Astra Control Center 22.04 文档

Astra Control Center

NetApp
November 21, 2023

目录

Astra Control Center 22.04 文档

发行说明

我们很高兴地宣布发布了 Astra 控制中心 22.04.0 版。

- "此版本的 Astra 控制中心包含哪些内容"
- "已知问题"
- "Astra 数据存储和此 Astra 控制中心版本的已知问题"
- "已知限制"

在 Twitter @NetAppDoc 上关注我们。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的 Astra 控制中心 22.04.0 版。

2022 年 4 月 26 日（22.04.0）

新增功能和支持

- "从Astra控制中心部署Astra Data Store"
- "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"
- "Astra 数据存储和此 Astra 控制中心版本的已知问题"
- "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 Data Store 预览后端管理和监控"
- "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"
- "有关 Astra Data Store 预览版和此 Astra 控制中心版本的已知问题"
- "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"
- "Astra Data Store 文档"
- "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：

应用程序

- 还原应用程序会导致 PV 大小大于原始 PV
- 使用特定版本的 PostgreSQL 时应用程序克隆失败
- 使用服务帐户级别 OCP 安全上下文限制（SCC）时应用程序克隆失败
- [使用设置的存储类部署应用程序后，应用程序克隆将失败]

集群

- 如果默认的 kubeconfig 文件包含多个上下文，则使用 Astra 控制中心管理集群将失败

其他问题

- 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 ）
- 使用 Snapshot 控制器 4.2.0 版时，快照可能会失败

还原应用程序会导致 PV 大小大于原始 PV

如果在创建备份后调整永久性卷的大小，然后从该备份还原，则此永久性卷的大小将与 PV 的新大小匹配，而不是使用备份的大小。

使用特定版本的 PostgreSQL 时应用程序克隆失败

使用 BitNami PostgreSQL 11.5.0 图表时，同一集群中的应用程序克隆始终会失败。要成功克隆，请使用图表的早期或更高版本。

使用服务帐户级别 OCP 安全上下文限制（ SCC ）时应用程序克隆失败

如果在 OpenShift 容器平台集群的命名空间中的服务帐户级别配置了原始安全上下文约束，则应用程序克隆可能会失败。如果应用程序克隆失败，它将显示在 Astra 控制中心的受管应用程序区域中，状态为 `removed` 。请参见 "知识库文章" 有关详细信息 …

使用设置的存储类部署应用程序后，应用程序克隆将失败

在使用显式设置的存储类（例如， `helm install …-set global.storageClass=netapp-cvs-perf-至 至至` ）部署应用程序后，后续克隆应用程序的尝试要求目标集群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。此情况下没有恢复步骤。

如果默认的 kubeconfig 文件包含多个上下文，则使用 Astra 控制中心管理集群将失败

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。

使用 Snapshot 控制器 4.2.0 版时，快照可能会失败

如果将 Kubernetes Snapshot-controller （也称为外部快照程序） 4.2.0 与 Kubernetes 1.20 或 1.21 结合使用，则快照最终可能会开始失败。要防止出现这种情况，请使用其他 "支持的版本" 使用 Kubernetes 版本 1.20 或 1.21 的外部快照程序，例如 4.2.1 版。

1. 运行 POST 调用，将更新后的 kubeconfig 文件添加到 `凭据` 端点，并从响应正文中检索分配的 `id` 。
2. 使用适当的集群 ID 从 `集群` 端点运行 PUT 调用，并将 `credentialId` 设置为上一步中的 `id` 值。

完成这些步骤后，将更新与集群关联的凭据，集群应重新连接并将其状态更新为 `Available` 。

了解更多信息

- "Astra Data Store prreview 和此 Astra Control Center 版本的已知问题"
- "已知限制"

Astra 数据存储和此 Astra 控制中心版本的已知问题

已知问题可确定可能妨碍您成功使用此版本产品的问题。

"请参见以下已知问题" 这可能会影响当前版本的Astra控制中心对Astra数据存储的管理。

了解更多信息

- "已知问题"
- "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]

应用程序管理限制

- [无法停止正在进行的应用程序备份]
- [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

一般限制

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

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

受命名空间名称 /ID 或命名空间标签约束的任何 m成员 用户均可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。通过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 m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。

无法停止正在进行的应用程序备份

无法停止正在运行的备份。如果需要删除备份，请等待备份完成，然后按照中的说明进行操作 "删除备份"。要删除失败的备份，请使用 "Astra Control API"。

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"

请注意， 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。

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"。

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。

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。

了解更多信息

- "已知问题"
- "Astra 数据存储和此 Astra 控制中心版本的已知问题"

概念

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
- 轻松地将应用程序从生产环境克隆到暂存环境
- 直观显示应用程序运行状况和保护状态
- 使用用户界面或 API 实施备份和迁移工作流

Astra Control 会持续监控您的计算状态变化，因此它可以识别您在此过程中添加的任何新应用程序。

部署模式

Astra Control 有两种部署模式：

- * Astra Control Service*： NetApp 管理的一项服务，可在 Google Kubernetes Engine （ GKEE ）和 Azure Kubernetes Service （ AKS ）中为 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	作为您下载，安装和管理的软件
它托管在何处？	基于 NetApp 选择的公有云	在您提供的 Kubernetes 集群上
如何更新？	由 NetApp 管理	您可以管理任何更新
应用程序数据管理功能是什么？	两个平台上的功能相同，但存储后端或外部服务除外	两个平台上的功能相同，但存储后端或外部服务除外

	Astra 控制服务	Astra 控制中心
什么是存储后端支持？	NetApp 云服务产品	• NetApp ONTAP AFF 和 FAS 系统 • 作为存储后端的 Astra 数据存储 • Cloud Volumes ONTAP 存储后端

支持的应用程序

NetApp 已对某些应用程序进行了验证，以确保快照和备份的安全性和一致性。

- "了解Astra Control中经过验证的应用程序与标准应用程序之间的区别"。

无论与 Astra Control 结合使用哪种类型的应用程序，您都应始终自行测试备份和还原工作流，以确保满足灾难恢复要求。

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 磁盘存储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进行安装 "Astra Trident" 以创建一个或多个存储类。
  - 如果您使用 Azure NetApp Files 或 NetApp Cloud Volumes Service for Google Cloud 作为存储后端，则 Astra 控制服务将使用 Astra Trident 为应用程序配置永久性卷。
- 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 Service 会持续监控您的计算状态变化，因此它可以识别您在此过程中添加的任何新应用程序。

Astra Control 的免费计划支持您管理帐户中多达 10 个应用程序。如果您要管理 10 个以上的应用程序，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 控制中心支持具有以下功能的 OpenShift Kubernetes 集群：

- ONTAP 9.5 及更高版本的 Trident 存储后端
- Astra 数据存储存储后端

在云互联环境中， Astra 控制中心使用 Cloud Insights 提供高级监控和遥测功能。如果没有 Cloud Insights 连接，则 Astra 控制中心可提供有限的（ 7 天的指标）监控和遥测功能，并通过开放式指标端点导出到 Kubernetes 原生监控工具（例如 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 天评估许可证试用 Astra Control Center 。评估版可通过电子邮件和社区（ Slack 通道）选项来支持。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"。

Astra 控制中心可执行以下操作：

- 发现有关受管 Kubernetes 集群的详细信息。
- 在您选择管理的集群上发现您的 Astra Trident 或 Astra 数据存储配置，并可用于监控存储后端。
- 发现这些集群上的应用程序，并使您能够管理和保护这些应用程序。

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 控制中心来发现和管理它们。然后，使用 Astra 控制中心创建快照，备份和克隆。

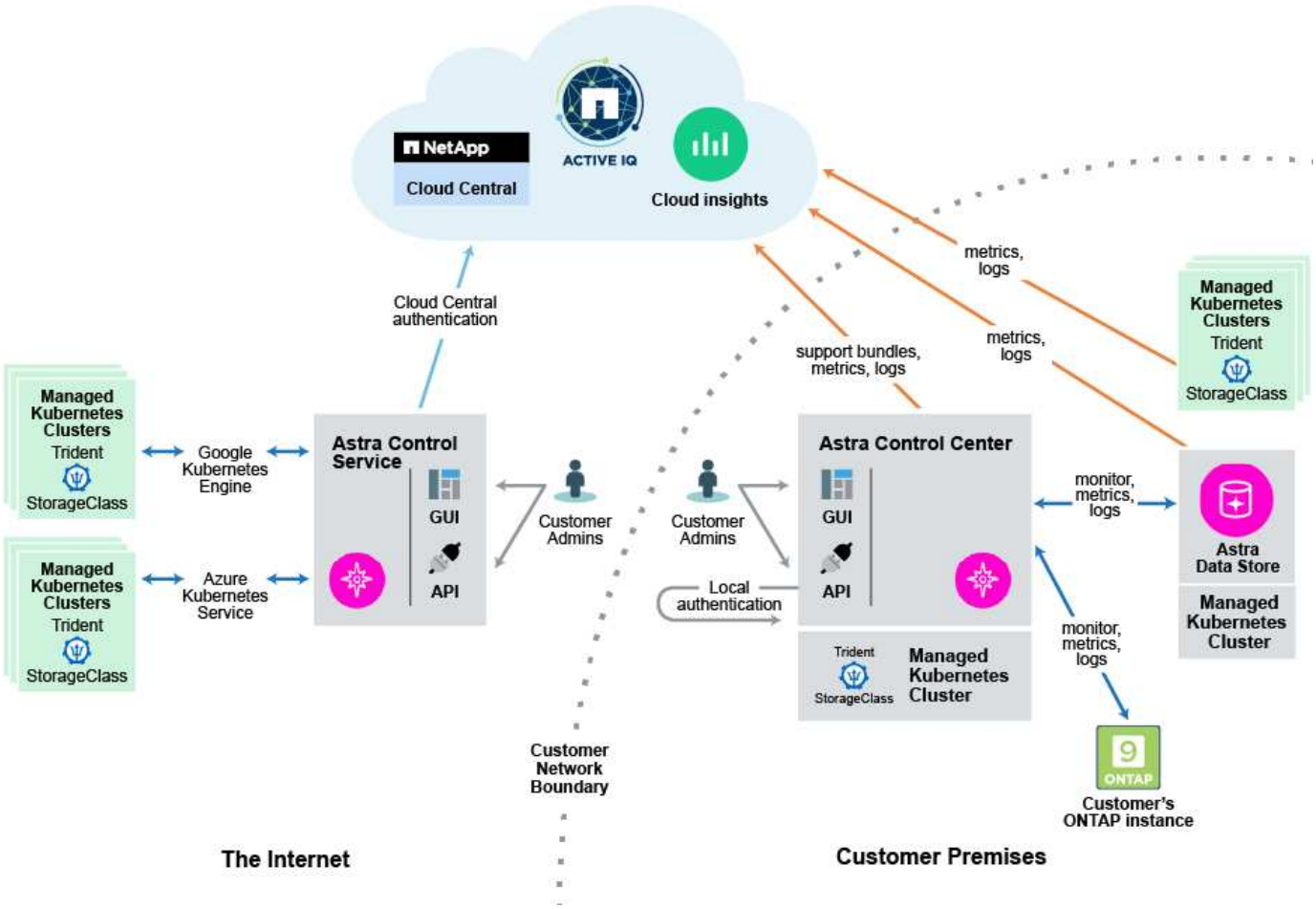
有关详细信息 ...

- "Astra Control Service 文档"
- "Astra 控制中心文档"
- "Astra Data Store 文档"

- "Astra Trident 文档"
- "使用 Astra Control API"
- "Cloud Insights 文档"
- "ONTAP 文档"

架构和组件

下面简要介绍了 Astra Control 环境的各个组件。



Astra Control 组件

- * Kubernetes 集群 *： Kubernetes 是一个可移植，可扩展的开源平台，用于管理容器化工作负载和服务，便于进行声明性配置和自动化。Astra 为 Kubernetes 集群中托管的应用程序提供管理服务。
- * Astra Trident *：作为 NetApp 维护的完全受支持的开源存储配置程序和编排程序， Trident 使您能够为 Docker 和 Kubernetes 管理的容器化应用程序创建存储卷。在使用 Astra 控制中心部署时， Trident 会包括一个已配置的 ONTAP 存储后端，并支持将 Astra 数据存储作为存储后端。
- * 存储后端 *：
  - Astra Control Service 使用 "适用于 Google Cloud 的 NetApp Cloud Volumes Service" 作为 GKE- 集群和的存储后端 "Azure NetApp Files" 作为 AKS 集群的存储后端。
  - Astra 控制服务还支持将 Azure 受管磁盘和 Google 永久性磁盘用作后端存储选项。

- Astra 控制中心使用以下存储后端：
  - Astra Data Store 存储后端
  - ONTAP AFF 和 FAS 存储后端。作为存储软件和硬件平台， ONTAP 可提供核心存储服务，支持多个存储访问协议以及快照和镜像等存储管理功能。
  - Cloud Volumes ONTAP 存储后端

- * Cloud Insights *： Cloud Insights 是一款 NetApp 云基础架构监控工具，可用于监控由 控制中心管理的 Kubernetes 集群的性能和利用率。Cloud Insights 将存储使用量与工作负载相关联。在 Astra 控制中心中启用 Cloud Insights 连接后，遥测信息将显示在 Astra 控制中心 UI 页面中。

Astra Control 接口

您可以使用不同的界面完成任务：

- * Web 用户界面（ UI ） *： Astra 控制服务和 Astra 控制中心使用同一个基于 Web 的 UI ，您可以在其中管理，迁移和保护应用程序。此外，还可以使用 UI 管理用户帐户和配置设置。
- * API *： Astra 控制服务和 Astra 控制中心使用相同的 Astra 控制 API 。使用 API ，您可以执行与使用 UI 相同的任务。

您还可以通过 Astra 控制中心管理，迁移和保护 VM 环境中运行的 Kubernetes 集群。

有关详细信息 ...

