的帐户登录到 Astra 控制中心。
2. 选择 * 帐户 * > * 连接 * 。
3. 从下拉列表中选择*Edit*以编辑连接。
4. 编辑服务器详细信息和身份验证信息。
5. 选择 * 保存 * 。

禁用代理服务器连接

您可以禁用代理服务器连接。在禁用之前、系统会警告您可能会中断其他连接。

步骤

1. 使用具有 * 管理员 / 所有者 * 权限的帐户登录到 Astra 控制中心。
2. 选择 * 帐户 * > * 连接 * 。
3. 从下拉列表中选择 * 断开连接 * 以禁用连接。
4. 在打开的对话框中，确认操作。

连接到Prometheus

您可以使用Prometheus监控Astra控制中心数据。您可以将Prometheus配置为从Kubernetes集群指标端点收集指标、也可以使用Prometheus可视化指标数据。

有关使用Prometheus的详细信息、请参见其文档、网址为 "Prometheus入门"。

您需要的内容

确保已在Astra控制中心集群或可与Astra控制中心集群通信的其他集群上下载并安装Prometheus软件包。

按照官方文档中的说明进行操作 "安装 Prometheus"。

Prometheus需要能够与Astra控制中心Kubernetes集群进行通信。如果Astra控制中心集群上未安装Prometheus、您需要确保这些模块能够与Astra控制中心集群上运行的指标服务进行通信。

配置 Prometheus

Astra控制中心会在Kubernetes集群中的TCP端口9090上公开指标服务。您需要配置 Prometheus 以从此服务收集指标。

步骤

1. 登录到Prometheus服务器。
2. 将集群条目添加到中 `prometheus.yml` 文件在中 `yml` 文件中、为集群添加一个类似于以下内容的条目 `scrape_configs section`：

```
job_name: '<Add your cluster name here. You can abbreviate. It just
needs to be a unique name>'
  metrics_path: /accounts/<replace with your account ID>/metrics
  authorization:
     credentials: <replace with your API token>
  tls_config:
     insecure_skip_verify: true
  static_configs:
    - targets: ['<replace with your astraAddress. If using FQDN, the
prometheus server has to be able to resolve it>']
```

> 如果您设置了 `tls_config insecure_skip_verify` to `true`、不需要TLS加密协议。

3. 重新启动Prometheus服务：

```
sudo systemctl restart prometheus
```

访问Prometheus

访问Prometheus URL。

步骤

1. 在浏览器中、输入端口为9090的Prometheus URL。
2. 选择*状态*>*目标*以验证您的连接。

在**Prometheus**中查看数据

您可以使用Prometheus查看Astra控制中心数据。

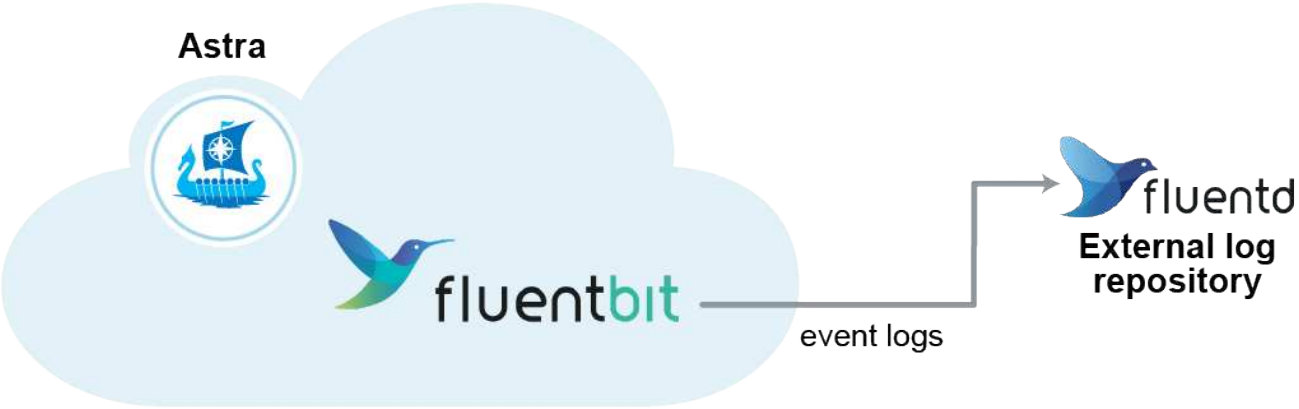
步骤

1. 在浏览器中、输入Prometheus URL。
2. 从Prometheus菜单中、选择*图形*。
3. 要使用指标资源管理器、请选择*执行*旁边的图标。
4. 选择 … `scrape_samples_scraped` 并选择*执行*。
5. 要查看随时间推移的样本擦除了、请选择*图形*。

> 如果收集了多个集群数据、则每个集群的指标将以不同的颜色显示。

连接到 Fluentd

您可以将日志(Kubennet事件)从Astra Control Center监控的系统发送到Fluentd端点。默认情况下， Fluentd 连接处于禁用状态。



> 只有受管集群中的事件日志才会转发到 Fluentd 。

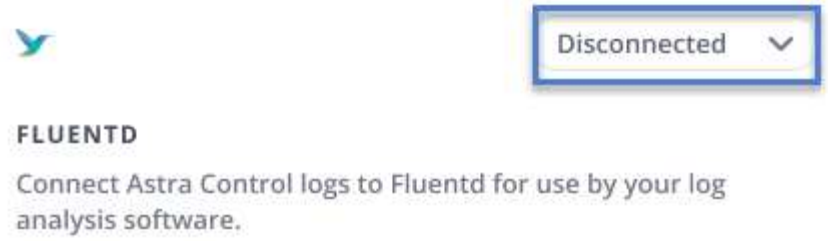
开始之前

- 具有 * 管理 / 所有者 * 权限的 Astra 控制中心帐户。
- 已在 Kubernetes 集群上安装并运行 Astra Control Center 。

> Astra 控制中心不会验证您为 Fluentd 服务器输入的详细信息。请确保输入正确的值。

步骤

1. 使用具有 * 管理员 / 所有者 * 权限的帐户登录到 Astra 控制中心。
2. 选择 * 帐户 * > * 连接 * 。
3. 从显示 * 已断开连接 * 的下拉列表中选择 * 连接 * 以添加连接。



4. 输入 Fluentd 服务器的主机 IP 地址，端口号和共享密钥。
5. 选择 * 连接 * 。

结果

如果您为 Fluentd 服务器输入的详细信息已保存，则 * 帐户 * > * 连接 * 页面的 * 通量 * 部分将指示它已连接。现在，您可以访问已连接的 Fluentd 服务器并查看事件日志。

如果连接因某种原因失败，则状态将显示 * 失败 * 。您可以在用户界面右上角的 * 通知 * 下找到失败的原因。

您还可以在 * 帐户 * > * 通知 * 下找到相同的信息。

> 如果您在收集日志时遇到问题，应登录到工作节点，并确保日志在 ` /var/log/containers/` 中可用。

编辑 **Fluentd** 连接

您可以编辑与 Astra Control Center 实例的 Fluentd 连接。

步骤

1. 使用具有 * 管理员 / 所有者 * 权限的帐户登录到 Astra 控制中心。
2. 选择 * 帐户 * > * 连接 * 。
3. 从下拉列表中选择*Edit*以编辑连接。
4. 更改 Fluentd 端点设置。
5. 选择 * 保存 * 。

禁用 **Fluentd** 连接

您可以禁用与 Astra Control Center 实例的 Fluentd 连接。

步骤

1. 使用具有 * 管理员 / 所有者 * 权限的帐户登录到 Astra 控制中心。
2. 选择 * 帐户 * > * 连接 * 。
3. 从下拉列表中选择 * 断开连接 * 以禁用连接。
4. 在打开的对话框中，确认操作。

取消管理应用程序和集群

从 Astra 控制中心删除不再需要管理的任何应用程序或集群。

取消管理应用程序

从 Astra 控制中心停止管理不再需要备份，快照或克隆的应用程序。

取消管理应用程序时：

- 所有现有备份和快照都将被删除。
- 应用程序和数据始终可用。

步骤

1. 从左侧导航栏中，选择 * 应用程序 * 。
2. 选择应用程序。
3. 从选项菜单的操作列中、选择*取消管理*。
4. 查看相关信息。
5. 键入 "unmanage" 进行确认。
6. 选择*是、取消管理应用程序*。

结果

Astra 控制中心停止管理应用程序。

取消管理集群

停止从Astra控制中心管理不再需要管理的集群。

> 在取消管理集群之前，您应取消管理与集群关联的应用程序。

取消管理集群时：

- 此操作将停止由 Astra 控制中心管理集群。它不会对集群的配置进行任何更改，也不会删除集群。
- Astra Control配置程序或Astra三端存储不会从集群中卸载。 "了解如何卸载 Astra Trident"。

步骤

1. 从左侧导航栏中，选择 * 集群 * 。
2. 选中不再要管理的集群对应的复选框。
3. 从选项菜单的 * 操作 * 列中，选择 * 取消管理 * 。
4. 确认要取消管理集群，然后选择 * 是，取消管理集群 * 。

结果

集群状态将更改为*正在删除*。之后、集群将从*集群*页面中删除、不再由Astra控制中心管理。

取消管理集群将删除为发送遥测数据而安装的所有资源。

升级 Astra 控制中心

要升级Astra Control Center、请下载安装映像并完成以下说明。您可以使用此操作步骤在互联网连接或通风环境中升级 Astra 控制中心。

这些说明介绍了Astra Control Center从第二个最新版本升级到此最新版本的过程。您不能直接从比当前版本落后两个或更多版本的版本升级。如果您安装的Astra Control Center版本比最新版本晚许多版本、您可能需要执行链式升级到更高版本、直到您安装的Astra Control Center仅比最新版本晚一个版本。有关已发布版本的完整列表、请参见 "发行说明"。

开始之前

在升级之前、请确保您的环境仍满足 "Asta Control Center部署的最低要求"。您的环境应具有以下内容：

- 已启用 **"Asta Control配置程序" A**作用 于运行**Astra**三端存储
  a. 确定您正在运行的Astra三项目标版本：

```
kubectl get tridentversion -n trident
```

> 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。如果您的Astra三端存储在版本24.02的四个版本窗口中、则可以直接升级到Astra Control置备程序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。

  b. 确认Asta Control配置程序已配置 "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、使其与要升级的Astra Control Center版本相同、以访问最新功能。

- 支持的**Kubernetes**分发

  确定您正在运行的Kubenetes版本：

```
kubectl get nodes -o wide
```

- 集群资源充足

  确定可用的集群资源：

```
kubectl describe node <node name>
```

- 默认存储类

  确定默认存储类：

```
kubectl get storageclass
```

- 运行状况良好且可用的**API**服务

  确保所有 API 服务均处于运行状况良好且可用：

```
kubectl get apiservices
```

- **(**仅限本地注册表**)**可用于推送和上传**Astra Control Center**映像的本地注册表
- **(**仅限**OpenShift)**运行状况良好且可用的集群操作符

  确保所有集群操作员均处于运行状况良好且可用。

```
kubectl get clusteroperators
```

您还应考虑以下事项：

> 如果计划，备份和快照未运行，请在维护窗口中执行升级。

- 访问**NetApp Astra**控件映像注册表：
  您可以选择从NetApp映像注册表中获取Astra控件的安装映像和增强功能、例如Astra控件配置程序。
  a. 记录您登录注册表所需的Astra Control帐户ID。

     您可以在Astra Control Service Web UI中查看您的帐户ID。选择页面右上角的图图标，选择*API access*并记下您的帐户ID。

  b. 在同一页面中，选择*Generate API t令牌*并将API令牌字符串复制到剪贴板，然后将其保存在编辑器中。
  c. 登录到Asta Control注册表：

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

- 伊斯提奥服务网状部署
  如果您在Astra Control Center安装期间安装了Isio服务网格、Astra Control Center的此升级将包括Isio服务网格。如果您还没有服务网格、则只能在期间安装一个 "初始部署" Astra控制中心。

关于此任务

Astra 控制中心升级过程将指导您完成以下高级步骤：

> 在开始升级之前、请从Astra控制中心用户界面中注销。

- 下载并提取Astra控制中心

- [如果使用本地注册表、请完成其他步骤]
- 安装更新后的 Astra 控制中心操作员
- 升级 Astra 控制中心
- [验证系统状态]

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

下载并提取**Astra**控制中心

从以下位置之一下载Astra Control Center映像：

- **Astra**控制服务映像注册表：如果您不对Astra控制中心映像使用本地注册表，或者如果您更喜欢使用此方法从NetApp 支持站点 下载捆绑包，请使用此选项。
- **Astra**：如果将本地注册表与NetApp 支持站点 控制中心映像一起使用，请使用此选项。

Astra Control图像注册表

1. 登录Asta Control Service。
2. 在信息板上，选择*Deploy a self-managed instance* of Asta Control*。
3. 按照说明登录到Astra Control映像注册表、提取Astra Control Center安装映像并提取该映像。

NetApp 支持站点

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

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

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

   此时将显示输出 `Verified OK` 验证成功后。

3. 从Astra Control Center捆绑包中提取映像：

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

如果使用本地注册表、请完成其他步骤

如果您计划将Astra控制中心捆绑包推送到本地注册表、则需要使用NetApp Astra kubect命令行插件。

删除**NetApp Astra kubectl**插件并重新安装

您需要使用最新版本的NetApp Astra kubect命令行插件将映像推送到本地Docker存储库。

1. 确定是否已安装此插件：

```
kubectl astra
```

2. 执行以下操作之一：
   - 如果已安装此插件、则此命令应返回kubecTn插件帮助、您可以删除现有版本的kubecT1-Asta：`delete /usr/local/bin/kubectl-astra`。
   - 如果此命令返回错误、则表示未安装此插件、您可以继续执行下一步以安装它。
3. 安装插件：
   a. 列出可用的NetApp Astra kubectl插件二进制文件、并记下操作系统和CPU架构所需的文件名称：

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

```
ls kubectl-astra/
```

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

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

将映像添加到注册表

1. 如果您计划将Astra Control Center捆绑包推送到本地注册表、请为容器引擎完成相应的步骤顺序：

Docker

a. 更改为tarball的根目录。您应看到 `acc.manifest.bundle.yaml` 文件和以下目录：

```
acc/
kubectl-astra/
acc.manifest.bundle.yaml
```

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

- 将<BUNDLE_FILE> 替换为Astra Control捆绑包文件的名称 (`acc.manifest.bundle.yaml` ）。
- 将<MY_FULL_REGISTRY_PATH> 替换为Docker存储库的URL；例如 "<a href="https://<docker-registry>"" class="bare">https://<docker-registry>"。
- 将<MY_REGISTRY_USER> 替换为用户名。
- 将<MY_REGISTRY_TOKEN> 替换为注册表的授权令牌。

```
kubectl astra packages push-images -m <BUNDLE_FILE> -r
<MY_FULL_REGISTRY_PATH> -u <MY_REGISTRY_USER> -p
<MY_REGISTRY_TOKEN>
```

Podman

a. 更改为tarball的根目录。您应看到此文件和目录：

```
acc/
kubectl-astra/
acc.manifest.bundle.yaml
```

b. 登录到注册表：

```
podman login <YOUR_REGISTRY>
```

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

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

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=24.02.0-69
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed
's/Loaded image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

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

```
export REGISTRY=<MY_FULL_REGISTRY_PATH>
export PACKAGENAME=acc
export PACKAGEVERSION=24.02.0-69
export DIRECTORYNAME=acc
for astraImageFile in $(ls ${DIRECTORYNAME}/images/*.tar) ; do
astraImage=$(podman load --input ${astraImageFile} | sed
's/Loaded image: //')
astraImageNoPath=$(echo ${astraImage} | sed 's:.*/::')
podman tag ${astraImageNoPath} ${REGISTRY}/netapp/astra/
${PACKAGENAME}/${PACKAGEVERSION}/${astraImageNoPath}
podman push ${REGISTRY}/netapp/astra/${PACKAGENAME}/
${PACKAGEVERSION}/${astraImageNoPath}
done
```

根据您的注册表配置、此脚本创建的映像路径应类似于以下内容：

```
https://downloads.example.io/docker-astra-control-
prod/netapp/astra/acc/24.02.0-69/image:version
```

2. 更改目录：

```
cd manifests
```

安装更新后的 Astra 控制中心操作员

1. (仅限本地注册表)如果使用的是本地注册表、请完成以下步骤：

   a. 打开Asta控制中心操作员部署YAML：

   ```
   vim astra_control_center_operator_deploy.yaml
   ```

   > 以下步骤将提供一个标注的YAML示例。

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

   ```
   imagePullSecrets: [{name: astra-registry-cred}]
   ```

   c. 更改 `ASTRA_IMAGE_REGISTRY` 。 `kube-rbac-proxy` 将映像推送到注册表路径中 上一步。

   d. 更改 `ASTRA_IMAGE_REGISTRY` 。 `acc-operator` 将映像推送到注册表路径中 上一步。

   e. 将以下值添加到 `env` 部分：

   ```
   - name: ACCOP_HELM_UPGRADETIMEOUT
     value: 300m
   ```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    control-plane: controller-manager
  name: acc-operator-controller-manager
  namespace: netapp-acc-operator
spec:
  replicas: 1
  selector:
    matchLabels:
      control-plane: controller-manager
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        control-plane: controller-manager
    spec:
      containers:
      - args:
```

```
        - --secure-listen-address=0.0.0.0:8443
        - --upstream=http://127.0.0.1:8080/
        - --logtostderr=true
        - --v=10
        image: ASTRA_IMAGE_REGISTRY/kube-rbac-proxy:v4.8.0
        name: kube-rbac-proxy
        ports:
        - containerPort: 8443
          name: https
      - args:
        - --health-probe-bind-address=:8081
        - --metrics-bind-address=127.0.0.1:8080
        - --leader-elect
        env:
        - name: ACCOP_LOG_LEVEL
          value: "2"
        - name: ACCOP_HELM_UPGRADETIMEOUT
          value: 300m
        image: ASTRA_IMAGE_REGISTRY/acc-operator:24.02.68
        imagePullPolicy: IfNotPresent
        livenessProbe:
          httpGet:
            path: /healthz
            port: 8081
          initialDelaySeconds: 15
          periodSeconds: 20
        name: manager
        readinessProbe:
          httpGet:
            path: /readyz
            port: 8081
          initialDelaySeconds: 5
          periodSeconds: 10
        resources:
          limits:
            cpu: 300m
            memory: 750Mi
          requests:
            cpu: 100m
            memory: 75Mi
        securityContext:
          allowPrivilegeEscalation: false
      imagePullSecrets: []
      securityContext:
        runAsUser: 65532
      terminationGracePeriodSeconds: 10
```

2. 安装更新后的 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 unchanged
customresourcedefinition.apiextensions.k8s.io/astracontrolcenters.as
tra.netapp.io configured
role.rbac.authorization.k8s.io/acc-operator-leader-election-role
unchanged
clusterrole.rbac.authorization.k8s.io/acc-operator-manager-role
configured
clusterrole.rbac.authorization.k8s.io/acc-operator-metrics-reader
unchanged
clusterrole.rbac.authorization.k8s.io/acc-operator-proxy-role
unchanged
rolebinding.rbac.authorization.k8s.io/acc-operator-leader-election-
rolebinding unchanged
clusterrolebinding.rbac.authorization.k8s.io/acc-operator-manager-
rolebinding configured
clusterrolebinding.rbac.authorization.k8s.io/acc-operator-proxy-
rolebinding unchanged
configmap/acc-operator-manager-config unchanged
service/acc-operator-controller-manager-metrics-service unchanged
deployment.apps/acc-operator-controller-manager configured
```