- "Astra Control Service 文档"
- "Astra 控制中心文档"
- "Astra Trident 文档"
- "使用 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

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。

backup 存储在外部对象存储中，与本地快照相比，创建速度可能较慢。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的快照和备份数量。使用保护策略自动执行备份和快照是确保每个应用程序都根据组织需求受到保护的最佳方式。

> *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。

克隆

clone 是应用程序，其配置及其永久性存储的精确副本。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。

许可

要启用完整的应用程序数据管理功能、需要安装Astra Control Center许可证。如果在没有许可证的情况下部署 Astra 控制中心，则 Web UI 中会显示一个横幅，警告系统功能受限。

以下操作需要有效的许可证：

- 管理新应用程序
- 创建快照或备份
- 配置保护策略以计划快照或备份
- 从快照或备份还原
- 从快照或当前状态克隆

> 您可以在没有许可证的情况下添加集群，添加存储分段以及管理 Astra Data Store 存储后端。但是，要使用 Astra Data Store 作为存储后端来管理应用程序，您需要有效的 Astra Control Center 许可证。

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。您还可以将 Astra Data Store 存储后端添加到 Astra 控制中心，而不会影响许可证使用。这样，您就可以从未经许可的 Astra 控制中心系统管理 Astra 数据存储后端。

开始管理集群上的应用程序时，计算 Astra 控制中心许可证占用情况时会考虑集群的 CPU 单元。

了解更多信息

- "更新现有许可证"

经验证的应用程序与标准应用程序

您可以为 Astra Control 引入两种类型的应用程序：经验证的应用程序和标准应用程序。了解这两个类别之间的差异以及对项目和战略的潜在影响。

将这两类产品视为 " 受支持 " 和 " 不受支持 " 很容易被认为。 但如您所见，在 Astra Control 中没有 " 不受支持 " 的应用程序。您可以将任何应用程序添加到 Astra Control 中，但与标准应用程序相比，经过验证的应用程序在其 Astra Control 工作流中构建的基础架构更多。

经验证的应用程序

经验证的 Astra Control 应用程序包括以下内容：

- MySQL 8.0.25
- MariaDB 10.5.9
- PostgreSQL 11.12
- Jenkins 2.277.4 LTS 和 2.289.1 LTS

经过验证的应用程序列表表示 Astra Control 可识别的应用程序。Astra Control 团队已对这些应用程序进行了分析并确认，这些应用程序需要经过全面测试才能恢复。Astra Control 执行自定义工作流，以帮助确保快照和备份的应用程序级别一致性。

如果某个应用程序已通过验证，则 Astra Control 团队已确定并实施了在创建快照之前暂停该应用程序的步骤，以便获取应用程序一致的快照。例如，当 Astra Control 对 PostgreSQL 数据库进行备份时，它会首先暂停数据库。备份完成后， Astra Control 会将数据库还原到正常运行状态。

无论您将哪种类型的应用程序与 Astra Control 结合使用，始终自行测试备份和还原工作流，以确保您能够满足灾难恢复要求。

标准应用程序

包括自定义程序在内的其他应用程序被视为标准应用程序。您可以通过 Astra Control 添加和管理标准应用程序。您还可以为标准应用程序创建崩溃状态一致的基本快照和备份。但是，这些功能尚未经过全面测试，无法将应用程序还原到其原始状态。

Astra Control 本身不是一个标准应用程序，而是一个 " 系统应用程序 " 。 默认情况下，用于管理的 Astra Control 本身不会显示。您不应尝试管理 Astra Control 本身。

存储类和永久性卷大小

Astra 控制中心支持使用 ONTAP 或 Astra 数据存储作为存储后端。

概述

Astra 控制中心支持以下功能：

- * 由 Astra Data Store 存储提供支持的 Trident 存储类 *： 如果您手动安装了一个或多个 Astra Data Store 集群，则 Astra 控制中心可以导入这些集群并检索其拓扑（节点，磁盘）以及各种状态。

  Astra 控制中心显示 Astra Data Store 配置中的底层 Kubernetes 集群， Kubernetes 集群所属的云，由 Astra Data Store 配置的任何永久性卷，相应内部卷的名称，使用永久性卷的应用程序以及包含此应用程序的集群。

- * 由 ONTAP 存储提供支持的 Trident 存储类 *：如果您使用的是 ONTAP 后端，则 Astra 控制中心可以导入 ONTAP 后端以报告各种监控信息。

> 应在 Astra 控制中心之外预先配置 Trident 存储类。

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。

如果使用与 Astra Data Store 存储后端集成的 Astra 控制中心，则在安装后，不会定义任何存储类。您需要创建 Trident 默认存储类并将其应用于存储后端。请参见 "Astra Data Store 入门" 创建默认的 Astra Data Store 存储类。

有关详细信息 ...

- "Astra Trident 文档"

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

"管理角色"

入门

Astra 控制中心要求

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

操作环境要求

Astra 控制中心需要以下类型的操作环境之一：

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

确保您选择托管 Astra 控制中心的操作环境满足环境官方文档中概述的基本资源要求。除了环境的资源要求之外， Astra 控制中心还需要以下资源：

组件	要求
存储后端容量	至少500 GB可用
工作节点	总共至少 3 个辅助节点，每个节点有 4 个 CPU 核和 12 GB RAM
FQDN 地址	Astra 控制中心的 FQDN 地址
Astra Trident	• 已安装并配置 Astra Trident 21.04 或更高版本 • 如果使用Astra数据存储作为存储后端、则已安装并配置Astra Trident 21.10.1或更高版本

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

- * 映像注册表 *： 您必须具有可将 Astra 控制中心构建映像推送到的现有私有 Docker 映像注册表。您需要提供要将映像上传到的映像注册表的 URL 。
- * 天文学 Trident / ONTAP 配置 *： 天文学控制中心要求创建一个存储类并将其设置为默认存储类。Astra 控制中心支持由 Astra Trident 提供的以下 ONTAP 驱动程序：
  - ontap-NAS
  - ontap-san
  - ontap-san-economy.

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

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

VMware Tanzu Kubernetes Grid 集群要求

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

- 在任何要由 Astra Control 管理的应用程序集群上禁用 TKG 或 TKGi 默认存储类强制实施。为此，您可以编辑命名空间集群上的 `TanuKubernetes Cluster` 资源。
- 您必须创建一个允许 Astra 控制中心在集群中创建 Pod 的安全策略。您可以使用以下命令执行此操作：

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

- 在 TKG 或 TKGi 环境中部署 Astra 控制中心时，请注意 Astra Trident 的特定要求。有关详细信息，请参见 "Astra Trident 文档"。

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

支持的存储后端

Astra 控制中心支持以下存储后端。

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

应用程序集群要求

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

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

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

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

应用程序管理要求

Astra Control 具有以下应用程序管理要求：

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

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

支持的应用程序安装方法

Astra Control 支持以下应用程序安装方法：

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

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

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

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

访问 Internet

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

许可证

要实现全部功能， Astra 控制中心需要获得 Astra 控制中心许可证。从 NetApp 获取评估版许可证或完整许可证。如果没有许可证，您将无法：

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

如果您要尝试使用 Astra 控制中心，可以 "使用 90 天评估许可证"。

要了解有关许可证工作原理的详细信息，请参见 "许可"。

内部 Kubernetes 集群的传入

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

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

有关详细信息，请参见 "设置传入以进行负载平衡"。

网络要求

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

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

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

支持的 **Web** 浏览器

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

下一步行动

查看 "快速入门" 概述。

Astra 控制中心快速入门

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

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

1. 查看 **Kubernetes** 集群要求

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

- 集群必须运行 Kubernetes 。

"了解有关 Astra 控制中心要求的更多信息"。

2 下载并安装**Astra**控制中心

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

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

"了解有关安装 Astra 控制中心的更多信息"。

3 完成一些初始设置任务

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

"了解有关初始设置过程的更多信息"。

4 使用 **Astra** 控制中心

设置完 Astra 控制中心后，您接下来可能会执行以下操作：

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

5 从此快速入门继续

"安装 Astra 控制中心"。

了解更多信息

- "使用 Astra Control API"

安装概述

选择并完成以下 Astra 控制中心安装过程之一：

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

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

使用标准流程安装 Astra 控制中心

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

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

您需要的内容

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

  OpenShift 示例：

  ```
  oc get clusteroperators
  ```

- 确保所有 API 服务均处于运行状况良好且可用：

  OpenShift 示例：

  ```
  oc get apiservices
  ```

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

关于此任务

Astra 控制中心安装过程将执行以下操作：

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

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

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

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

步骤

要安装 Astra 控制中心，请执行以下步骤：

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

下载并解包**Astra Control Center**软件包

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

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

4. 提取映像：

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

安装**NetApp Astra kubectl**插件

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

您需要的内容

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

步骤

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

```
ls kubectl-astra/
```

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

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

将映像添加到本地注册表

1. 更改为Astra目录：

```
cd acc
```

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

> 有关自动加载映像的信息，请参见下面的示例脚本。

a. 登录到注册表：

Docker：

```
docker login [your_registry_path]
```

播客：

```
podman login [your_registry_path]
```

b. 使用适当的脚本加载映像，标记映像，并将这些映像推送到本地注册表：

Docker：

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

播客：

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

为具有身份验证要求的注册表设置命名空间和密钥

1. 如果您使用的注册表需要身份验证，则需要执行以下操作：

   a. 创建 `NetApp-Acc-operator` 命名空间：

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

   响应：

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

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

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

响应示例：

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

c. 创建 `NetApp-Accc` （或自定义命名）命名空间。

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

响应示例：

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

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

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

响应

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

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

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

安装 **Astra** 控制中心操作员

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

```
vim astra_control_center_operator_deploy.yaml
```

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

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

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

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

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

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

2. 安装 Astra 控制中心操作员：

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

响应示例：

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

配置 **Astra** 控制中心

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

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

```
vim astra_control_center_min.yaml
```

> 首次部署 Astra 控制中心后，无法更改 CR 配置的属性。

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

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

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

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

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

i. 完成以下步骤之一：

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

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

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

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

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

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

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

完成 **Astra** 控制中心和操作员安装

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

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

响应示例：

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

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

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

响应示例：

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

验证系统状态

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

1. 验证是否已成功安装所有系统组件。

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

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

响应示例：

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

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

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

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

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

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

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

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

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

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

YAML 详细信息示例

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

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

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

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

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

设置传入以进行负载平衡

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

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

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

根据您使用的入口控制器类型，步骤会有所不同：

- nginx 入口控制器
- OpenShift 入口控制器

您需要的内容

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

nginx 入口控制器的步骤

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

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

b. 对于 `v1` 新架构，请遵循以下示例：

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

OpenShift 入口控制器的步骤

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

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

登录到 **Astra** 控制中心 **UI**

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

步骤

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

> 您可以在登录后创建自定义证书。

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

> 如果您输入的密码三次不正确，管理员帐户将锁定 15 分钟。

4. 选择 * 登录 * 。
5. 根据提示更改密码。

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

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

对安装进行故障排除

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

步骤

1. 要检查 Astra 控制中心操作员日志，请输入以下内容：

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

下一步行动

执行以完成部署 "设置任务"。

使用 OpenShift OperatorHub 安装 Astra 控制中心

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

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

您需要的内容

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

```
oc get clusteroperators
```

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

```
oc get apiservices
```