3. 验证Pod是否正在运行：

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

升级 Astra 控制中心

1. 编辑Astra Control Center自定义资源(CR)：

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

> 以下步骤将提供一个标注的YAML示例。

2. 更改Astra版本号 (`astraVersion` 在中 `spec`) `23.10.0` to `24.02.0`：

您不能直接从比当前版本落后两个或更多版本的版本升级。有关已发布版本的完整列表、请参见 "发行说明"。

```
spec:
  accountName: "Example"
  astraVersion: "[Version number]"
```

3. 更改图像注册表：
   - (仅限本地注册表)如果使用的是本地注册表、请验证映像注册表路径是否与中将映像推送到的注册表路径匹配 上一步。更新 `imageRegistry` 在中 `spec` 如果本地注册表自上次安装以来发生了更改。
   - (Astra Control图像注册表)使用Astra Control图像注册表 (`cr.astra.netapp.io`)您曾下载更新的Astra Control捆绑包。

```
imageRegistry:
  name: "[cr.astra.netapp.io or your_registry_path]"
```

4. 将以下内容添加到 `crds` 中的配置 `spec`：

```
crds:
  shouldUpgrade: true
```

5. 在中添加以下行 `additionalValues` 在中 `spec` 在Astra控制中心CR中：

```
additionalValues:
    nautilus:
      startupProbe:
        periodSeconds: 30
        failureThreshold: 600
    keycloak-operator:
      livenessProbe:
        initialDelaySeconds: 180
      readinessProbe:
        initialDelaySeconds: 180
```

6. 保存并退出文件编辑器。此时将应用所做的更改、并开始升级。
7. （可选）验证 Pod 是否终止并重新可用：

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

8. 等待Astra Control状态条件指示升级已完成且准备就绪 (`True`）：

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

响应：

```
NAME    UUID                                      VERSION     ADDRESS
READY
astra   9aa5fdae-4214-4cb7-9976-5d8b4c0ce27f      24.02.0-69
10.111.111.111  True
```

> 要在操作期间监控升级状态、请运行以下命令： `kubectl get AstraControlCenter -o yaml -n [netapp-acc or custom namespace]`

> 要检查Astra控制中心操作员日志、请运行以下命令：
> `kubectl logs deploy/acc-operator-controller-manager -n netapp-acc-operator -c manager -f`

验证系统状态

1. 登录到 Astra 控制中心。
2. 验证此版本是否已升级。请参见用户界面中的*支持*页面。
3. 验证所有受管集群和应用程序是否仍存在并受到保护。

使用OpenShift OperatorHub升级Astra Control Center

如果您使用Red Hat认证的操作员安装了Astra Control Center、则可以使用OperatorHub中更新的操作员升级Astra Control Center。使用此操作步骤从升级Astra控制中心 "Red Hat生态系统目录" 或使用 Red Hat OpenShift 容器平台。

开始之前

- 满足环境前提条件：在升级之前、请确保您的环境仍满足 "Asta Control Center部署的最低要求"。
- 确保已启用 **"Asta Control配置程序" A**作用 于运行**Astra**三端存储
  a. 确定您正在运行的Astra三项目标版本：

```
kubectl get tridentversion -n trident
```

> 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。如果您的Astra三端存储在版本24.02的四个版本窗口中、则可以直接升级到Astra Control置备程序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。

b. 确认Asta Control配置程序已配置 "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、使其与要升级的Astra Control Center版本相同、以访问最新功能。

- 确保集群操作员和**API**服务运行正常：
  - 在OpenShift集群中、确保所有集群操作员均处于运行状况良好的状态：

```
oc get clusteroperators
```

  - 在OpenShift集群中、确保所有API服务均处于运行状况良好的状态：

```
oc get apiservices
```

- * OpenShift权限*：您拥有执行所述升级步骤所需的所有权限和对Red Hat OpenShift容器平台的访问权限。
- **(**仅限**ONTAP SAN**驱动程序**)**启用多路径：如果使用的是ONTAP SAN驱动程序、请确保在所有Kubernetes集群上启用了多路径。

您还应考虑以下事项：

- 获取**NetApp Astra**控件映像注册表的访问权限：

  您可以选择从NetApp映像注册表中获取Astra控件的安装映像和增强功能、例如Astra控件配置程序。

  a. 记录您登录注册表所需的Astra Control帐户ID。

     您可以在Astra Control Service Web UI中查看您的帐户ID。选择页面右上角的图图标，选择*API access*并记下您的帐户ID。

  b. 在同一页面中，选择*Generate API t令牌*并将API令牌字符串复制到剪贴板，然后将其保存在编辑器中。

  c. 登录到Asta Control注册表：

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

步骤

- [访问操作员安装页面]
- [卸载现有操作员]
- [安装最新的操作员]
- 升级 Astra 控制中心

访问操作员安装页面

1. 完成OpenShift容器平台或生态系统目录对应的操作步骤：

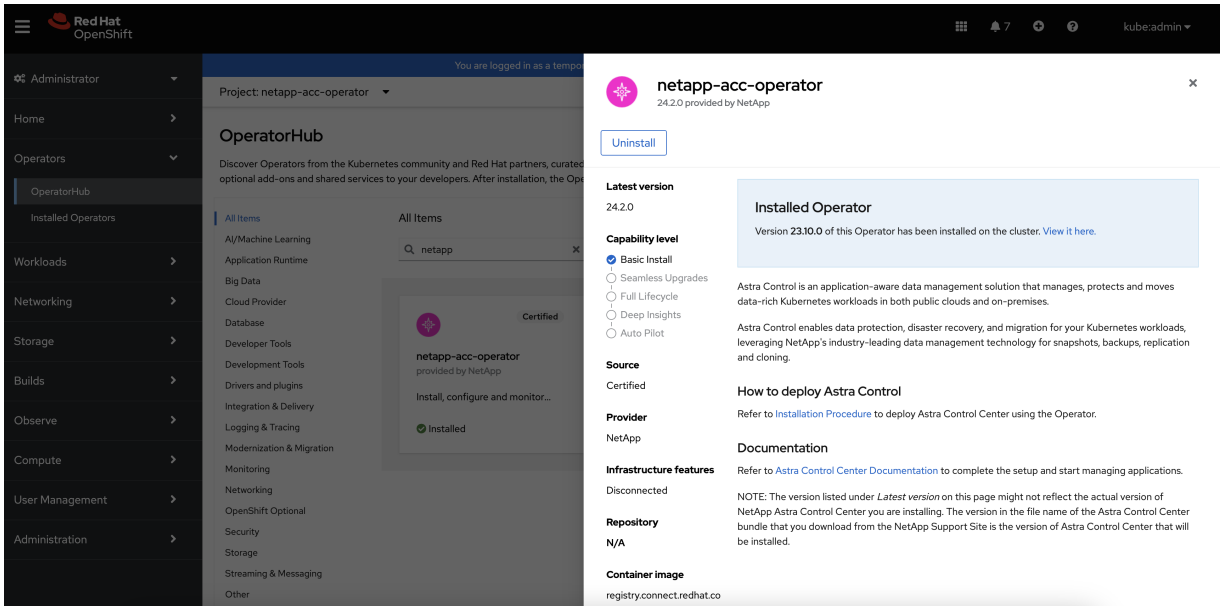
Red Hat OpenShift Web控制台

a. 登录到 OpenShift 容器平台 UI 。

b. 从侧面菜单中，选择 * 运算符 > OperatorHub * 。

> 使用此运算符只能升级到Astra Control Center的当前版本。

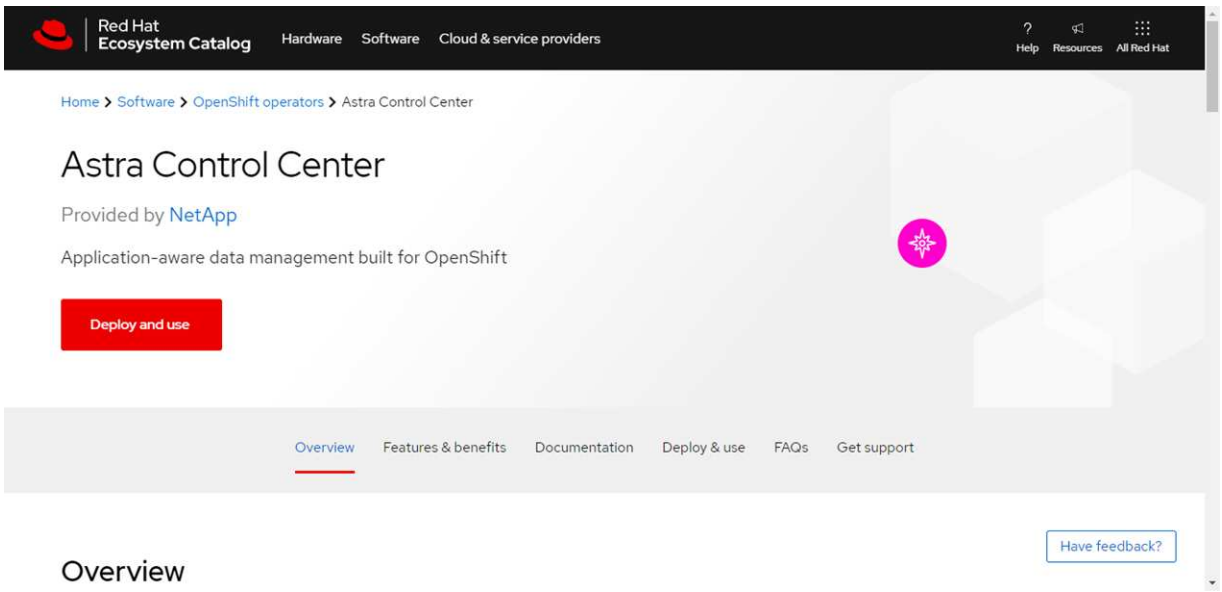
c. 搜索 `netapp-acc` 并选择NetApp Asta控制中心操作员。