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

步骤

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

下载并解包**Astra Control Center**软件包

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

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

4. 提取映像：

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

安装**NetApp Astra kubectl**插件

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

您需要的内容

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

步骤

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

```
ls kubectl-astra/
```

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

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

将映像添加到本地注册表

1. 更改为Astra目录：

```
cd acc
```

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

> 有关自动加载映像的信息，请参见下面的示例脚本。

a. 登录到注册表：

Docker：

```
docker login [your_registry_path]
```

播客：

```
podman login [your_registry_path]
```

b. 使用适当的脚本加载映像，标记映像，并将这些映像推送到本地注册表：

Docker：

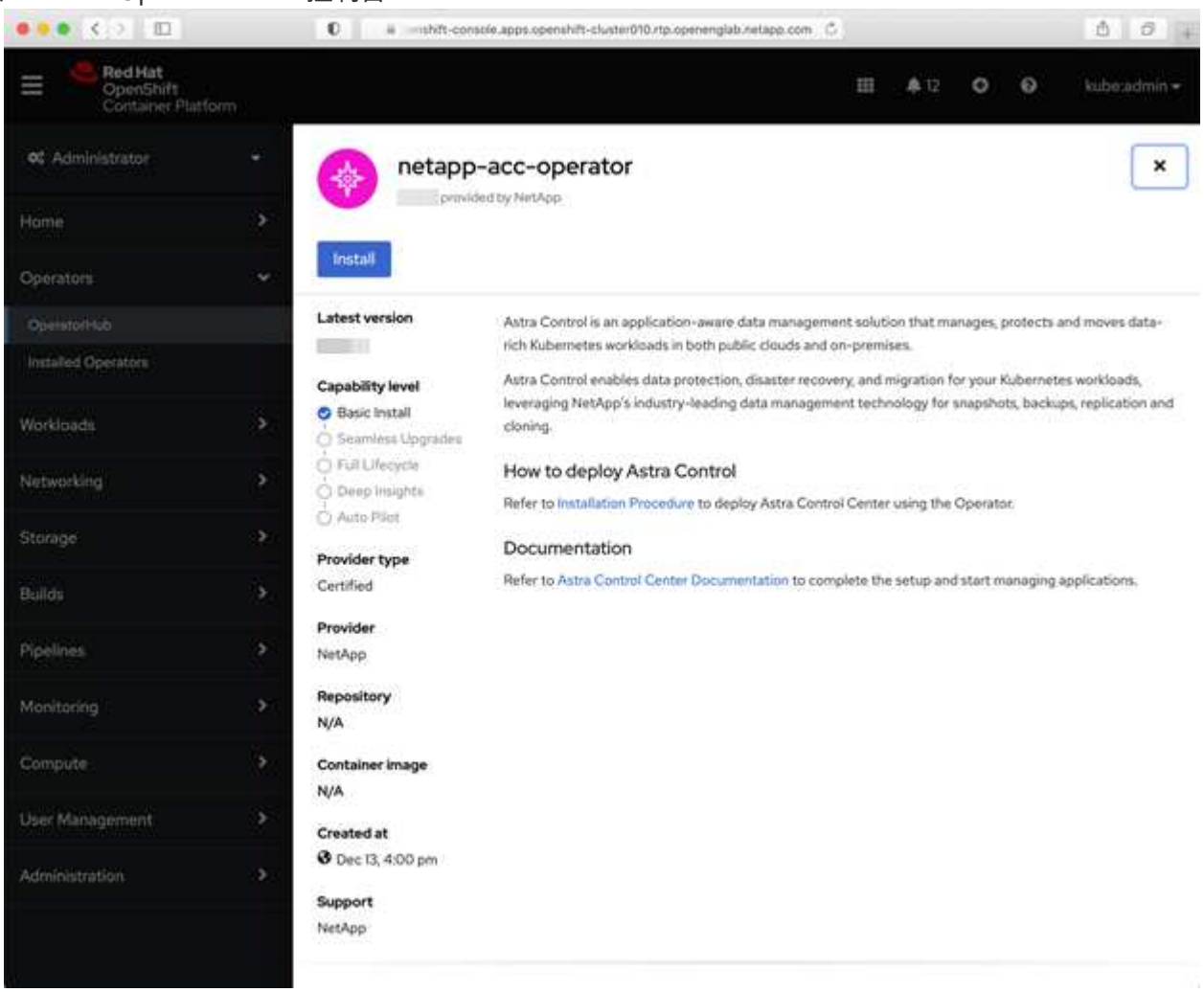
```
export REGISTRY=[Docker_registry_path]
for astraImageFile in $(ls images/*.tar) ; do
  # Load to local cache. And store the name of the loaded image
trimming the 'Loaded images: '
  astraImage=$(docker load --input ${astraImageFile} | sed 's/Loaded
image: //')
  astraImage=$(echo ${astraImage} |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

播客：

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

找到操作员安装页面

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



：

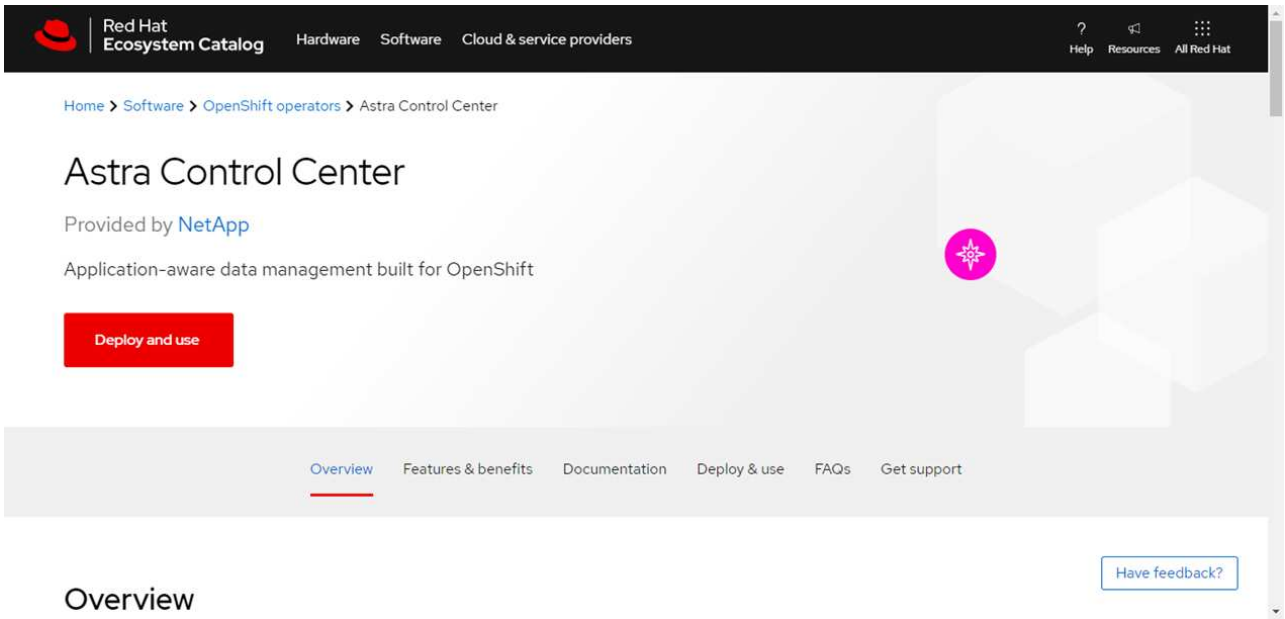
i. 登录到 OpenShift 容器平台 UI 。

ii. 从侧面菜单中，选择 * 运算符 > OperatorHub * 。

iii. 选择 NetApp Astra Control Center 操作员。

iv. 选择 * 安装 * 。

- 从 Red Hat 生态系统目录
：



i. 选择 NetApp Astra 控制中心 "运算符"。

ii. 选择 * 部署并使用 * 。

安装操作员

1. 完成 * 安装操作员 * 页面并安装操作员：

> 操作员将在所有集群命名空间中可用。

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

b. 选择手动或自动批准策略。

> 建议手动批准。每个集群只能运行一个操作员实例。

c. 选择 * 安装 * 。

> 如果您选择了手动批准策略，系统将提示您批准此操作员的手动安装计划。

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

安装 **Astra** 控制中心

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

下一步行动

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

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

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

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

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

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

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

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

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

AWS所需的功能

在 AWS 中部署 Astra 控制中心之前，您需要满足以下条件：

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

AWS 的操作环境要求

Astra 控制中心需要以下 AWS 操作环境：

- Red Hat OpenShift 容器平台 4.8

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

除了环境的资源要求之外， Astra 控制中心还需要以下资源：

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

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

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

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

AWS 部署概述

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

下面详细介绍了其中每个步骤。

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

确保您具有足够的 IAM 权限

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

请参见 "初始 AWS 凭据"。

在 AWS 上安装 RedHat OpenShift 集群

在 AWS 上安装 RedHat OpenShift 容器平台集群。

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

配置 **AWS**

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

按照 AWS 文档完成以下步骤。请参见 "AWS 安装文档"。

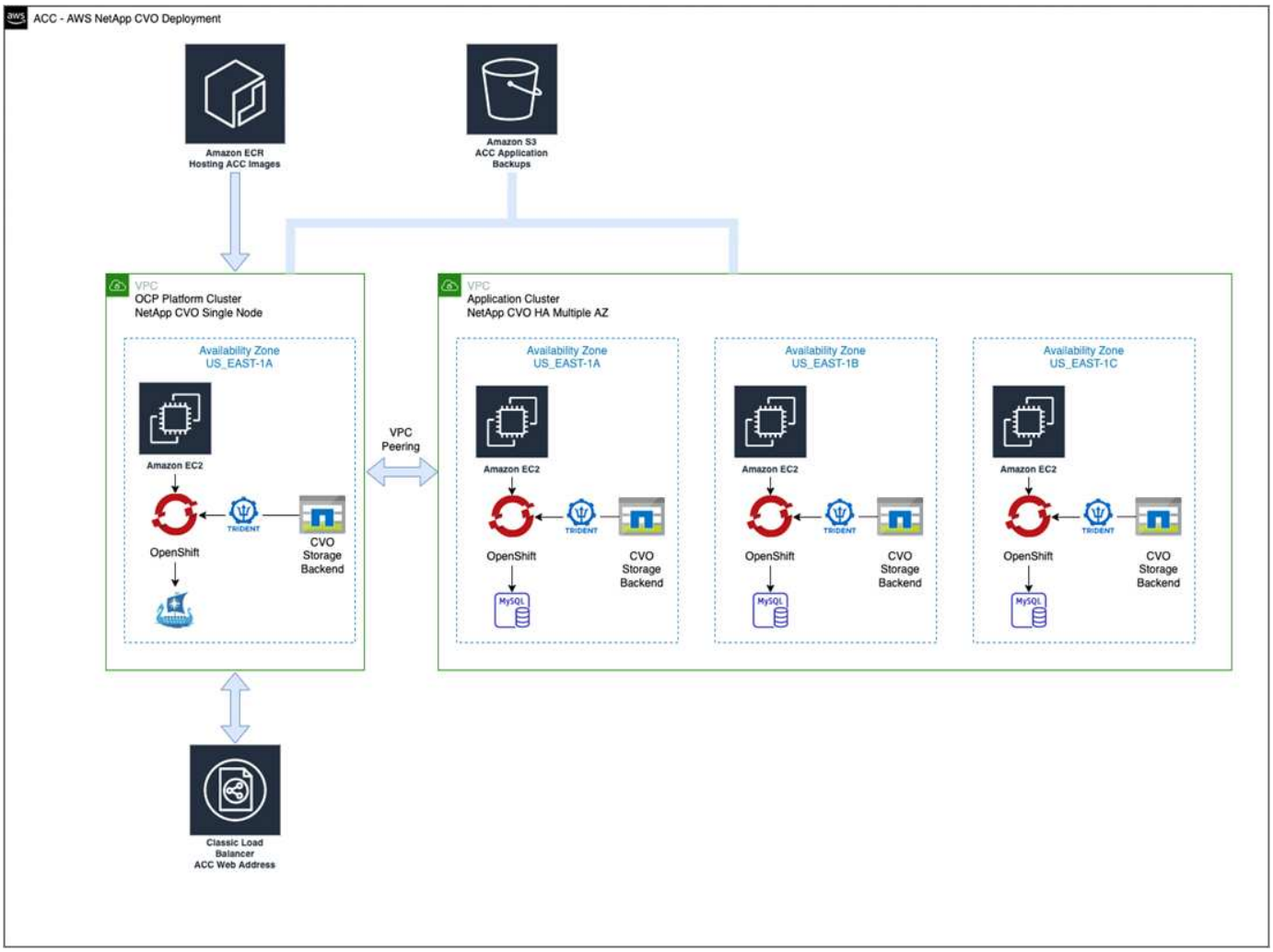
1. 创建AWS虚拟网络。
2. 查看 EC2 计算实例。这可以是 AWS 中的裸机服务器或 VM 。
3. 如果实例类型尚未与主节点和工作节点的 Astra 最低资源要求匹配，请更改 AWS 中的实例类型以满足 Astra 要求。 请参见 "Astra 控制中心要求"。
4. 至少创建一个 AWS S3 存储分段来存储备份。
5. 创建 AWS 弹性容器注册表（ ECR ）以托管所有 AccR 映像。

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

6. 将这些 Accc 映像推送到您定义的注册表。

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

以下是 AWS 部署示例：



配置 **NetApp Cloud Manager**

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

按照 Cloud Manager 文档完成以下步骤。请参见以下内容：

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

步骤

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

b. 在右上角，记下 Trident 版本。

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

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

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

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

安装 **Astra** 控制中心

请遵循标准 "Astra 控制中心安装说明"。

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

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

Azure所需的功能

在 Azure 中部署 Astra 控制中心之前，您需要满足以下条件：

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

Azure 的操作环境要求

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

除了环境的资源要求之外， Astra 控制中心还需要以下资源：

请参见 "Astra 控制中心运营环境要求"。

组件	要求
后端 **NetApp Cloud Volumes ONTAP** 存储容量	至少 300 GB 可用
员工节点（ **Azure** 计算要求）	总共至少 3 个辅助节点，每个节点有 4 个 vCPU 核心和 12 GB RAM
负载平衡器	服务类型 "loadbalancer" 可用于将传入流量发送到操作环境集群中的服务
FQDN （ **Azure DNS** 区域）	一种将 Astra 控制中心的 FQDN 指向负载平衡 IP 地址的方法

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

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

Azure 部署概述

下面简要介绍了适用于 Azure 的 Astra 控制中心的安装过程。

下面详细介绍了其中每个步骤。

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

在 Azure 上安装 RedHat OpenShift 集群

第一步是在 Azure 上安装 RedHat OpenShift 集群。

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

创建 **Azure** 资源组

至少创建一个 Azure 资源组。

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

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

确保您具有足够的 **IAM** 权限

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

请参见 "Azure 凭据和权限"。

配置 **Azure**

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

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

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