Red Hat 生态系统目录

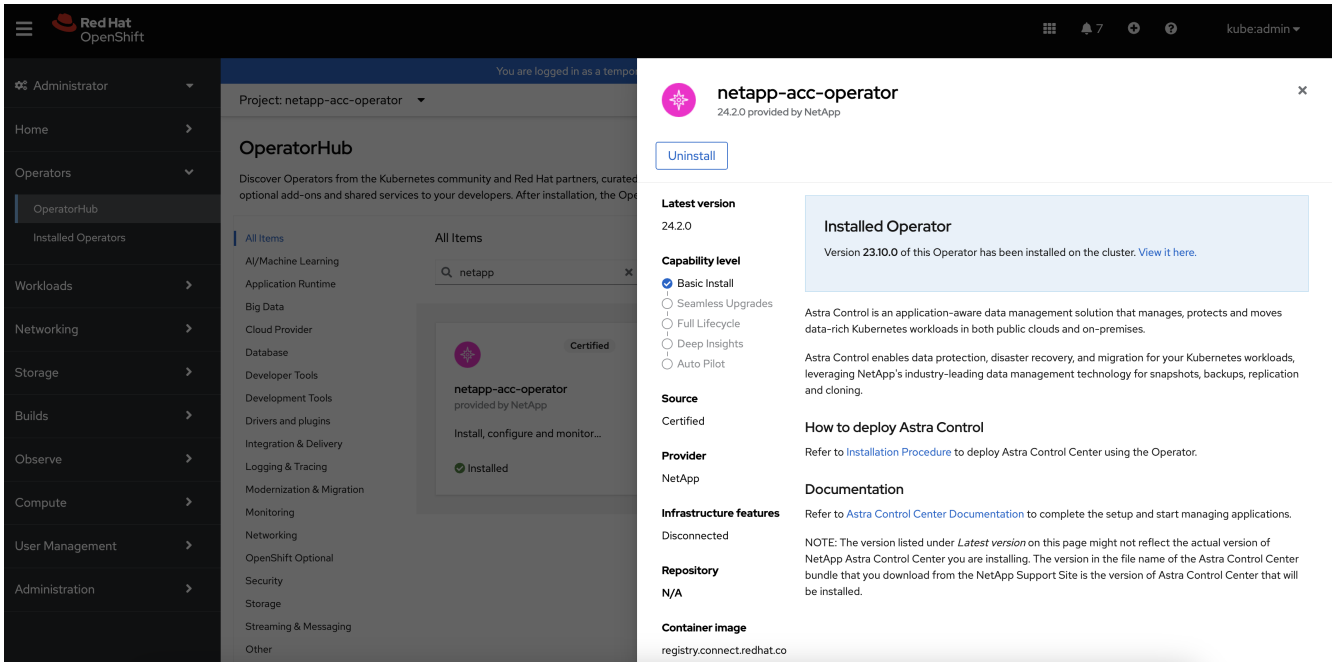
a. 选择 NetApp Astra 控制中心 "运算符"。

b. 选择*部署和使用*。



卸载现有操作员

1. 从*NetApp-ACC-OPERATOR *页面中，选择*Uninstall*以删除现有操作员。

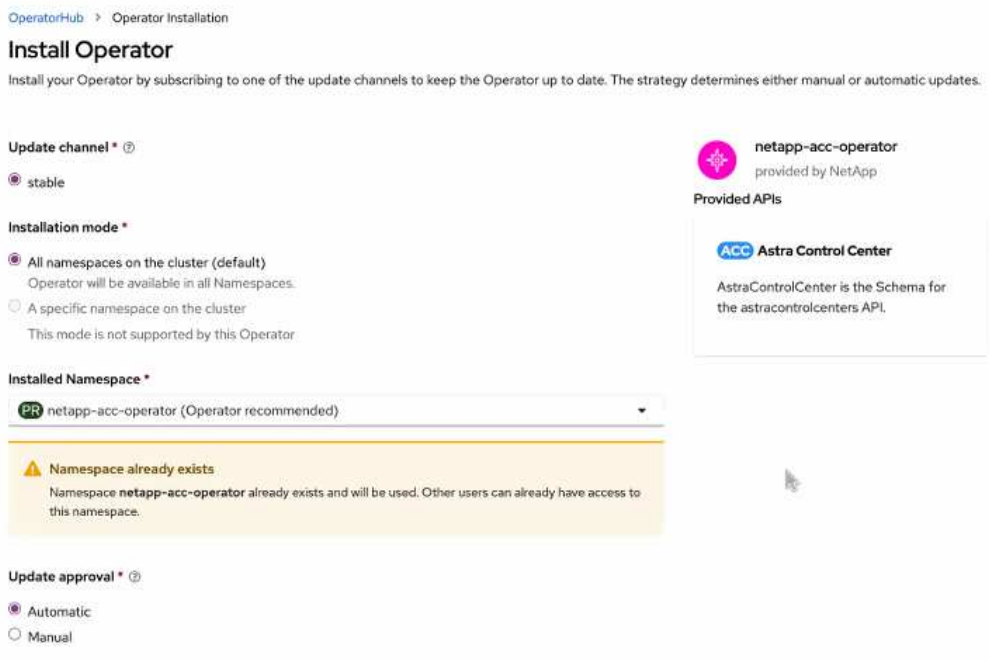


2. 确认操作。

> 此操作将删除NetApp-ACC-operator、但会保留原始关联的命名空间和资源、例如机密信息。

安装最新的操作员

1. 导航到 `netapp-acc` 再次显示操作员页面。
2. 完成*安装操作员*页并安装最新的操作员：



> 操作员将在所有集群命名空间中可用。

a. 选择操作员的 `netapp-acc-operator` 已删除操作员先前安装的命名空间(或自定义命名空间)。

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。

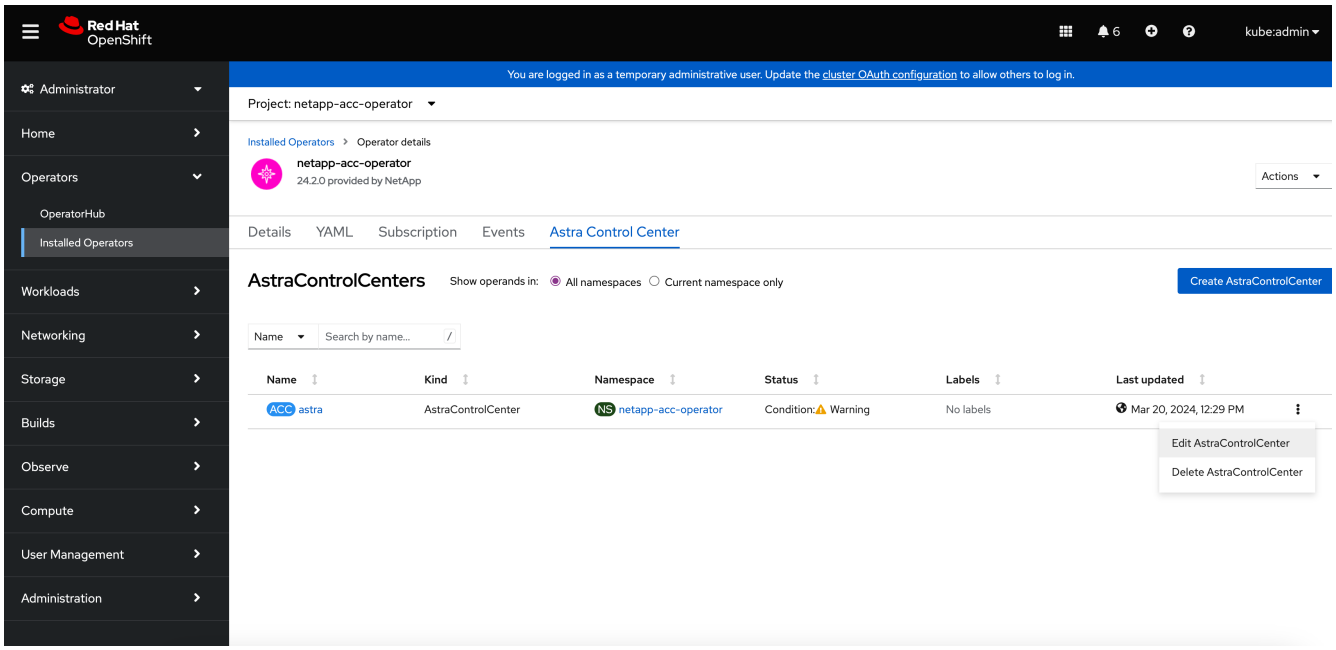
c. 选择 * 安装 * 。

> 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。

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

升级 **Astra** 控制中心

1. 从Astra Control Center operator选项卡中，选择保留先前安装的Astra Control Center，然后选择*编辑AstraControlCenter*。