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

   - 示例 * ：

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

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

10. 设置 DNS 区域。

配置 NetApp Cloud Manager

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

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

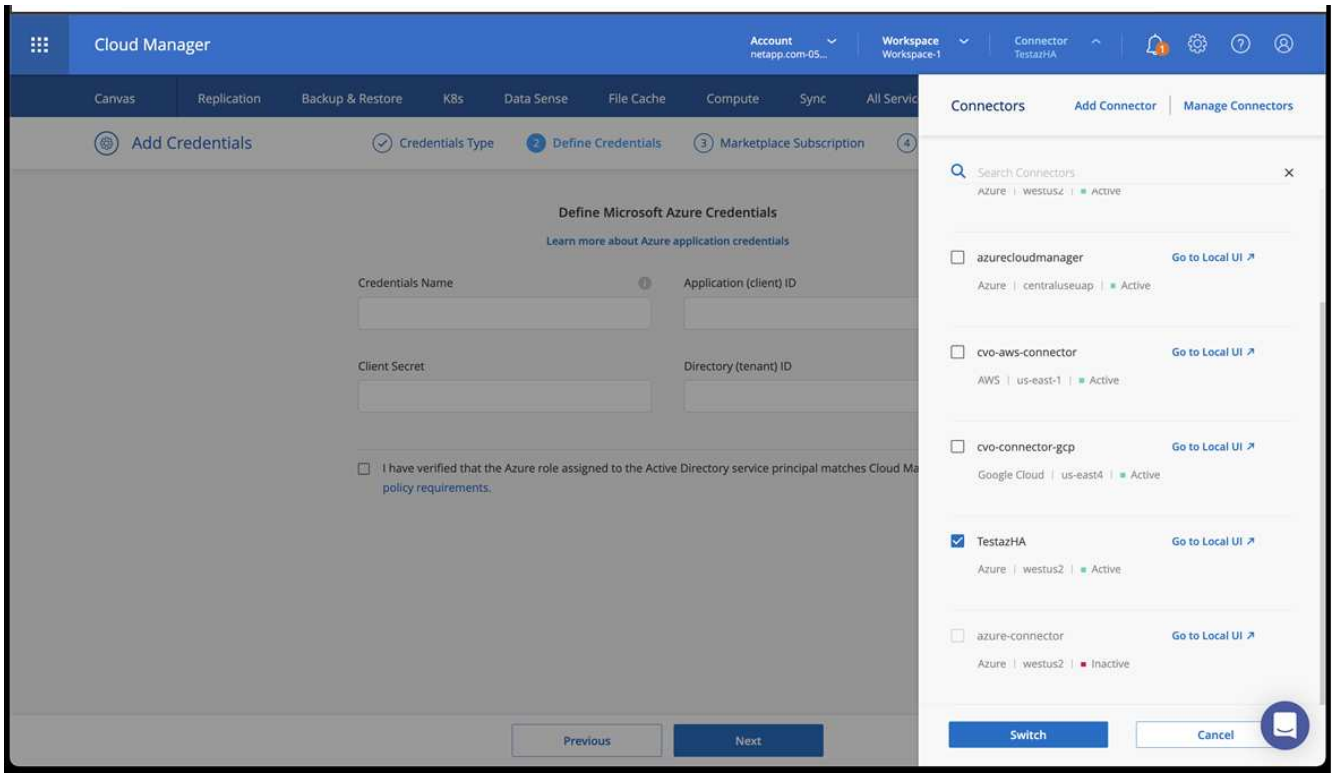
您需要的内容

使用所需的 IAM 权限和角色访问 Azure 帐户

步骤

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

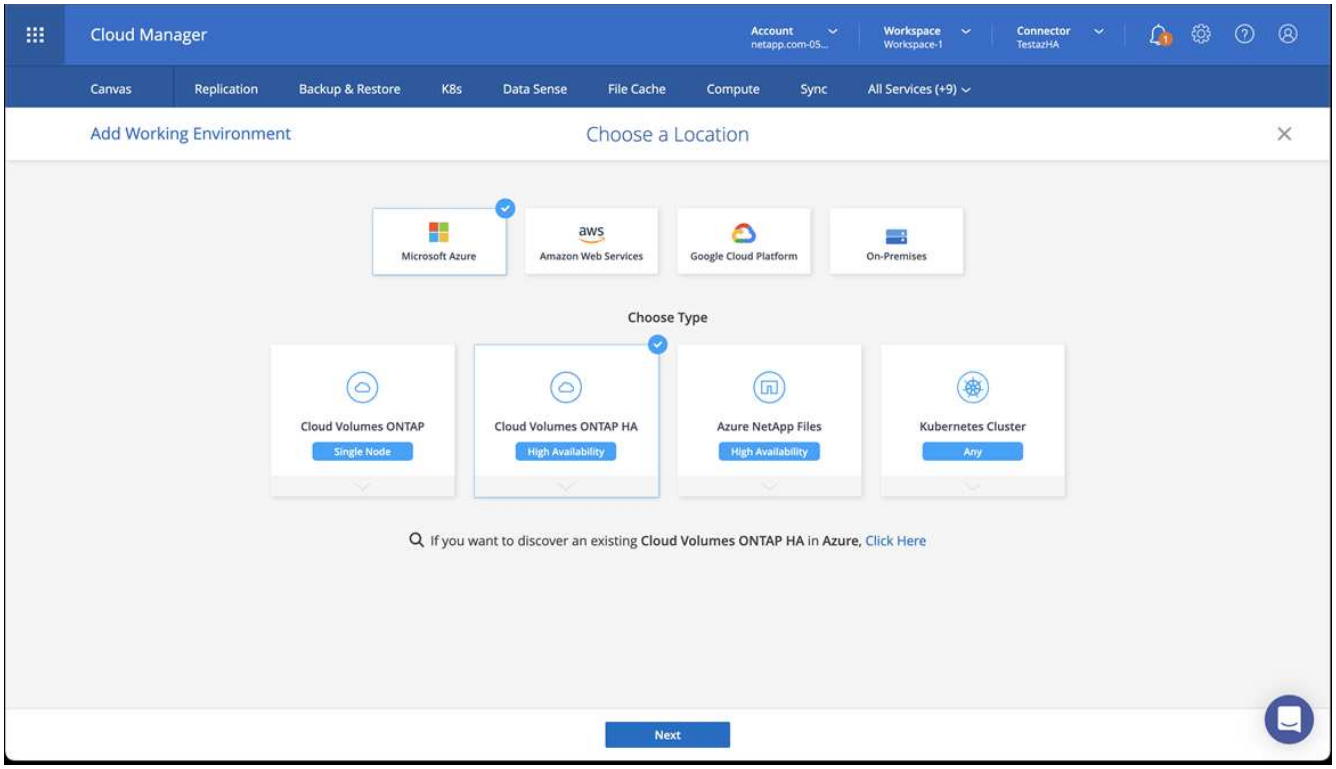
      请参见 "从 Cloud Manager 在 Azure 中创建连接器"。
3. 确保连接器正在运行，然后切换到该连接器。



4. 为您的云环境创建一个工作环境。

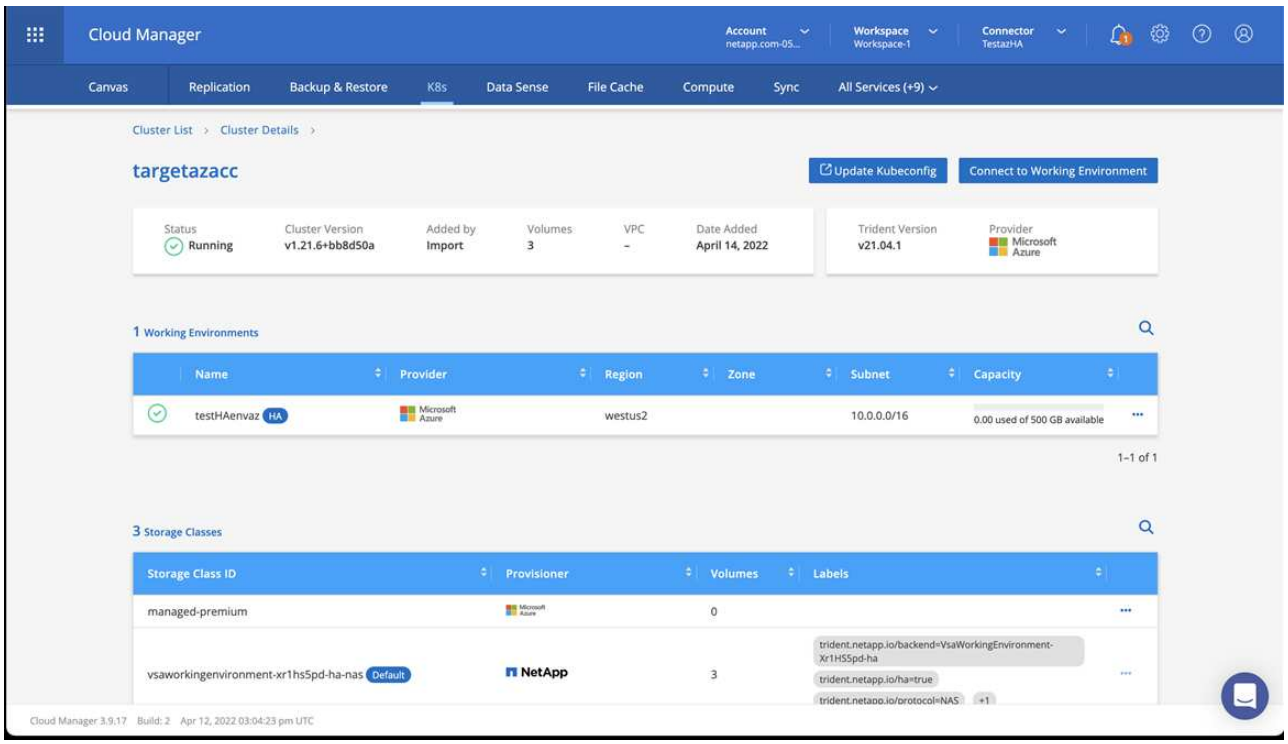
   a. 位置： "Microsoft Azure" 。

   b. 键入： Cloud Volumes ONTAP HA 。



5. 导入 OpenShift 集群。集群将连接到您刚刚创建的工作环境。

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



b. 在右上角，记下 Trident 版本。

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

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

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

安装和配置 **Astra** 控制中心

按照标准安装 Astra 控制中心 "安装说明"。

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

设置 Astra 控制中心

Astra 控制中心支持并监控 ONTAP 和 Astra 数据存储作为存储后端。安装 Astra 控制中心，登录到 UI 并更改密码后，您将需要设置许可证，添加集群，管理存储以及添加存储分段。

任务

- 添加 Astra 控制中心的许可证
- [添加集群]
- [添加存储后端]

- [添加存储分段]

添加 Astra 控制中心的许可证

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

有关如何计算许可证的详细信息，请参见 "许可"。

> 要更新现有评估版或完整许可证，请参见 "更新现有许可证"。

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

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

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

您需要的内容

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

步骤

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

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

添加集群

要开始管理应用程序，请添加 Kubernetes 集群并将其作为计算资源进行管理。您必须为 Astra 控制中心添加一个集群，才能发现您的 Kubernetes 应用程序。对于 Astra 数据存储，您希望添加 Kubernetes 应用程序集群，其中包含使用由 Astra 数据存储配置的卷的应用程序。

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

当Astra Control管理集群时、它会跟踪集群的默认存储类。如果使用`kubectl`命令更改存储类、则Astra Control将还原此更改。要更改由Astra Control管理的集群中的默认存储类、请使用以下方法之一：

- 使用Astra Control API `PUT /managedClusters` Endpoint、并使用`DefaultStorageClass`参数分配其他默认存储类。
- 使用Astra Control Web UI分配其他默认存储类。请参见 [更改默认存储类]。

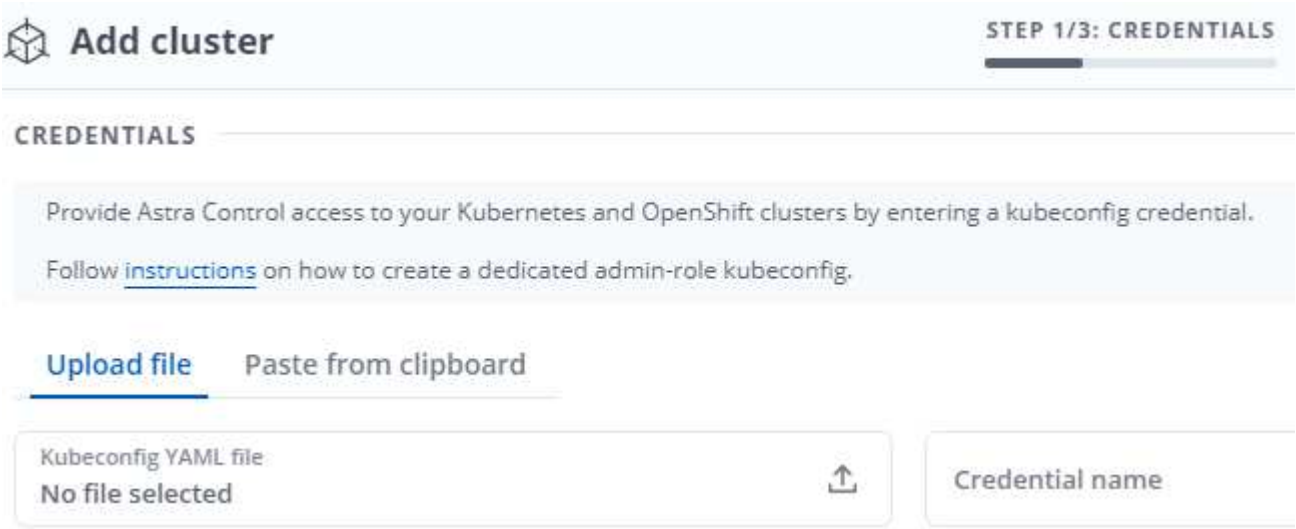
您需要的内容

- 在添加集群之前，请查看并执行必要的操作 "前提条件任务"。

步骤

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

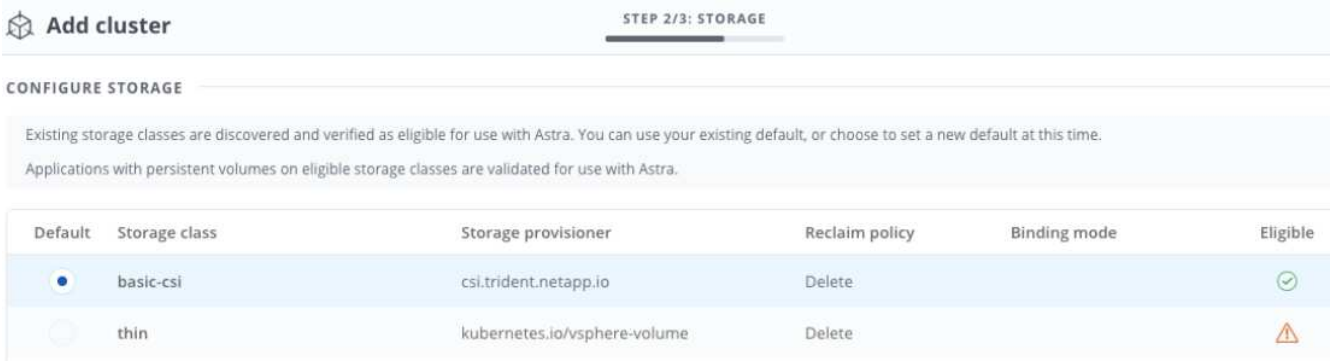
   `kubeconfig.yaml` 文件应仅包含一个集群的集群凭据 * 。



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

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

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



6. 查看相关信息，如果一切正常，请选择 * 添加集群 * 。

结果

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

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

添加存储后端

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

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

现有**Astra Data Store**部署所需的资源

- 您已添加Kubernetes应用程序集群和底层计算集群。

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

新的**Astra Data Store**部署所需的资源

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

选项

- [部署存储资源]
- [使用现有存储后端]

部署存储资源

您可以部署新的Astra数据存储并管理关联的存储后端。

步骤

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

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

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

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

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

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

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

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

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

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

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

结果

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

> 您可能需要刷新页面才能显示后端。

使用现有存储后端

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

步骤

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

结果

后端会在列表中显示为 `available` 状态，并显示摘要信息。

您可能需要刷新页面才能显示后端。

添加存储分段

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

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

如果要将应用程序配置和永久性存储克隆到同一集群，则不需要存储分段。

使用以下任一存储分段类型：

- NetApp ONTAP S3
- NetApp StorageGRID S3
- 通用 S3

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

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

步骤

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

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

   c. 创建新的存储分段名称或输入现有存储分段名称和可选的问题描述。

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

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

      创建的第一个存储分段不会显示此选项。

   f. 通过添加继续 凭据信息。

添加 **S3** 访问凭据

随时添加 S3 访问凭据。

步骤

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

更改默认存储类

您可以更改集群的默认存储类。

步骤

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

下一步是什么?

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

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

了解更多信息

- "使用 Astra Control API"
- "已知问题"

添加集群的前提条件

在添加集群之前，应确保满足前提条件。您还应运行资格检查，以确保集群已准备好添加到 Astra 控制中心。

添加集群之前需要满足的要求

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

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

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

- 使用配置了的Astra Trident StorageClasses "支持的存储后端" (对于任何类型的集群都是必需的)
- 在备份 ONTAP 系统上设置的超级用户和用户 ID ，用于使用 Astra 控制中心备份和还原应用程序。在 ONTAP 命令行中运行以下命令： `export-policy rule modify -vserver <storage virtual machine name> -policyname <policy name> -ruleindex 1 -superuser sysm -anon 65534`
- 管理员定义的 Astra Trident `volumesnapshotclass` 对象。请参见 Astra Trident "说明" 使用 Astra Trident 启用和测试卷快照。
- 确保您仅为 Kubernetes 集群定义了一个默认存储类。

运行资格检查

运行以下资格检查，以确保您的集群已准备好添加到 Astra 控制中心。

步骤

1. 检查 Trident 版本。

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

如果存在 Trident ，您将看到类似于以下内容的输出：

```
NAME       VERSION
trident    21.04.0
```

如果 Trident 不存在，您将看到类似于以下内容的输出：

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

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

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

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

创建管理员角色 **kubeconfig**

执行这些步骤之前，请确保您的计算机上具有以下内容：

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

步骤

1. 按如下所示创建服务帐户：

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

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

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

+

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

   a. 应用服务帐户：

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

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

   a. 创建一个 `ClusterRoleBindingm` 文件，该文件名为 `astracontrol — clusterrolebind.YAML` 。

      根据需要调整创建服务帐户时修改的任何名称和命名空间。

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

+

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

   a. 应用集群角色绑定：

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

4. 列出服务帐户密码，将 ` <context>` 替换为适用于您的安装的正确上下文：

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

输出的结尾应类似于以下内容：

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

sec白 烟 数组中每个元素的索引均以 0 开头。在上面的示例中， `asacontrol service-account-dockercfg-vhz87` 的索引为 0 ， `asacontrol service-account-token-r59rk` 的索引为 1 。在输出中，记下包含 "token" 一词的服务帐户名称的索引。

5. 按如下所示生成 kubeconfig：
   a. 创建 `create-kubeconfig.sh` 文件。将以下脚本开头的 `token_index` 替换为正确的值。

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

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

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

CONTEXT=$(kubectl config current-context)

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

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

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

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

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

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

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

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

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

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

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

b. 获取用于将其应用于 Kubernetes 集群的命令。

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

6. （* 可选 *）将 kubeconfig 重命名为集群的有意义名称。保护集群凭据。

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

下一步是什么？

确认满足了这些前提条件后，您便已准备就绪 "添加集群"。

了解更多信息

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

添加自定义 TLS 证书

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

您需要的内容

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

删除自签名证书

删除现有的自签名 TLS 证书。

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

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

3. 使用以下命令删除当前安装的密钥和证书：

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

添加新证书

添加一个由 CA 签名的新 TLS 证书。

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

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

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

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

#spec:
#  selfSigned: {}

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

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

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

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

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

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

5. 使用以下命令创建证书：

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

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

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

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

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

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

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

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

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

创建自定义 POD 安全策略

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

步骤

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

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

2. 为 POD 安全策略创建新角色。

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

3. 将新角色绑定到服务帐户。

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

有关 Astra 控制中心的常见问题

如果您只是想快速了解问题解答，此常见问题解答会很有帮助。

概述

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

访问 Astra 控制中心

- 什么是 Astra Control URL？ *

Astra 控制中心使用本地身份验证以及每个环境专用的 URL 。

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

- 我正在使用评估版许可证。如何更改为完整许可证？ *

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

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

可以，您可以使用评估版许可证测试管理应用程序功能。

注册 Kubernetes 集群

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

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

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

从 Astra Control 中删除集群不会对集群的配置（应用程序和永久性存储）进行任何更改。对该集群上的应用程序执行的任何 Astra Control 快照或备份都将无法还原。由 Astra Control 创建的永久性存储备份仍保留在 Astra Control 中，但无法还原。

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

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

管理应用程序

- Astra Control 是否可以部署应用程序？ *

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

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

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

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

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

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

数据管理操作

- 我的帐户中存在未创建的快照。它们来自何处？ *

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

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

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

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

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

使用 Astra

管理应用程序

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 中的应用程序页面开始管理应用程序及其资源。

有关详细信息，请参见 "应用程序管理要求"。

支持的应用程序安装方法

Astra Control 支持以下应用程序安装方法：

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

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

- * Helm 3* ：如果使用 Helm 安装应用程序，则 Astra Control 需要 Helm 版本 3 。完全支持管理和克隆随 Helm 3 安装的应用程序（或从 Helm 2 升级到 Helm 3 ）。不支持管理随 Helm 2 安装的应用程序。
- * 操作员部署的应用程序 * ： Astra Control 支持使用命名空间范围的运算符安装的应用程序。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"
  - "Jenkins CI"
  - "Percona XtraDB 集群"

请注意， 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。

> 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。

在集群上安装应用程序

现在，您已将集群添加到 Astra Control 中，您可以在集群上安装应用程序或管理现有应用程序。可以管理范围限定于命名空间的任何应用程序。Pod 联机后，您可以使用 Astra Control 管理应用程序。

有关从 Helm 图表部署经过验证的应用程序的帮助，请参见以下内容：

- "从 Helm 图表部署 MariaDB"
- "从 Helm 图表部署 MySQL"
- "从 Helm 图表部署 Postgres"
- "从 Helm 图表中部署 Jenkins"

管理应用程序

使用 Astra Control 可以在命名空间级别或通过 Kubernetes 标签管理应用程序。

> 不支持随 Helm 2 安装的应用程序。

您可以执行以下活动来管理应用程序：

- 管理应用程序
  - [按命名空间管理应用程序]
  - 按 Kubernetes 标签管理应用程序
- [忽略应用程序]
- [取消管理应用程序]

> Astra Control 本身不是一个标准应用程序，而是一个 " 系统应用程序 " 。 您不应尝试管理 Astra Control 本身。默认情况下，用于管理的 Astra Control 本身不会显示。要查看系统应用程序，请使用 " 显示系统应用程序 " 筛选器。

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"。

> 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。

按命名空间管理应用程序

" 应用程序 " 页面的 * 已发现 * 部分显示命名空间以及这些命名空间中由 Helm 安装的任何应用程序或自定义标记的应用程序。您可以选择单独管理每个应用程序，也可以选择在命名空间级别管理每个应用程序。这一切都可以细化到数据保护操作所需的粒度级别。

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，您需要单独管理应用程序，而不是在一个命名空间下进行管理。

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），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。

步骤

1. 从左侧导航栏中，选择 * 应用程序 * 。
2. 选择 * 已发现 * 筛选器。

3. 查看已发现的命名空间列表。展开命名空间以查看应用程序和关联资源。

   Astra Control 会向您显示命名空间中的 Helm 应用程序和自定义标记的应用程序。如果 Helm 标签可用，则会使用标记图标来指定这些标签。

4. 查看 * 组 * 列，查看应用程序运行在哪个命名空间中（使用文件夹图标指定）。
5. 确定是单独管理每个应用程序，还是在命名空间级别管理每个应用程序。
6. 在层次结构中的所需级别找到所需的应用程序，然后从 * 操作 * 列的选项菜单中选择 * 管理 * 。
7. 如果您不想管理某个应用程序，请从 * 操作 * 列的选项菜单中选择 * 忽略 * 。

   例如，如果您希望同时管理 "Maria" 命名空间下的所有应用程序，以便它们具有相同的快照和备份策略，则可以管理命名空间并忽略命名空间中的应用程序。

8. 要查看受管应用程序的列表，请选择 * 受管 * 作为显示筛选器。

> 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。

9. 要查看特定应用程序的详细信息，请选择应用程序名称。

结果

您选择管理的应用程序现在可从 * 受管 * 选项卡访问。任何被忽略的应用程序都将移至 * 已忽略 * 选项卡。理想情况下， " 已发现 " 选项卡将显示零个应用程序，以便在安装新应用程序后更容易找到和管理这些应用程序。

按 **Kubernetes** 标签管理应用程序

Astra Control 在应用程序页面顶部包含一个名为 * 定义自定义应用程序 * 的操作。您可以使用此操作管理使用 Kubernetes 标签标识的应用程序。 "了解有关通过 Kubernetes 标签定义自定义应用程序的更多信息"。

步骤

1. 从左侧导航栏中，选择 * 应用程序 * 。
2. 选择 * 定义 * 。
3. 在 * 定义自定义应用程序 * 对话框中，提供管理该应用程序所需的信息：
   a. * 新建应用程序 * ： 输入应用程序的显示名称。
   b. * 集群 * ： 选择应用程序所在的集群。
   c. * 命名空间： * 选择应用程序的命名空间。
   d. * 标签： * 输入标签或从以下资源中选择标签。
   e. * 选定资源 * ： 查看和管理要保护的选定 Kubernetes 资源（ Pod ，机密，永久性卷等）。
      - 通过展开资源并选择标签数量来查看可用标签。
      - 选择一个标签。

        选择标签后，它将显示在 * 标签 * 字段中。Astra Control 还会更新 * 未选定资源 * 部分，以显示与选定标签不匹配的资源。

   f. * 未选择资源 * ： 验证您不想保护的应用程序资源。
4. 选择 * 定义自定义应用程序 * 。

结果

使用 Astra Control 可以管理应用程序。现在，您可以在 * 受管 * 选项卡中找到它。

忽略应用程序

如果已发现某个应用程序，它将显示在已发现列表中。在这种情况下，您可以清理已发现的列表，以便更容易找到新安装的应用程序。或者，您可能会管理一些应用程序，稍后决定不再需要管理这些应用程序。如果您不想管理这些应用程序，可以指示应忽略它们。

此外，您可能希望在一个命名空间下同时管理应用程序（命名空间管理）。您可以忽略要从命名空间中排除的应用程序。

步骤

1. 从左侧导航栏中，选择 * 应用程序 * 。
2. 选择 * 已发现 * 作为筛选器。
3. 选择应用程序。
4. 从选项菜单的 * 操作 * 列中，选择 * 忽略 * 。
5. 要取消忽略，请选择 * 取消忽略 * 。

取消管理应用程序

如果您不再需要备份，创建快照或克隆某个应用程序，则可以停止对其进行管理。

> 如果取消管理某个应用程序，则先前创建的任何备份或快照都将丢失。

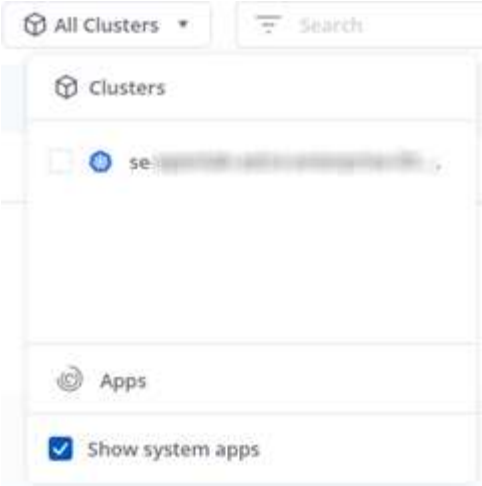
步骤

1. 从左侧导航栏中，选择 * 应用程序 * 。
2. 选择 * 受管 * 作为筛选器。
3. 选择应用程序。
4. 从选项菜单的 * 操作 * 列中，选择 * 取消管理 * 。
5. 查看相关信息。
6. 键入 "unmanage" 进行确认。
7. 选择 * 是，取消管理应用程序 * 。

系统应用程序如何？

Astra Control 还会发现 Kubernetes 集群上运行的系统应用程序。默认情况下，我们不会向您显示这些系统应用程序，因为您很少需要备份这些应用程序。

您可以从 " 应用程序 " 页面显示系统应用程序，方法是选中工具栏中 " 集群 " 筛选器下的 * 显示系统应用程序 * 复选框。



> Astra Control 本身不是一个标准应用程序，而是一个 " 系统应用程序 " 。 您不应尝试管理 Astra Control 本身。默认情况下，用于管理的 Astra Control 本身不会显示。

了解更多信息

- "使用 Astra Control API"

定义自定义应用示例

通过创建自定义应用程序，您可以将 Kubernetes 集群中的元素分组到一个应用程序中。此 Kubernetes 资源集合基于命名空间和标签。

通过自定义应用程序，您可以更精细地控制要包含在 Astra Control 操作中的内容，其中包括：

- 克隆
- Snapshot
- 备份
- 保护策略

大多数情况下，您需要在整个应用程序上使用 Astra Control 的功能。但是，您也可以创建一个自定义应用程序，以便通过为命名空间中的 Kubernetes 对象分配的标签来使用这些功能。

> 只能在单个集群上的指定命名空间中创建自定义应用程序。Astra Control 不支持自定义应用程序跨越多个命名空间或集群。

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"。

> 名称不同的同一资源的重叠策略可能会发生发生原因数据冲突。如果要为某个资源创建自定义应用程序，请确保不会根据任何其他策略克隆或备份该应用程序。

您需要的内容

- 已添加到 Astra Control 的集群

步骤

1. 从 "Apps" 页面中，选择 "++ define （超过定义） "* 。

   " 自定义应用程序 " 窗口将显示哪些资源将包含在您的自定义应用程序中或从该应用程序中排除。这有助于您确保选择正确的标准来定义自定义应用程序。

2. 在弹出窗口中，输入应用程序名称，在 "* 集群 "* 下拉列表中选择集群，然后从 "* 命名空间 "* 下拉列表中选择应用程序的命名空间。
3. 从 * 标签 * 下拉列表中，选择应用程序和命名空间的标签。
4. 为一个部署定义自定义应用程序后，根据需要对其他部署重复此过程。

创建完这两个自定义应用程序后，您可以将这些资源视为任何其他 Astra Control 应用程序。他们可以克隆这些资源，创建备份和快照，并根据 Kubernetes 标签为每个资源组创建自定义保护策略。

示例：不同版本的单独保护策略

在此示例中， DevOps 团队正在管理一个 Canary 版本部署。他们的集群中有三个 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，管理员将创建两个自定义应用程序：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。

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"。

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] 备份所有应用程序

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] 可选：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] 发生灾难时，请还原您的应用程序

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）。

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应用程序。您可以使用 Astra UI 或 "Astra Control API" 保护应用程序。

> 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。

> 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
> ```

配置保护策略

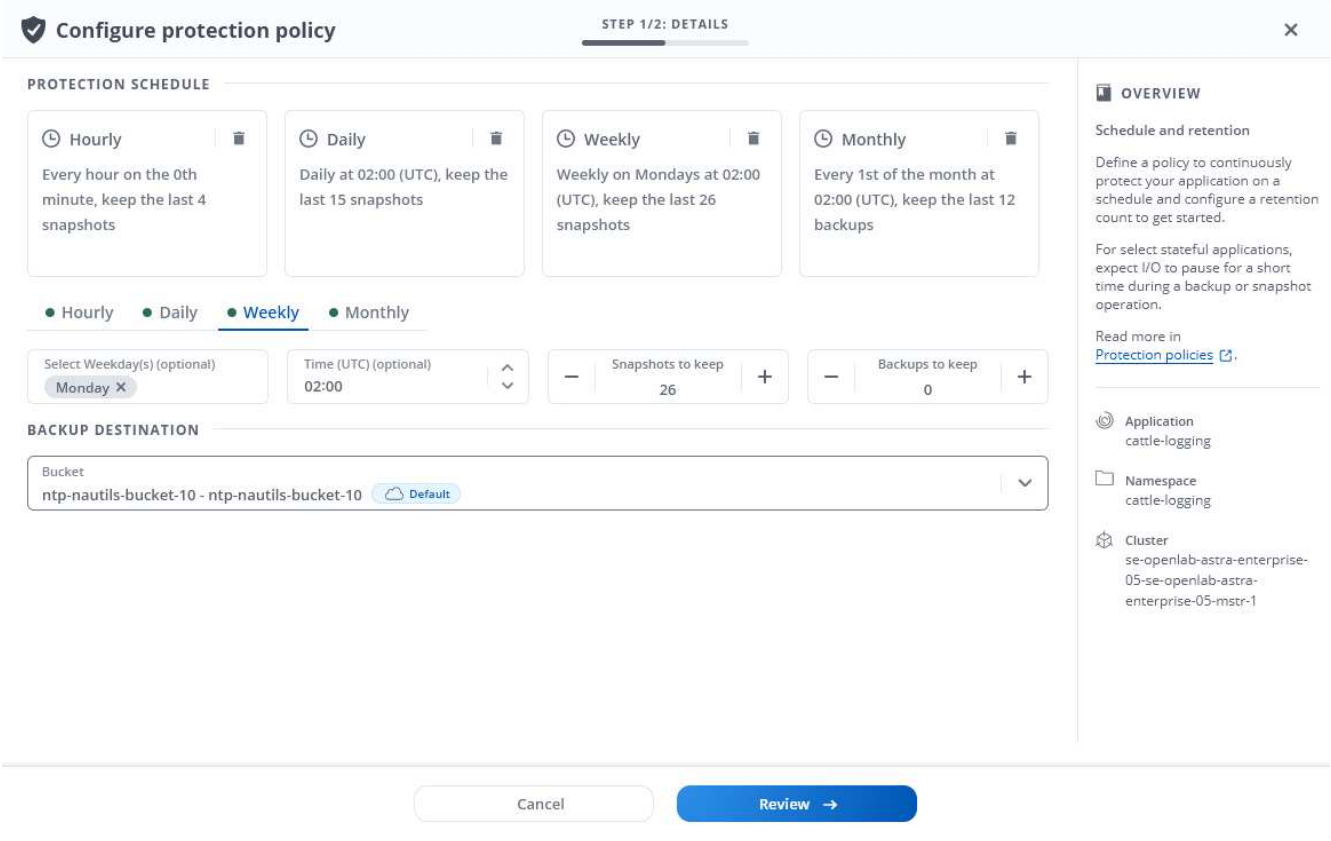
保护策略通过按定义的计划创建快照，备份或这两者来保护应用程序。您可以选择每小时，每天，每周和每月创建快照和备份，并且可以指定要保留的副本数。例如，保护策略可能会创建每周备份和每日快照，并将备份和快照保留一个月。创建快照和备份的频率以及保留时间取决于组织的需求。

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。

   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. 选择 * 审阅 * 。
6. 选择 * 设置保护策略。 *

结果

Astra 控制中心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。

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 * 审阅 * 。
4. 查看快照摘要并选择 * 快照 * 。

结果

快照过程开始。如果在 * 数据保护 * > * 快照 * 页面的 * 操作 * 列中的状态为 * 可用 * ，则快照将成功。

创建备份

您也可以随时备份应用程序。

> 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。

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. 通过从存储分段列表中选择来选择备份的目标。
6. 选择 * 审阅 * 。
7. 查看备份摘要并选择 * 备份 * 。

结果

Astra 控制中心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。

> 无法停止正在运行的备份。如果需要删除备份，请等待备份完成，然后按照中的说明进行操作 [删除备份]。删除失败的备份， "使用 Astra Control API"。

> 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。

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。

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. 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 控制中心会删除快照。

删除备份

删除不再需要的计划备份或按需备份。

> 无法停止正在运行的备份。如果需要删除备份，请等待备份完成，然后按照以下说明进行操作。删除失败的备份， "使用 Astra Control API"。

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 控制中心删除备份。

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。

关于此任务

- 强烈建议在还原应用程序之前为其创建快照或备份。这样、您可以在还原失败时从快照或备份克隆。
- 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。
- 如果要还原到其他集群，请确保此集群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。
- 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。通过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 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 * ，然后选择应用程序的名称。
2. 选择 * 数据保护 * 。
3. 如果要从快照还原，请保持选中 * 快照 * 图标。否则，请选择 * 备份 * 图标以从备份中还原。
4. 从要还原的快照或备份的 * 操作 * 列的选项菜单中，选择 * 还原应用程序 * 。
5. * 还原详细信息 * ： 指定已还原应用程序的详细信息。默认情况下，将显示当前集群和命名空间。保留这些值不变，以便原位还原应用程序，从而将应用程序还原到其自身的早期版本。如果要还原到其他集群或命名空间，请更改这些值。
   - 输入应用程序的名称和命名空间。
   - 选择应用程序的目标集群。
   - 选择 * 审阅 * 。

   > 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。

6. * 还原摘要 * ： 查看有关还原操作的详细信息，键入 "restore" ，然后选择 * 还原 * 。

结果

Astra 控制中心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，则任何现有永久性卷的内容将替换为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、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。

克隆和迁移应用程序

克隆现有应用程序以在同一个 Kubernetes 集群或另一个集群上创建重复的应用程序。当 Astra 控制中心克隆应用程序时，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 UI 或 "Astra Control API" 克隆和迁移应用程序。

您需要的内容

要将应用程序克隆到其他集群，您需要一个默认存储分段。添加第一个存储分段时，它将成为默认存储分段。

关于此任务

- 如果您部署的应用程序明确设置了 StorageClass ，并且需要克隆该应用程序，则目标集群必须具有最初指定的 StorageClass 。将显式设置了 StorageClass 的应用程序克隆到不具有相同 StorageClass 的集群将失败。
- 如果克隆操作员部署的 Jenkins CI 实例，则需要手动还原永久性数据。这是应用程序部署模式的一个限制。
- 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。

- 在应用程序备份或应用程序还原期间，您可以选择指定存储分段 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。
- 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。通过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。

OpenShift 注意事项

- 如果您在集群之间克隆应用程序，则源集群和目标集群必须是 OpenShift 的同一分发版。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。
- 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. * 克隆详细信息 * ： 指定克隆的详细信息：
   - 输入名称。
   - 输入克隆的命名空间。
   - 选择克隆的目标集群。
   - 选择是要从现有快照还是备份创建克隆。如果不选择此选项，则 Astra 控制中心将根据应用程序的当前状态创建克隆。
5. * 源 * ： 如果选择从现有快照或备份克隆，请选择要使用的快照或备份。
6. 选择 * 审阅 * 。
7. * 克隆摘要 * ： 查看有关克隆的详细信息并选择 * 克隆 * 。

结果

Astra 控制中心会根据您提供的信息克隆该应用程序。如果新应用程序克隆在 * 应用程序 * 页面上处于 可用 状态，则克隆操作将成功。

> 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。

管理应用程序执行挂钩

执行钩是一种自定义脚本，您可以在托管应用程序快照之前或之后运行该脚本。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。

默认执行挂钩和正则表达式

对于某些应用程序， Astra Control 附带了 NetApp 提供的默认执行挂钩，用于处理快照前后的冻结和解冻操作。Astra Control 使用正则表达式将应用程序的容器映像与以下应用程序匹配：

- MariaDB
  - 匹配正则表达式： \bmariadb\b
- MySQL
  - 匹配正则表达式： \bmysql\b
- PostgreSQL
  - 匹配正则表达式： \bpostgresql\b

如果存在匹配项，则 NetApp 为该应用程序提供的默认执行挂钩将显示在该应用程序的活动执行挂钩列表中，这些挂钩将在该应用程序创建快照时自动运行。如果某个自定义应用程序的映像名称类似，恰好与其中一个正则表达式匹配（并且您不想使用默认执行挂钩），则可以更改映像名称， 或者禁用该应用程序的默认执行连接，而改用自定义连接。

您不能删除或修改默认执行挂钩。

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。

- Astra Control 要求以可执行 Shell 脚本的格式编写执行挂钩。
- 脚本大小限制为 128 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。
- 所有执行挂机故障均为软故障；即使某个挂机发生故障，仍会尝试使用其他挂机和快照。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。

> 由于执行挂钩通常会减少或完全禁用其所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。

运行快照时，执行钩事件按以下顺序发生：

1. NetApp 提供的任何适用的默认快照前执行挂钩都会在相应的容器上运行。
2. 任何适用的自定义快照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预快照挂钩，但在创建快照之前执行这些挂钩的顺序既不能保证也不可配置。

3. 执行快照。
4. 任何适用的自定义快照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快照后挂钩，但这些挂钩在快照后的执行顺序既不能保证也不可配置。
5. NetApp 提供的任何适用的默认快照后执行挂钩都会在相应的容器上运行。

> 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，测试生成的快照以确保其一致。为此，您可以将应用程序克隆到临时命名空间，还原快照，然后测试应用程序。

查看现有执行挂钩

您可以查看应用程序的现有自定义或 NetApp 提供的默认执行挂钩。

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。您可以查看挂机的状态，源以及运行时间（快照前或快照后）。要查看与执行挂钩相关的事件日志，请转到左侧导航区域中的 * 活动 * 页面。

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。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 Linux 命令或提供可执行文件的完整路径。

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. 选择 * 添加新挂钩 * 。
4. 在 * 挂机详细信息 * 区域中，根据挂机应运行的时间，选择 * 预 Snapshot * 或 * 后 Snapshot * 。
5. 输入此挂钩的唯一名称。
6. （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。
7. 在 * 容器映像 * 区域中，如果此挂钩应针对应用程序中包含的所有容器映像运行，请启用 * 应用于所有容器映像 * 复选框。如果该挂钩只能作用于一个或多个指定的容器映像，请在 * 要匹配的容器映像名称 * 字段中输入容器映像名称。
8. 在 * 脚本 * 区域中，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。
  - 从剪贴板粘贴到自定义脚本中。
    i. 选择 * 从剪贴板粘贴 * 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。
9. 选择 * 添加挂钩 * 。

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 * 。

执行钩示例

使用以下示例了解如何构建执行挂钩。您可以将这些挂钩用作模板或测试脚本。

简单的成功示例

这是一个简单的钩子示例，它成功地将消息写入标准输出和标准错误。