2. 更新 `AstraControlCenter` YAML：
   a. 输入最新的Astra Control Center版本；例如24.02.0-69。
   b. 在中 `imageRegistry.name`，根据需要更新映像注册表路径：
      - 如果使用的是Asta Control注册表选项、请将路径更改为 `cr.astra.netapp.io`。
      - 如果您配置了本地注册表、请更改或保留上一步中推送图像的本地图像注册表路径。

      请止步 `http://` 或 `https://` 在地址字段中。

   c. 更新 `imageRegistry.secret` 根据需要。

      操作员卸载过程不会删除现有机密。只有在使用与现有机密不同的名称创建新机密时、才需要更新此字段。

   d. 将以下内容添加到 `crds` 配置：

```
crds:
  shouldUpgrade: true
```

3. 保存所做的更改。
4. 此UI将确认升级已成功。

卸载 Astra 控制中心

如果要从试用版升级到完整版本的产品，您可能需要删除 Astra Control Center 组件。要删除 Astra 控制中心和 Astra 控制中心操作员，请按顺序运行此操作步骤中所述的命令。

如果您在卸载时遇到任何问题，请参见 [对卸载问题进行故障排除]。

开始之前

1. "取消管理所有应用程序" 。
2. "取消管理所有集群"。

步骤

1. 删除 Astra 控制中心。以下命令示例基于默认安装。如果已进行自定义配置，请修改命令。

```
kubectl delete -f astra_control_center.yaml -n netapp-acc
```

结果

```
astracontrolcenter.astra.netapp.io "astra" deleted
```

2. 使用以下命令删除 `netapp-acc` (或自定义名称)命名空间：

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

结果示例：

```
namespace "netapp-acc" deleted
```

3. 使用以下命令删除 Astra 控制中心操作员系统组件：

```
kubectl delete -f astra_control_center_operator_deploy.yaml
```

结果

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

对卸载问题进行故障排除

使用以下解决方法解决卸载 Astra 控制中心时出现的任何问题。

卸载 **Astra** 控制中心无法清理受管集群上的监控操作员 **POD**

如果在卸载 Astra Control Center 之前未取消管理集群，则可以使用以下命令手动删除 netapp-monitoring 命名空间和命名空间中的 Pod：

步骤

1. 删除 附件监控 代理：

   ```
   kubectl delete agents acc-monitoring -n netapp-monitoring
   ```

   结果

   ```
   agent.monitoring.netapp.com "acc-monitoring" deleted
   ```

2. 删除命名空间：

   ```
   kubectl delete ns netapp-monitoring
   ```

   结果

```
namespace "netapp-monitoring" deleted
```

3. 确认已删除资源：

```
kubectl get pods -n netapp-monitoring
```

结果

```
No resources found in netapp-monitoring namespace.
```

4. 确认已删除监控代理：

```
kubectl get crd|grep agent
```

示例结果：

```
agents.monitoring.netapp.com                    2021-07-21T06:08:13Z
```

5. 删除自定义资源定义（CRD）信息：

```
kubectl delete crds agents.monitoring.netapp.com
```

结果

```
customresourcedefinition.apiextensions.k8s.io
"agents.monitoring.netapp.com" deleted
```

卸载 **Astra** 控制中心无法清理 **Traefik CRD**

您可以手动删除 Traefik CRD 。CRD 是全局资源，删除它们可能会影响集群上的其他应用程序。

步骤

1. 列出集群上安装的 Traefik CRD：

```
kubectl get crds |grep -E 'traefik'
```

响应

```
ingressroutes.traefik.containo.us             2021-06-23T23:29:11Z
ingressroutetcps.traefik.containo.us          2021-06-23T23:29:11Z
ingressrouteudps.traefik.containo.us          2021-06-23T23:29:12Z
middlewares.traefik.containo.us               2021-06-23T23:29:12Z
middlewaretcps.traefik.containo.us            2021-06-23T23:29:12Z
serverstransports.traefik.containo.us         2021-06-23T23:29:13Z
tlsoptions.traefik.containo.us                2021-06-23T23:29:13Z
tlsstores.traefik.containo.us                 2021-06-23T23:29:14Z
traefikservices.traefik.containo.us           2021-06-23T23:29:15Z
```

2. 删除 CRD：

```
kubectl delete crd ingressroutes.traefik.containo.us
ingressroutetcps.traefik.containo.us
ingressrouteudps.traefik.containo.us middlewares.traefik.containo.us
serverstransports.traefik.containo.us tlsoptions.traefik.containo.us
tlsstores.traefik.containo.us traefikservices.traefik.containo.us
middlewaretcps.traefik.containo.us
```

了解更多信息

- "卸载的已知问题"

使用Astra Control配置程序

配置存储后端加密

通过使用Astra Control配置程序、您可以对受管集群和存储后端之间的流量启用加密、从而提高数据访问安全性。

Astra Control配置程序支持对两种类型的存储后端进行Kerberos加密：

- 内部部署**Kubernet**—Astra配置程序支持通过从ONTAP OpenShift和上游Kubernetes集群到内部部署ONTAP卷的NFS3和NFSv4连接进行Kerberos加密。
- **NFSv**-控件配置程序支持通过从上游Azure NetApp Files集群到Azure NetApp Files卷的NFSv4.1连接进行Kerberos加密。

您可以创建、删除、调整大小、创建快照、克隆、 只读克隆、并导入使用NFS加密的卷。

为内部**ONTAP**卷配置传输中的**Kerberos**加密

您可以对受管集群与内部ONTAP存储后端之间的存储流量启用Kerberos加密。

> 仅支持使用对使用内部ONTAP存储后端的NFS流量进行Kerberos加密 `ontap-nas` 存储驱动程序。

开始之前

- 确保您已安装 "已启用Asta Control配置程序" 在受管集群上。
- 确保您可以访问 `tridentctl` 实用程序。
- 确保您对ONTAP存储后端具有管理员访问权限。
- 确保您知道要从ONTAP存储后端共享的一个或多个卷的名称。
- 确保已准备好ONTAP Storage VM以支持NFS卷的Kerberos加密。请参见 "在数据 LIF 上启用 Kerberos" 有关说明，请参见。
- 确保已正确配置使用Kerberos加密的任何NFSv4卷。请参阅的NetApp NFSv4域配置一节(第13页) "《NetApp NFSv4增强功能和最佳实践指南》"。

添加或修改**ONTAP**导出策略

您需要向现有ONTAP导出策略添加规则、或者创建新的导出策略、以便对ONTAP Storage VM根卷以及与上游Kubornetes集群共享的任何ONTAP卷支持Kerberos加密。您添加的导出策略规则或创建的新导出策略需要支持以下访问协议和访问权限：

访问协议

使用NFS、NFSv3和NFSv4访问协议配置导出策略。

访问详细信息

您可以根据卷的需求配置以下三种不同版本的Kerberos加密之一：

- **Kerberos 5**-(身份验证和加密)
- **Kerberos 5i**-(身份验证和加密与身份保护)
- **Kerberos 5p**-(身份验证和加密、具有身份和隐私保护功能)

使用适当的访问权限配置ONTAP导出策略规则。例如、如果集群要挂载混合使用Kerberos 5i和Kerberos 5p加密的NFS卷、请使用以下访问设置：

Type	只读访问	读/写访问	超级用户访问
"unix"	enabled	enabled	enabled
Kerberos 5i	enabled	enabled	enabled
Kerberos 5p	enabled	enabled	enabled

有关如何创建ONTAP导出策略和导出策略规则、请参见以下文档：

- "创建导出策略"
- "向导出策略添加规则"

创建存储后端

您可以创建包含Kerberos加密功能的A作用 力控制配置程序存储后端配置。

关于此任务

在创建用于配置Kerberos加密的存储后端配置文件时、您可以使用指定三个不同版本的Kerberos加密之一 `spec.nfsMountOptions` 参数：

- `spec.nfsMountOptions: sec=krb5` (身份验证和加密)
- `spec.nfsMountOptions: sec=krb5i` (身份验证和加密以及身份保护)
- `spec.nfsMountOptions: sec=krb5p` (身份验证和加密以及身份和隐私保护)

请仅指定一个Kerberos级别。如果在参数列表中指定多个Kerberos加密级别、则仅会使用第一个选项。

步骤

1. 在受管集群上、使用以下示例创建存储后端配置文件。将括号<>中的值替换为您环境中的信息：

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-ontap-nas-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-ontap-nas
spec:
  version: 1
  storageDriverName: "ontap-nas"
  managementLIF: <STORAGE_VM_MGMT_LIF_IP_ADDRESS>
  dataLIF: <PROTOCOL_LIF_FQDN_OR_IP_ADDRESS>
  svm: <STORAGE_VM_NAME>
  username: <STORAGE_VM_USERNAME_CREDENTIAL>
  password: <STORAGE_VM_PASSWORD_CREDENTIAL>
  nasType: nfs
  nfsMountOptions: ["sec=krb5i"] #can be krb5, krb5i, or krb5p
  qtreesPerFlexvol:
  credentials:
    name: backend-ontap-nas-secret
```

2. 使用您在上一步中创建的配置文件创建后端：

```
tridentctl create backend -f <backend-configuration-file>
```

如果后端创建失败，则后端配置出现问题。您可以运行以下命令来查看日志以确定发生原因：

```
tridentctl logs
```

确定并更正配置文件中的问题后，您可以再次运行 create 命令。

创建存储类。

您可以创建存储类来配置采用Kerberos加密的卷。

关于此任务

创建存储类对象时、您可以使用指定三个不同版本的Kerberos加密之一 `mountOptions` 参数：

- `mountOptions: sec=krb5` (身份验证和加密)
- `mountOptions: sec=krb5i` (身份验证和加密以及身份保护)
- `mountOptions: sec=krb5p` (身份验证和加密以及身份和隐私保护)

请仅指定一个Kerberos级别。如果在参数列表中指定多个Kerberos加密级别、则仅会使用第一个选项。如果您在存储后端配置中指定的加密级别与您在存储类对象中指定的加密级别不同、则存储类对象优先。

步骤

1. 使用以下示例创建StorageClass Kubenetes对象：

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-nas-sc
provisioner: csi.trident.netapp.io
mountOptions: ["sec=krb5i"] #can be krb5, krb5i, or krb5p
parameters:
  backendType: "ontap-nas"
  storagePools: "ontapnas_pool"
  trident.netapp.io/nasType: "nfs"
allowVolumeExpansion: True
```

2. 创建存储类：

```
kubectl create -f sample-input/storage-class-ontap-nas-sc.yaml
```

3. 确保已创建存储类：

```
kubectl get sc ontap-nas-sc
```

   您应看到类似于以下内容的输出：

```
NAME            PROVISIONER             AGE
ontap-nas-sc    csi.trident.netapp.io   15h
```

配置卷

创建存储后端和存储类后、您现在可以配置卷。请参阅以下说明 "配置卷"。

为Azure NetApp Files卷配置传输中的Kerberos加密

您可以对受管集群与单个Azure NetApp Files存储后端或Azure NetApp Files存储后端虚拟池之间的存储流量启用Kerberos加密。

开始之前

- 确保已在受管Red Hat OpenShift集群上启用Asta Control配置程序。请参见 "启用Asta Control配置程序" 有关说明，请参见。
- 确保您可以访问 `tridentctl` 实用程序。
- 请注意中的要求并按照中的说明、确保您已为Kerberos加密准备好Azure NetApp Files存储后端 "Azure NetApp Files 文档"。
- 确保已正确配置使用Kerberos加密的任何NFSv4卷。请参阅的NetApp NFSv4域配置一节(第13页) "《NetApp NFSv4增强功能和最佳实践指南》"。

创建存储后端

您可以创建包含Kerberos加密功能的Azure NetApp Files存储后端配置。

关于此任务

在创建配置Kerberos加密的存储后端配置文件时、您可以对其进行定义、使其应用于以下两个可能的级别之一：

- 使用的*存储后端级别* `spec.kerberos` 字段
- 使用的*虚拟池级别* `spec.storage.kerberos` 字段

在虚拟池级别定义配置时、系统会使用存储类中的标签来选择该池。

在任一级别、您都可以指定以下三种不同版本的Kerberos加密之一：

- `kerberos: sec=krb5` (身份验证和加密)
- `kerberos: sec=krb5i` (身份验证和加密以及身份保护)
- `kerberos: sec=krb5p` (身份验证和加密以及身份和隐私保护)

步骤

1. 在受管集群上、根据需要定义存储后端的位置(存储后端级别或虚拟池级别)、使用以下示例之一创建存储后端配置文件。将括号<>中的值替换为您环境中的信息：

存储后端级别示例

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-anf-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-anf-secret
```

虚拟池级别示例

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-anf-secret
type: Opaque
stringData:
  clientID: <CLIENT_ID>
  clientSecret: <CLIENT_SECRET>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: <SUBSCRIPTION_ID>
  tenantID: <TENANT_ID>
  location: <AZURE_REGION_LOCATION>
  serviceLevel: Standard
  networkFeatures: Standard
  capacityPools: <CAPACITY_POOL>
  resourceGroups: <RESOURCE_GROUP>
  netappAccounts: <NETAPP_ACCOUNT>
  virtualNetwork: <VIRTUAL_NETWORK>
  subnet: <SUBNET>
  nasType: nfs
  storage:
    - labels:
        type: encryption
      kerberos: sec=krb5i #can be krb5, krb5i, or krb5p
  credentials:
    name: backend-tbc-anf-secret
```

2. 使用您在上一步中创建的配置文件创建后端：

```
tridentctl create backend -f <backend-configuration-file>
```

如果后端创建失败，则后端配置出现问题。您可以运行以下命令来查看日志以确定发生原因：

```
tridentctl logs
```

确定并更正配置文件中的问题后，您可以再次运行 create 命令。

创建存储类。

您可以创建存储类来配置采用Kerberos加密的卷。

步骤

1. 使用以下示例创建StorageClass Kubenetes对象：

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-nfs
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "nfs"
  selector: "type=encryption"
```

2. 创建存储类：

```
kubectl create -f sample-input/storage-class-anf-sc-nfs.yaml
```

3. 确保已创建存储类：

```
kubectl get sc anf-sc-nfs
```

您应看到类似于以下内容的输出：

```
NAME          PROVISIONER              AGE
anf-sc-nfs     csi.trident.netapp.io    15h
```

配置卷

创建存储后端和存储类后、您现在可以配置卷。请参阅以下说明 "配置卷"。

使用快照恢复卷数据

Asta Control配置程序可使用从快照快速原位还原卷 `TridentActionSnapshotRestore` (TSR) CR。此CR用作要务Kubbernetes操作、在操作完成后不会持久保留。

Asta Control配置程序支持在上执行快照还原 `ontap-san`， `ontap-san-economy`， `ontap-nas`， `ontap-`

`nas-flexgroup`， `azure-netapp-files`， `gcp-cvs`，和 `solidfire-san` 驱动程序。

开始之前

您必须具有绑定的PVC和可用的卷快照。

- 验证PVC状态是否已绑定。

```
kubectl get pvc
```

- 确认卷快照已准备就绪、可以使用。

```
kubectl get vs
```

步骤

1. 创建TSR CR。此示例将为PVC创建CR `pvc1` 和卷快照 `pvc1-snapshot`。

```
cat tasr-pvc1-snapshot.yaml

apiVersion: trident.netapp.io/v1
kind: TridentActionSnapshotRestore
metadata:
  name: this-doesnt-matter
  namespace: trident
spec:
  pvcName: pvc1
  volumeSnapshotName: pvc1-snapshot
```

2. 应用CR以从快照还原。此示例将从Snapshot还原 `pvc1`。

```
kubectl create -f tasr-pvc1-snapshot.yaml

tridentactionsnapshotrestore.trident.netapp.io/this-doesnt-matter
created
```

结果

Asta Control配置程序从快照还原数据。您可以验证快照还原状态。

```
kubectl get tasr -o yaml

apiVersion: trident.netapp.io/v1
items:
- apiVersion: trident.netapp.io/v1
  kind: TridentActionSnapshotRestore
  metadata:
    creationTimestamp: "2023-04-14T00:20:33Z"
    generation: 3
    name: this-doesnt-matter
    namespace: trident
    resourceVersion: "3453847"
    uid: <uid>
  spec:
    pvcName: pvc1
    volumeSnapshotName: pvc1-snapshot
  status:
    startTime: "2023-04-14T00:20:34Z"
    completionTime: "2023-04-14T00:20:37Z"
    state: Succeeded
kind: List
metadata:
  resourceVersion: ""
```

- 在大多数情况下、如果发生故障、Asta Control配置程序不会自动重试此操作。您需要重新执行此操作。
- 没有管理员访问权限的Kubbernetes用户可能必须获得管理员授予的权限、才能在其应用程序命名空间中创建TSR CR。

使用SnapMirror复制卷

您可以使用Astra Control配置程序在一个集群上的源卷和对等集群上的目标卷之间创建镜像关系、以便为灾难恢复复制数据。您可以使用具有名称流的自定义资源定义(CRD)执行以下操作：

- 在卷之间创建镜像关系(PVC)
- 删除卷之间的镜像关系
- 中断镜像关系
- 在灾难情况下提升二级卷(故障转移)
- 在集群之间执行应用程序无中断过渡(在计划内故障转移或迁移期间)

复制前提条件

开始之前、请确保满足以下前提条件：

ONTAP 集群

- **Astra Control**配置程序：Astra Control配置程序23.10或更高版本或A "支持的Asta三项功能" 必须位于使用ONTAP作为后端的源和目标Kubbernetes集群上。
- 许可证：必须在源和目标ONTAP集群上启用使用数据保护包的ONTAP SnapMirror异步许可证。请参见 "ONTAP 中的SnapMirror许可概述" 有关详细信息 …

对等

- 集群和**SVM**：ONTAP存储后端必须建立对等状态。请参见 "集群和 SVM 对等概述" 有关详细信息 …

> 确保两个ONTAP集群之间的复制关系中使用的SVM名称是唯一的。

- **Astra Control**置备程序和**SVM**：对等远程SVM必须可供目标集群上的Astra Control置备程序使用。

支持的驱动程序

- ONTAP -NAS和ONTAP SAN驱动程序支持卷复制。

创建镜像**PVC**

按照以下步骤并使用CRD示例在主卷和二级卷之间创建镜像关系。

步骤

1. 在主Kubbernetes集群上执行以下步骤：
   a. 使用创建StorageClass对象 `trident.netapp.io/replication: true` 参数。

      示例

      ```
      apiVersion: storage.k8s.io/v1
      kind: StorageClass
      metadata:
        name: csi-nas
      provisioner: csi.trident.netapp.io
      parameters:
        backendType: "ontap-nas"
        fsType: "nfs"
        trident.netapp.io/replication: "true"
      ```

   b. 使用先前创建的StorageClass创建PVC。

示例

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: csi-nas
```

c. 使用本地信息创建镜像关系CR。

示例

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: promoted
  volumeMappings:
  - localPVCName: csi-nas
```

Astra Control配置程序会提取卷的内部信息以及卷的当前数据保护(DP)状态、然后填充镜像关系的状态字段。

d. 获取TridentMirorRelationship CR以获取PVC的内部名称和SVM。