```
#!/bin/sh

# success_sample.sh
#
# A simple noop success hook script for testing purposes.
#
```

```
# args: None
#


#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}


#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}


#
# main
#

# log something to stdout
info "running success_sample.sh"

# exit with 0 to indicate success
info "exit 0"
exit 0
```

简单成功示例（**bash** 版本）

这是一个简单的钩子示例，该钩子成功地将消息写入标准输出和标准错误，并写入 bash 。

```
#!/bin/bash

# success_sample.bash
#
# A simple noop success hook script for testing purposes.
#
# args: None


#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}


#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}


#
# main
#
```

```
# log something to stdout
info "running success_sample.bash"

# exit with 0 to indicate success
info "exit 0"
exit 0
```

简单成功示例（**zsh** 版本）

这是一个简单的钩子示例，该钩子成功地将消息写入标准输出和标准错误，并写入 Z shell 。

```
#!/bin/zsh

# success_sample.zsh
#
# A simple noop success hook script for testing purposes.
#
# args: None
#


#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}


#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
```

```
    msg "ERROR: $*" 1>&2
}

#
# main
#

# log something to stdout
info "running success_sample.zsh"

# exit with 0 to indicate success
info "exit 0"
exit 0
```

成功使用参数示例

以下示例演示了如何在挂机中使用 args 。

```
#!/bin/sh

# success_sample_args.sh
#
# A simple success hook script with args for testing purposes.
#
# args: Up to two optional args that are echoed to stdout

#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}


#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
```

```
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}


#
# main
#

# log something to stdout
info "running success_sample_args.sh"


# collect args
arg1=$1
arg2=$2

# output args and arg count to stdout
info "number of args: $#"
info "arg1 ${arg1}"
info "arg2 ${arg2}"

# exit with 0 to indicate success
info "exit 0"
exit 0
```

快照前 / 快照后挂钩示例

以下示例演示了如何对快照前和快照后挂钩使用同一脚本。

```
#!/bin/sh

# success_sample_pre_post.sh
#
# A simple success hook script example with an arg for testing purposes
# to demonstrate how the same script can be used for both a prehook and
posthook
#
# args: [pre|post]

# unique error codes for every error case
ebase=100
```

```
eusage=$((ebase+1))
ebadstage=$((ebase+2))
epre=$((ebase+3))
epost=$((ebase+4))


#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}


#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}


#
# Would run prehook steps here
#
prehook() {
    info "Running noop prehook"
    return 0
}

#
# Would run posthook steps here
#
```

```
posthook() {
    info "Running noop posthook"
    return 0
}


#
# main
#

# check arg
stage=$1
if [ -z "${stage}" ]; then
    echo "Usage: $0 <pre|post>"
    exit ${eusage}
fi

if [ "${stage}" != "pre" ] && [ "${stage}" != "post" ]; then
    echo "Invalid arg: ${stage}"
    exit ${ebadstage}
fi

# log something to stdout
info "running success_sample_pre_post.sh"

if [ "${stage}" = "pre" ]; then
    prehook
    rc=$?
    if [ ${rc} -ne 0 ]; then
        error "Error during prehook"
    fi
fi

if [ "${stage}" = "post" ]; then
    posthook
    rc=$?
    if [ ${rc} -ne 0 ]; then
        error "Error during posthook"
    fi
fi

exit ${rc}
```

故障示例

以下示例演示了如何处理挂机故障。

```
#!/bin/sh

# failure_sample_arg_exit_code.sh
#
# A simple failure hook script for testing purposes.
#
# args: [the exit code to return]
#


#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}


#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}


#
# main
#

# log something to stdout
info "running failure_sample_arg_exit_code.sh"
```

```
argexitcode=$1

# log to stderr
error "script failed, returning exit code ${argexitcode}"

# exit with specified exit code
exit ${argexitcode}
```

详细故障示例

以下示例演示了如何处理挂机故障，并提供更详细的日志记录。

```
#!/bin/sh

# failure_sample_verbose.sh
#
# A simple failure hook script with args for testing purposes.
#
# args: [The number of lines to output to stdout]