```
kubectl get tmr csi-nas
```

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
  generation: 1
spec:
  state: promoted
  volumeMappings:
  - localPVCName: csi-nas
status:
  conditions:
  - state: promoted
    localVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
    localPVCName: csi-nas
    observedGeneration: 1
```

2. 在二级Kubbernetes集群上执行以下步骤：
   a. 使用trident.netapp.io/replication: true参数创建StorageClass。

      示例

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-nas
provisioner: csi.trident.netapp.io
parameters:
  trident.netapp.io/replication: true
```

   b. 使用目标和源信息创建镜像关系CR。

      示例

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: established
  volumeMappings:
  - localPVCName: csi-nas
    remoteVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
```

Asta控件配置程序将使用配置的关系策略名称(或ONTAP的默认策略名称)创建SnapMirror关系并对其进行初始化。

c. 使用先前创建的StorageClass创建一个PVC以用作二级(SnapMirror目标)。

示例

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
  annotations:
    trident.netapp.io/mirrorRelationship: csi-nas
spec:
  accessModes:
  - ReadWriteMany
resources:
  requests:
    storage: 1Gi
storageClassName: csi-nas
```

Astra Control配置程序将检查是否存在TridentMirorRelationship CRD、如果此关系不存在、则无法创建卷。如果存在此关系、Astra控件配置程序将确保将新FlexVol卷放置到与镜像关系中定义的远程SVM建立对等关系的SVM上。

卷复制状态

三级镜像关系(TCR)是一种CRD、表示PVC之间复制关系的一端。目标T关系 管理器具有一个状态、该状态会告诉Astra Control配置程序所需的状态是什么。目标T关系 管理器具有以下状态：

- 已建立：本地PVC是镜像关系的目标卷、这是一个新关系。
- 提升：本地PVC可读写并可挂载、当前未建立任何有效的镜像关系。
- 重新建立：本地PVC是镜像关系的目标卷、以前也位于该镜像关系中。
  - 如果目标卷曾经与源卷建立关系、因为它会覆盖目标卷的内容、则必须使用重新建立的状态。
  - 如果卷之前未与源建立关系、则重新建立的状态将失败。

在计划外故障转移期间提升辅助PVC

在二级Kubbernetes集群上执行以下步骤：

- 将TridentMirorRelationship的_spec.state_字 段更新为 `promoted`。

在计划内故障转移期间提升辅助PVC

在计划内故障转移(迁移)期间、执行以下步骤以提升二级PVC：

步骤

1. 在主Kubbernetes集群上、创建PVC的快照、并等待创建快照。
2. 在主Kubnetes集群上、创建SnapshotInfo CR以获取内部详细信息。

   示例

   ```
   kind: SnapshotInfo
   apiVersion: trident.netapp.io/v1
   metadata:
     name: csi-nas
   spec:
     snapshot-name: csi-nas-snapshot
   ```

3. 在二级Kubernetes集群上、将_TridentMirorRelationship_ CR的_spec.state_字 段更新为_promoted_和_spec.promotedSnapshotHandle_、以成为快照的内部名称。
4. 在二级Kubernetes集群上、确认Trident镜像 关系的状态(stats.state字段)为已提升。

在故障转移后还原镜像关系

在还原镜像关系之前、请选择要用作新主卷的那一端。

步骤

1. 在二级Kubernetes集群上、确保已更新TundentMirorRelationship上的_spic.netVolumeHandle_字段的值。
2. 在二级Kubernetes集群上、将Trident镜像 关系的_spec.mirector_字段更新到 `reestablished`。

其他操作

Asta Control配置程序支持在主卷和二级卷上执行以下操作：

将主**PVC**复制到新的二级**PVC**

确保您已有一个主PVC和一个次要PVC。

步骤

1. 从已建立的二级(目标)集群中删除PerbestentVolumeClaim和TridentMirorRelationship CRD。
2. 从主(源)集群中删除TridentMirorRelationship CRD。
3. 在主(源)集群上为要建立的新二级(目标) PVC创建新的TridentMirorRelationship CRD。

调整镜像、主**PVC**或二级**PVC**的大小

可以正常调整PVC的大小、如果数据量超过当前大小、ONTAP将自动扩展任何目标flevxvol。

从**PVC**中删除复制

要删除复制、请对当前二级卷执行以下操作之一：

- 删除次要PVC上的镜像关系。此操作将中断复制关系。
- 或者、将spec.state字段更新为_promoted_。

删除**PVC (**之前已镜像**)**

ASRA Control配置程序会检查是否存在复制的PVC、并在尝试删除卷之前释放复制关系。

删除**TTr**

删除镜像关系一端的T磁 还原会导致剩余的T磁 还原在Astra Control配置程序完成删除之前过渡到_promoted状态。如果选定要删除的TMirror已处于_Promote 状态、则不存在现有镜像关系、此时*TMirror*将被删除、*Astra Control*配置程序会将本地*PVC*提升为*_ReadWrite*。此删除操作将释放ONTAP中本地卷的SnapMirror元数据。如果此卷将来要在镜像关系中使用、则在创建新镜像关系时、它必须使用具有_re设立_卷复制状态的新TMirror。

在**ONTAP**联机时更新镜像关系

建立镜像关系后、可以随时更新这些关系。您可以使用 `state: promoted` 或 `state: reestablished` 用于更新关系的字段。
将目标卷提升为常规ReadWrite卷时、可以使用_promotedSnapshotHandle_指定要将当前卷还原到的特定快照。

在**ONTAP**脱机时更新镜像关系

您可以使用CRD执行SnapMirror更新、而Astra Control不直接连接到ONTAP集群。请参阅以下TridentAction镜像 更新的示例格式：

示例

```
apiVersion: trident.netapp.io/v1
kind: TridentActionMirrorUpdate
metadata:
  name: update-mirror-b
spec:
  snapshotHandle: "pvc-1234/snapshot-1234"
  tridentMirrorRelationshipName: mirror-b
```

`status.state` 反映TridentAction镜像 更新CRD的状态。它可以从_suced_、_in Progress _或_failed中获取值。

使用Astra Control REST API实现自动化

使用 Astra Control REST API 实现自动化

Astra Control 具有一个 REST API ，可用于使用编程语言或 Curl 等实用程序直接访问 Astra Control 功能。您还可以使用 Ansible 和其他自动化技术管理 Astra Control 部署。

要设置和管理Kubernetes应用程序、您可以使用Astra Control Center UI或Astra Control API。

要了解更多信息，请转到 "Astra 自动化文档"。

知识和支持

故障排除

了解如何解决您可能遇到的一些常见问题。

"NetApp Astra Control知识库"

了解更多信息

- "如何将文件上传到 NetApp （需要登录）"
- "如何手动将文件上传到 NetApp （需要登录）"

获取帮助

NetApp 以多种方式为 Astra Control 提供支持。全天候提供广泛的免费自助支持选项、例如知识库(KB)文章和不和渠道。您的 Astra Control 帐户包括通过 Web 服务单提供的远程技术支持。

> 如果您拥有 Astra 控制中心的评估许可证，则可以获得技术支持。但是，无法通过 NetApp 支持站点（ NSS ）创建案例。您可以通过反馈选项联系支持部门、也可以使用不和渠道自助服务。

您必须先执行此操作 "激活对您的 NetApp 序列号的支持" 以便使用这些非自助服务支持选项。聊天和 Web 服务单以及案例管理需要使用 NetApp 支持站点 (NSS) SSO 帐户。

自助支持选项

您可以从 Astra 控制中心用户界面访问支持选项，方法是从主菜单中选择 * 支持 * 选项卡。

这些选项全天候免费提供：

- "使用知识库(需要登录)"：搜索与 Astra Control 相关的文章，常见问题解答或中断修复信息。
- 请参阅产品文档：这是您当前正在查看的文档站点。
- "通过不和获得帮助"：转到"Pub类别"中的Astra、与同行和专家建立联系。
- * 创建支持案例 * ：生成支持包以提供给 NetApp 支持部门进行故障排除。
- * 提供有关 Astra Control* 的反馈：发送电子邮件至 astra.feedback@netapp.com ，告知我们您的想法，想法或顾虑。

启用每日计划的支持包上传至 **NetApp** 支持

在安装Astra Control Center期间(如果指定) `enrolled: true` 适用于 `autoSupport` 在Astra控制中心自定义资源(CR)文件中 (`astra_control_center.yaml`)、则会自动将每日支持包上传到 "NetApp 支持站点"。

生成要提供给 NetApp 支持的支持包

通过 Astra 控制中心，管理员用户可以生成捆绑包，其中包含对 NetApp 支持有用的信息，包括日志， Astra 部署的所有组件的事件，指标以及有关所管理集群和应用程序的拓扑信息。如果您已连接到 Internet ，则可以直接从 Astra 控制中心 UI 将支持包上传到 NetApp 支持站点（ NSS ）。

> Astra 控制中心生成该捆绑包所需的时间取决于您的 Astra 控制中心安装的大小以及请求的支持包的参数。您在请求支持包时指定的持续时间决定了生成支持包所需的时间（例如，较短的时间段会加快创建支持包的速度）。

开始之前

确定将捆绑包上传到 NSS 是否需要代理连接。如果需要代理连接，请验证是否已将 Astra 控制中心配置为使用代理服务器。

1. 选择 * 帐户 * > * 连接 * 。
2. 检查 * 连接设置 * 中的代理设置。

步骤

1. 使用 Astra 控制中心用户界面的 * 支持 * 页面上列出的许可证序列号在 NSS 门户上创建案例。
2. 要使用 Astra 控制中心 UI 生成支持包，请执行以下步骤：
   a. 在 * 支持 * 页面上的支持包磁贴中，选择 * 生成 * 。
   b. 在 * 生成支持包 * 窗口中，选择时间范围。

      您可以选择快速或自定义时间范围。

      > 您可以选择自定义日期范围，也可以指定日期范围内的自定义时间段。

   c. 选择后，选择 * 确认 * 。
   d. 选中 * 生成捆绑包时将其上传到 NetApp 支持站点 * 复选框。
   e. 选择 * 生成捆绑包 * 。

      支持包准备就绪后，警报区域中的 * 帐户 * > * 通知 * 页面， * 活动 * 页面以及通知列表（可通过选择 UI 右上角的图标来访问）中将显示一条通知。

   如果生成失败，则生成捆绑包页面上会显示一个图标。选择图标以查看消息。

> 用户界面右上角的通知图标提供了有关与支持包相关的事件的信息，例如，成功创建支持包的时间，创建支持包失败的时间，无法上传支持包的时间，无法下载支持包的时间等。

如果您安装了带气的安装

如果您安装了带风的安装，请在生成支持包后执行以下步骤。当该捆绑包可供下载时，在 * 支持 * 页面的 * 支持捆绑包 * 部分中的 * 生成 * 旁边会显示下载图标。

步骤

1. 选择下载图标以在本地下载此捆绑包。

2. 手动将捆绑包上传到 NSS 。

   您可以使用以下方法之一执行此操作：

   - 使用 … "NetApp 身份验证文件上传（需要登录）"。
   - 将捆绑包直接附加到 NSS 上的案例。
   - 使用 NetApp Active IQ 。

了解更多信息

- "如何将文件上传到 NetApp （需要登录）"
- "如何手动将文件上传到 NetApp （需要登录）"

早期版本的 Astra 控制中心文档

可提供先前版本的文档。

- "Astra Control Center 23.10文档"
- "Astra Control Center 23.07文档"
- "Astra Control Center 23.04文档"
- "Astra控制中心22.11文档"
- "Astra Control Center 22.08文档"
- "Astra Control Center 22.04 文档"
- "Astra Control Center 21.12 文档"
- "Astra Control Center 21.08 文档"

常见问题解答

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**Asta Control URL**?

Astra 控制中心使用本地身份验证以及每个环境专用的 URL 。

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

许可

我正在使用评估版许可证。如何更改为完整许可证?

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

- 步骤 *
  1. 从左侧导航栏中，选择 * 帐户 * > * 许可证 * 。
  2. 在许可证概述中、选择许可证信息右侧的选项菜单。
  3. 选择*替换*。
  4. 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。我仍然可以管理应用程序吗?

可以、您可以使用评估版许可证(包括默认安装的嵌入式评估版许可证)测试管理应用程序功能。评估版许可证与完整版许可证在功能上没有区别；评估版许可证的使用寿命更短。请参见 "许可" 有关详细信息 …

注册 Kubernetes 集群

在添加到**Astra Control**后、我需要向**Kubbernetes**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**Asta Control**中删除**Kubbernetes**集群后、我的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**Asta Control**配置程序**(**或**Asta Tricent)**时、它是否会自动从集群中卸载？

从Astra Control Center取消管理集群时、Astra Control配置程序或Astra三项功能不会自动从集群中卸载。要卸载Asta Control配置程序及其组件或Asta Trandent、您需要 "请按照以下步骤卸载包含Astra Control配置程序服务的Astra Trdent实例"。

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**Asta 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、但无法管理使用非NetApp存储的应用程序。

我应该自行管理**Astra Control**吗？

Astra Control Center默认情况下不会显示为您可以管理的应用程序、但您可以管理 "备份和还原" 使用另一个Asta Control Center实例的Asta Control Center实例。

运行状况不正常的**Pod**是否会影响应用程序管理？

不会、Pod的运行状况不会影响应用程序管理。

数据管理操作

我的应用程序使用多个**PV.Astra Control**是否会为这些**PVs**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。

我是否可以通过其他接口或对象存储直接管理**Astra Control**拍摄的快照？

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

Asta Control配置程序

Astra Control配置程序的存储配置功能与**Astra Trident**中的存储配置功能有何不同？

Asta Control配置程序作为Asta Control的一部分、支持一组超群的存储配置功能、这些功能在开源Asta三元数据中不可用。这些功能是对开放源码的三元数据可用的所有功能的补充。

Astra Control配置程序是否正在取代**Astra Trent**？

Asta Control配置程序已取代Asta Trandent、成为Asta Control架构中的存储配置程序和流程编排程序。Asta Control用户应执行此操作 "启用Asta Control配置程序" 使用A作用 力控制。此版本仍支持Asta三项功能、但未来版本不支持此功能。Asta三元数据将保持开源状态、并使用NetApp的新CSI和其他功能进行发布、维护、支持和更新。但是、只有包含A作用 力三项CSI功能以及扩展存储管理功能的A作用 力控制配置程序才能用于即将推出的A作用 力控制版本。

我是否必须为**Asta**三端存储付费?

否Asta三端技术将继续采用开源方式、并可免费下载。现在、使用Astra Control配置程序功能需要Astra Control许可证。

是否可以在不安装和使用所有**Astra Control**的情况下使用**Astra Control**中的存储管理和配置功能?

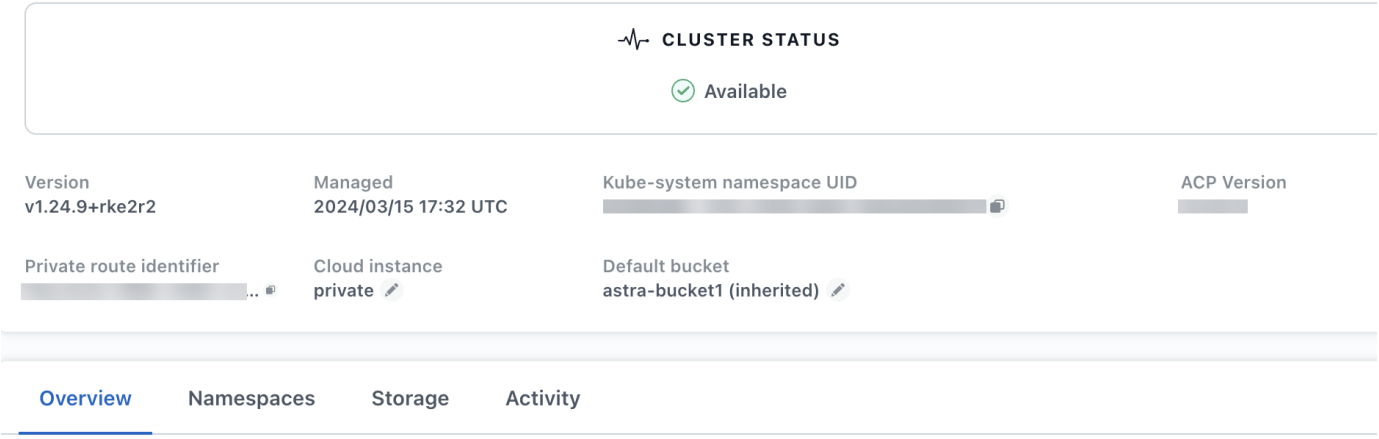
可以。即使您不想使用Astra Control数据管理功能的完整功能集、也可以升级到Astra Control配置程序并使用其功能。

如何从现有**Asta Trident**用户过渡到**Asta Control**以使用高级存储管理和配置功能?

如果您是现有的Asta Trident用户(包括公有云中的Asta Trident用户)、则需要先获取Asta Control许可证。完成此操作后、您可以下载Astra Control配置程序捆绑包、升级Astra三端、和 "启用Astra Control配置程序功能"。

如何知道**Astra Control**配置程序是否已取代了集群上的**Astra Trident**?

安装Asta Control配置程序后、Asta Control UI中的主机集群将显示 `ACP version` 而不是 `Trident version` 字段和当前安装的版本号。



如果您无权访问此UI、则可以使用以下方法确认安装成功:

Asta三端操作员

验证 `trident-acp` 容器正在运行 `acpVersion` 为 `23.10.0` 或更高版本(最低版本为23.10)、状态为 `Installed`：

```
kubectl get torc -o yaml
```

响应：

```
status:
  acpVersion: 24.10.0
  currentInstallationParams:
    ...
    acpImage: <my_custom_registry>/trident-acp:24.10.0
    enableACP: "true"
    ...
  ...
  status: Installed
```

Tridentctl

确认已启用Asta Control配置程序：

```
./tridentctl -n trident version
```

响应：

```
+----------------+----------------+-------------+ | SERVER VERSION |
CLIENT VERSION | ACP VERSION | +----------------+----------------
+-------------+ | 24.10.0 | 24.10.0 | 24.10.0. | +----------------
+----------------+-------------+
```

法律声明

法律声明提供对版权声明、商标、专利等的访问。

版权

"https://www.netapp.com/company/legal/copyright/"

商标

NetApp 、 NetApp 徽标和 NetApp 商标页面上列出的标记是 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。

"https://www.netapp.com/company/legal/trademarks/"

专利

有关 NetApp 拥有的专利的最新列表，请访问：

https://www.netapp.com/pdf.html?item=/media/11887-patentspage.pdf

隐私政策

"https://www.netapp.com/company/legal/privacy-policy/"

开放源代码

通知文件提供有关 NetApp 软件中使用的第三方版权和许可证的信息。

- "Astra 控制中心通知"

Astra Control API 许可证

https://docs.netapp.com/us-en/astra-automation/media/astra-api-license.pdf