#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}


#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
```

```
    msg "ERROR: $*" 1>&2
}


#
# main
#

# log something to stdout
info "running failure_sample_verbose.sh"


# output arg value to stdout
linecount=$1
info "line count ${linecount}"

# write out a line to stdout based on line count arg
i=1
while [ "$i" -le ${linecount} ]; do
    info "This is line ${i} from failure_sample_verbose.sh"
    i=$(( i + 1 ))
done

error "exiting with error code 8"
exit 8
```

退出代码示例失败

以下示例显示了一个连接失败并显示退出代码。

```
#!/bin/sh

# failure_sample_arg_exit_code.sh
#
# A simple failure hook script for testing purposes.
#
# args: [the exit code to return]
#


#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
```

```
    echo "$*"
}


#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}


#
# main
#

# log something to stdout
info "running failure_sample_arg_exit_code.sh"

argexitcode=$1

# log to stderr
error "script failed, returning exit code ${argexitcode}"

# exit with specified exit code
exit ${argexitcode}
```

失败后成功示例

以下示例显示了首次运行时发生故障的挂钩，但在第二次运行后仍会成功。

```
#!/bin/sh

# failure_then_success_sample.sh
#
```

```
# A hook script that fails on initial run but succeeds on second run for
testing purposes.
#
# Helpful for testing retry logic for post hooks.
#
# args: None
#

#
# Writes the given message to standard output
#
# $* - The message to write
#
msg() {
    echo "$*"
}


#
# Writes the given information message to standard output
#
# $* - The message to write
#
info() {
    msg "INFO: $*"
}

#
# Writes the given error message to standard error
#
# $* - The message to write
#
error() {
    msg "ERROR: $*" 1>&2
}


#
# main
#

# log something to stdout
info "running failure_success sample.sh"


if [ -e /tmp/hook-test.junk ] ; then
    info "File does exist.  Removing /tmp/hook-test.junk"
```

```
    rm /tmp/hook-test.junk
    info "Second run so returning exit code 0"
    exit 0
else
    info "File does not exist.  Creating /tmp/hook-test.junk"
    echo "test" > /tmp/hook-test.junk
    error "Failed first run, returning exit code 5"
    exit 5
fi
```

查看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。

查看集群的运行状况和详细信息

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。

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。

查看应用程序的运行状况和详细信息

开始管理某个应用程序后， Astra 会提供有关该应用程序的详细信息，您可以通过这些详细信息来确定其状态（是否运行正常），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. 查找您需要的信息：

   应用程序状态

   提供反映应用程序在 Kubernetes 中的状态的状态。例如， Pod 和永久性卷是否联机？如果某个应用程序运行状况不正常，您需要查看 Kubernetes 日志，对集群上的问题描述进行故障排除。Astra 不会提供任何信息来帮助您修复损坏的应用程序。

   应用程序保护状态

   提供应用程序受保护程度的状态：

   - * 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

与应用程序关联的 Pod 的状态信息。

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。

Resources

用于验证正在备份和管理哪些资源。

活动

显示了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

管理用户

您可以使用 Astra Control UI 邀请，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。

邀请用户

客户所有者和管理员可以邀请新用户访问 Astra 控制中心。

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 * 邀请用户 * 。
4. 输入用户的名称和电子邮件地址。
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 "管理角色"。

7. 选择 * 邀请用户 * 。

   用户会收到一封电子邮件，告知他们已受邀访问 Astra 控制中心。此电子邮件包含临时密码，需要在首次登录时更改此密码。

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 * 。

更改用户的角色

具有所有者角色的用户可以更改所有用户的角色，而具有管理员角色的用户可以更改具有管理员，成员或查看器角色的用户的角色。

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 * 编辑角色 * 。
4. 选择一个新角色。
5. 要对角色应用约束，请启用 * 将角色限制为约束条件 * 复选框，然后从列表中选择一个约束条件。

   如果没有限制，您可以添加限制。有关详细信息，请参见 "管理角色"。

6. 选择 * 确认 * 。

结果

Astra 控制中心会根据您选择的新角色更新用户的权限。

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添加命名空间约束。

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"

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"。

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

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 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"

管理存储库连接

您可以将存储库连接到Astra Control、以用作软件包安装映像和项目的参考。导入软件包时、Astra Control会引用映像存储库中的安装映像以及项目存储库中的二进制文件和其他项目。

您需要的内容

- 安装了 Astra 控制中心的 Kubernetes 集群
- 一个正在运行的Docker存储库、您可以访问该存储库
- 可访问的正在运行的项目存储库(如Artifactory)

连接**Docker**映像存储库

您可以连接Docker映像存储库以保存软件包安装映像、例如用于Astra数据存储的安装映像。安装软件包时、Astra Control会从映像存储库导入软件包映像文件。

步骤

1. 在 * 管理帐户 * 导航区域中，选择 * 帐户 * 。
2. 选择*连接*选项卡。
3. 在* Docker映像存储库*部分中、选择右上角的菜单。
4. 选择 * 连接 * 。
5. 添加存储库的URL和端口。
6. 输入存储库的凭据。
7. 选择 * 连接 * 。

结果

存储库已连接。在* Docker映像存储库*部分中、存储库应显示已连接状态。

断开**Docker**映像存储库的连接

如果不再需要与Docker映像存储库的连接、您可以将其删除。

步骤

1. 在 * 管理帐户 * 导航区域中，选择 * 帐户 * 。
2. 选择*连接*选项卡。
3. 在* Docker映像存储库*部分中、选择右上角的菜单。
4. 选择*断开连接*。
5. 选择*是、断开Docker映像存储库*。

结果

存储库已断开连接。在* Docker映像存储库*部分中、存储库应显示已断开连接状态。

连接项目存储库

您可以将项目存储库连接到主机项目、例如软件包二进制文件。安装软件包时、Astra Control会从映像存储库导入软件包的项目。

步骤

1. 在 * 管理帐户 * 导航区域中，选择 * 帐户 * 。
2. 选择*连接*选项卡。
3. 在*项目存储库*部分中、选择右上角的菜单。
4. 选择 * 连接 * 。
5. 添加存储库的URL和端口。
6. 如果需要身份验证、请启用*使用身份验证*复选框并输入存储库的凭据。
7. 选择 * 连接 * 。

结果

存储库已连接。在*项目存储库*部分中、存储库应显示已连接状态。

断开项目存储库的连接

如果不再需要与项目存储库的连接、您可以将其删除。

步骤

1. 在 * 管理帐户 * 导航区域中，选择 * 帐户 * 。
2. 选择*连接*选项卡。
3. 在*项目存储库*部分中、选择右上角的菜单。
4. 选择*断开连接*。
5. 选择*是、断开项目存储库*。

结果

存储库已断开连接。在*项目存储库*部分中、存储库应显示已连接状态。

了解更多信息

- "管理软件包"

管理软件包

NetApp通过可从NetApp支持站点下载的软件包为Astra控制中心提供更多功能。连接Docker和项目存储库后、您可以上传并导入软件包、以便将此功能添加到Astra控制中心。您可以使用命令行界面或Astra控制中心Web UI管理软件包。

您需要的内容

- 安装了 Astra 控制中心的 Kubernetes 集群
- 一个连接的Docker映像存储库、用于存放软件包映像。有关详细信息，请参见 "管理存储库连接"。
- 一个连接的项目存储库、用于存放软件包二进制文件和项目。有关详细信息，请参见 "管理存储库连接"。
- NetApp支持站点提供的软件包

将软件包映像上传到存储库

Astra控制中心引用已连接存储库中的软件包映像和项目。您可以使用命令行界面将映像和项目上传到存储库。

步骤

1. 从NetApp支持站点下载软件包、并将其保存在已安装`kubectl`实用程序的计算机上。
2. 提取压缩的软件包文件、然后将目录更改为Astra Control软件包文件的位置(例如、`Acc.manifest.bundle.YAML`)。
3. 将软件包映像推送到Docker存储库。进行以下替换：
    - 将bundle_file替换为Astra Control捆绑包文件的名称。
    - 将my_regRegistry替换为Docker存储库的URL。
    - 将my_registry_user和my_registry_password替换为存储库的凭据。

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

4. 如果软件包包含项目、请将这些项目复制到项目存储库。将bundle_file替换为Astra Control捆绑包文件的名称、将network_location替换为将项目文件复制到的网络位置：

```
kubectl astra packages copy-artifacts -m BUNDLE_FILE -n NETWORK_LOCATION
```

添加软件包

您可以使用Astra Control Center捆绑包文件导入软件包。这样将安装该软件包并使该软件可供Astra控制中心使用。

使用**Astra Control Web UI**添加软件包

您可以使用Astra控制中心Web UI添加已上传到已连接存储库的软件包。

步骤

1. 在 * 管理帐户 * 导航区域中，选择 * 帐户 * 。
2. 选择*软件包*选项卡。
3. 选择*添加*按钮。
4. 在文件选择对话框中、选择上传图标。
5. 选择一个格式为`.yaml`的Astra Control捆绑包文件进行上传。
6. 选择 * 添加 * 。

结果

如果捆绑包文件有效、并且软件包映像和项目位于已连接的存储库中、则软件包将添加到Astra控制中心。当*状态*列中的状态更改为*可用*时、您可以使用软件包。您可以将鼠标悬停在软件包的状态上以获取详细信息。

> 如果在存储库中未找到某个软件包的一个或多个映像或项目、则会显示该软件包的错误消息。

使用命令行界面添加软件包

您可以使用命令行界面导入已上传到已连接存储库的软件包。为此、您首先需要记录Astra控制中心帐户ID和API令牌。

步骤

1. 使用Web浏览器登录到Astra控制中心Web UI。
2. 从信息板中、选择右上角的用户图标。
3. 选择* API访问*。
4. 记下屏幕顶部附近的帐户ID。
5. 选择*生成API令牌*。

6. 在显示的对话框中、选择*生成API令牌*。
7. 记下生成的令牌、然后选择*关闭*。在命令行界面中、将目录更改为提取的软件包内容中`.yaml`软件包文件的位置。
8. 使用捆绑包文件导入软件包、进行以下替换：
   - 将bundle_file替换为Astra Control捆绑包文件的名称。
   - 将Server替换为Astra Control实例的DNS名称。
   - 将account_ID和token替换为先前记录的帐户ID和API令牌。

```
kubectl astra packages import -m BUNDLE_FILE -u SERVER -a ACCOUNT_ID
-k TOKEN
```

结果

如果捆绑包文件有效、并且软件包映像和项目位于已连接的存储库中、则软件包将添加到Astra控制中心。

> 如果在存储库中未找到某个软件包的一个或多个映像或项目、则会显示该软件包的错误消息。

删除软件包

您可以使用Astra控制中心Web UI删除先前在Astra控制中心导入的软件包。

步骤

1. 在 * 管理帐户 * 导航区域中，选择 * 帐户 * 。
2. 选择*软件包*选项卡。

   您可以在此页面上查看已安装软件包的列表及其状态。

3. 在软件包的*操作*列中、打开操作菜单。
4. 选择 * 删除 * 。

结果

该软件包将从Astra控制中心删除、但该软件包的映像和项目仍保留在存储库中。

了解更多信息

- "管理存储库连接"

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
- 通用 S3
- Microsoft Azure

> 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 ： 当前无法访问此存储分段。

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]
- [轮换或删除存储分段凭据]
- [删除存储分段]

> 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. 从选项菜单的 * 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 * 。

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。

您需要的内容

- 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 * 。

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。您可以监控存储容量和运行状况详细信息，包括当 Astra 控制中心连接到 Cloud Insights 时的性能。

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]
- [取消管理存储后端]
- [更新存储后端许可证]
- [将节点添加到存储后端集群]
- [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。

在存储后端详细信息页面中、对于Astra数据存储、您可以看到以下信息：

- Astra数据存储集群
  - 吞吐量、IOPS和延迟
  - 已用容量与总容量之比
- 用于每个Astra Data Store集群卷
  - 已用容量与总容量之比
  - 吞吐量

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. 查看存储后端部分，其中显示了以下状态：
  - * 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

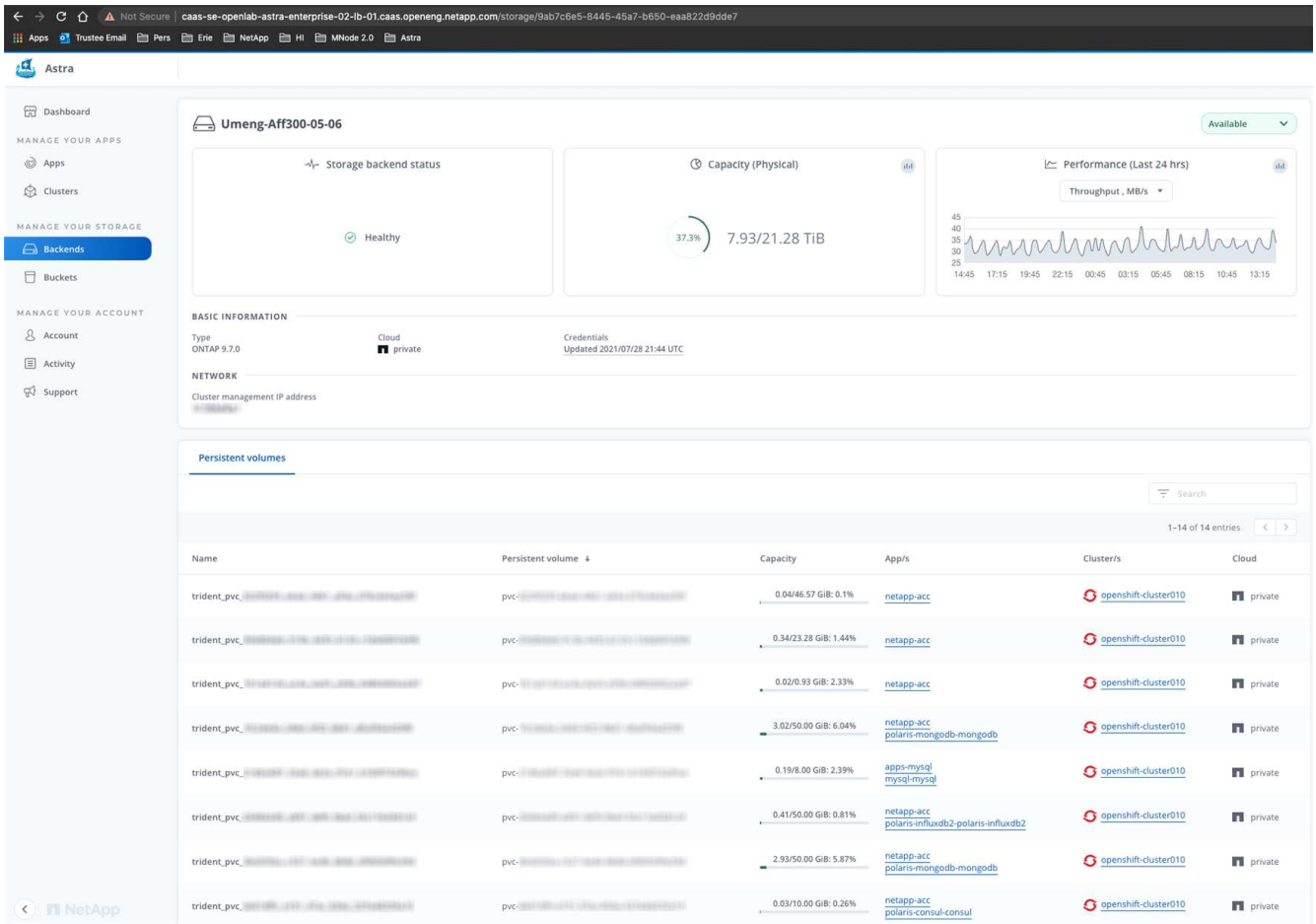
查看有关后端运行状况，容量和性能（ IOPS 吞吐量和 / 或延迟）的信息。

通过连接到 Cloud Insights ，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。

> 如果您连接到 NetApp Cloud Insights ，则 Cloud Insights 中的数据摘录将显示在后端页面上。



3. 要直接转到 Cloud Insights ，请选择指标图像旁边的 * Cloud Insights * 图标。

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。

> 如果要删除 Astra Data Store 后端，则 vCenter 不能创建它。

您需要的内容

- 确保存储后端未受管。
- 确保存储后端没有与 Astra Data Store 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 * 。

更新存储后端许可证

您可以更新 Astra Data Store 存储后端的许可证，以支持更大规模的部署或增强功能。

您需要的内容

- 已部署和管理的 Astra Data Store 存储后端
- Astra Data Store 许可证文件（请联系您的 NetApp 销售代表以购买 Astra Data Store 许可证）

步骤

1. 从左侧导航栏中，选择 * 后端 * 。

2. 选择存储后端的名称。
3. 在*基本信息*下、您可以看到安装的许可证类型。

   如果将鼠标悬停在许可证信息上，则会显示一个弹出窗口，其中包含更多信息，例如到期时间和授权信息。

4. 在 * 许可证 * 下，选择许可证名称旁边的编辑图标。
5. 在*更新许可证*页面中、执行以下操作之一：

许可证状态	**Action**
至少已向Astra数据存储添加一个许可证。	从列表中选择一个许可证。
尚未向Astra数据存储添加任何许可证。	a. 选择*添加*按钮。 b. 选择要上传的许可证文件。 c. 选择*添加*以上传许可证文件。

6. 选择 * 更新 * 。

将节点添加到存储后端集群

您可以向 Astra Data Store 集群添加节点，最多可添加为 Astra Data Store 安装的许可证类型所支持的节点数。

您需要的内容

- 已部署并获得许可的 Astra Data Store 存储后端
- 您已在 Astra 控制中心中添加 Astra 数据存储软件包
- 要添加到集群的一个或多个新节点

步骤

1. 从左侧导航栏中，选择 * 后端 * 。
2. 选择存储后端的名称。
3. 在 " 基本信息 " 下，您可以查看此存储后端集群中的节点数。
4. 在 * 节点 * 下，选择节点数旁边的编辑图标。
5. 在 * 添加节点 * 页面中，输入有关新节点的信息：
   a. 为每个节点分配一个节点标签。
   b. 执行以下操作之一：
      - 如果希望 Astra 数据存储始终根据您的许可证使用最大可用节点数，请启用 * 始终使用最多允许的最大节点数 * 复选框。
      - 如果您不希望 Astra 数据存储始终使用最大可用节点数，请选择所需的要使用的节点总数。
   c. 如果您部署的 Astra 数据存储启用了保护域，请将新节点分配给保护域。
6. 选择 * 下一步 * 。
7. 输入每个新节点的 IP 地址和网络信息。为一个新节点输入一个 IP 地址，为多个新节点输入一个 IP 地址

池。

如果 Astra 数据存储可以使用部署期间配置的 IP 地址，则无需输入任何 IP 地址信息。

8. 选择 * 下一步 * 。
9. 查看新节点的配置。
10. 选择 * 添加节点 * 。

了解更多信息

- "使用 Astra Control API"

监控和保护基础架构

您可以配置多种可选设置来增强您的 Astra 控制中心体验。如果运行 Astra 控制中心的网络需要一个代理来连接到 Internet （将支持包上传到 NetApp 支持站点或建立与 Cloud Insights 的连接），则应在 Astra 控制中心中配置一个代理服务器。要监控和深入了解整个基础架构，请与 NetApp Cloud Insights 建立连接。要从 Astra 控制中心监控的系统收集 Kubernetes 事件，请添加 Fluentd 连接。

添加代理服务器

如果运行 Astra 控制中心的网络需要一个代理来连接到 Internet （将支持包上传到 NetApp 支持站点或建立与 Cloud Insights 的连接），则应在 Astra 控制中心中配置一个代理服务器。

> 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 * 编辑 * 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对其他连接造成中断。

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 Cloud Insights

要监控和深入了解整个基础架构，请将 NetApp Cloud Insights 与您的 Astra 控制中心实例连接起来。Cloud Insights 包含在您的 Astra 控制中心许可证中。

Cloud Insights 应可从 Astra 控制中心使用的网络访问，也可通过代理服务器间接访问。

当 Astra 控制中心连接到 Cloud Insights 时，将创建采集单元 POD 。此 POD 从由 Astra 控制中心管理的存储后端收集数据并将其推送到 Cloud Insights 。此 POD 需要 8 GB RAM 和 2 个 CPU 核。

> 启用 Cloud Insights 连接后，您可以在 * 后端 * 页面上查看吞吐量信息，并在选择存储后端后从此处连接到 Cloud Insights 。您还可以在 "Cluster" 部分的 * 信息板 * 中找到相关信息，并从该处连接到 Cloud Insights 。

您需要的内容

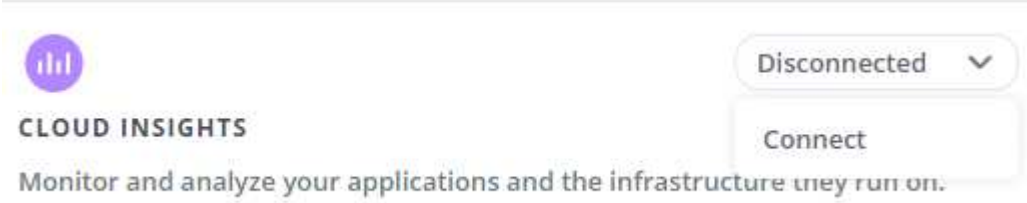
- 具有 * 管理 / 所有者 * 权限的 Astra 控制中心帐户。

- 有效的 Astra Control Center 许可证。
- 如果运行 Astra 控制中心的网络需要使用代理连接到 Internet ，则为代理服务器。

> 如果您是 Cloud Insights 的新用户，请熟悉其特性和功能。请参见 "Cloud Insights 文档"。

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. 选择 * 连接 * ，其中下拉列表中显示 * 已断开连接 * 以添加连接。

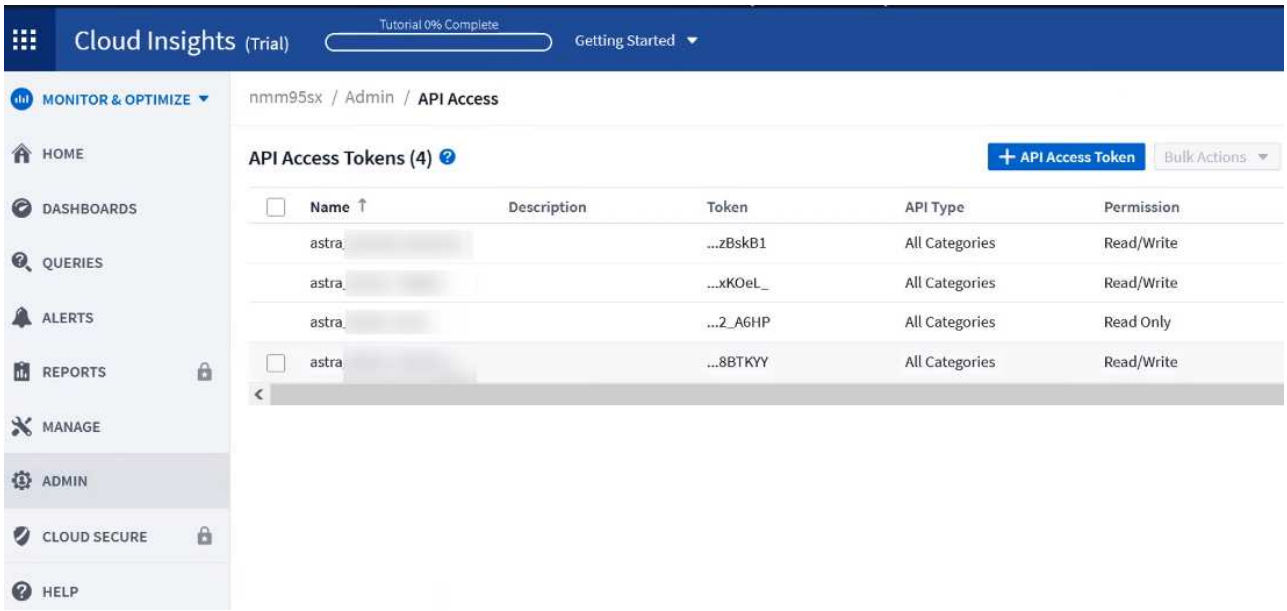


4. 输入 Cloud Insights API 令牌和租户 URL 。例如，租户 URL 采用以下格式：

```
https://<environment-name>.c01.cloudinsights.netapp.com/
```

获取 Cloud Insights 许可证后，您将获得租户 URL 。如果您没有租户 URL ，请参见 "Cloud Insights 文档"。

a. 以获取 "API 令牌"，登录到您的 Cloud Insights 租户 URL 。
b. 在 Cloud Insights 中，单击 * 管理 * > * API 访问 * 以生成 * 读 / 写 * 和 * 只读 * API 访问令牌。



c. 复制 * 只读 * 密钥。您需要将其粘贴到 Astra 控制中心窗口中以启用 Cloud Insights 连接。对于读取 API 访问令牌密钥权限，请选择：资产，警报，采集单元和数据收集。

d. 复制 * 读 / 写 * 密钥。您需要将其粘贴到 Astra 控制中心 * 连接 Cloud Insights * 窗口中。对于读 / 写 API 访问令牌密钥权限，请选择： Assets ， Data Inestion ， Log ingestion ， Acquisition Unit ， 和数据收集。

> 建议您生成 * 只读 * 密钥和 * 读 / 写 * 密钥，不要将同一密钥用于这两种用途。默认情况下，令牌到期期限设置为一年。我们建议您保留默认选择，以便为令牌提供到期前的最长持续时间。如果令牌过期，遥测将停止。

e. 将从 Cloud Insights 复制的密钥粘贴到 Astra 控制中心。

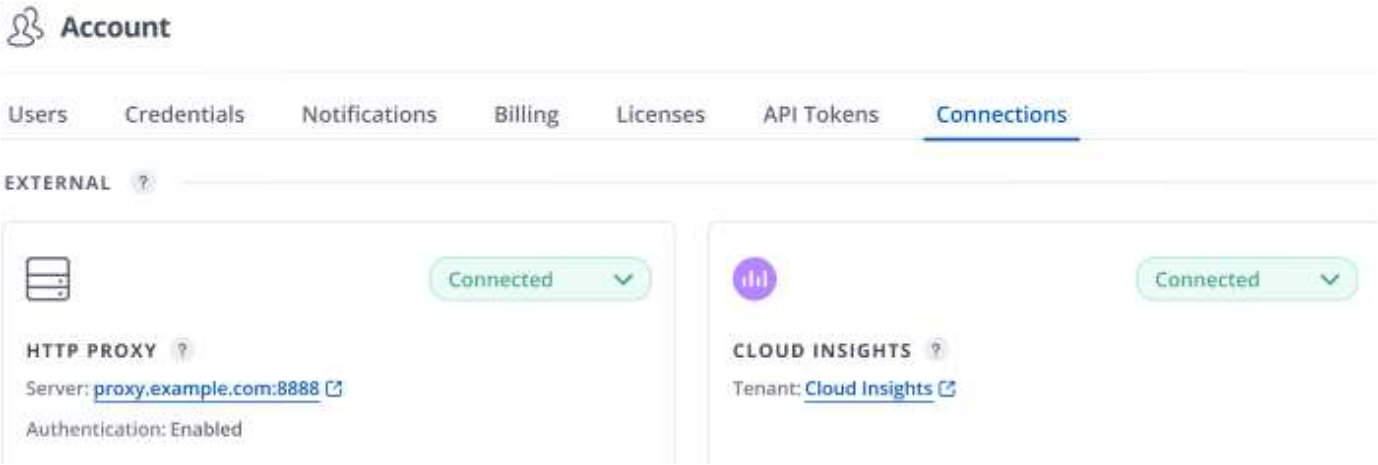
5. 选择 * 连接 * 。

> 选择 * 连接后，在 Cloud Insights * 帐户 * > * 连接 * 页面的 * 连接 * 部分中，连接状态将更改为 * 待定 * 。可以在几分钟内启用连接并将状态更改为 * 已连接 * 。

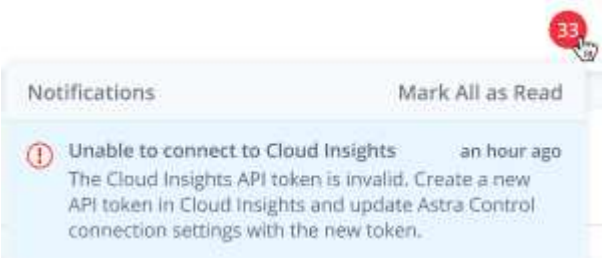
> 要在 Astra 控制中心和 Cloud Insights UI 之间轻松来回切换，请确保您已登录这两个。

在 **Cloud Insights** 中查看数据

如果连接成功，则 * 帐户 * > * 连接 * 页面的 * Cloud Insights * 部分将指示已连接，并显示租户 URL 。您可以访问 Cloud Insights 以查看成功接收和显示的数据。

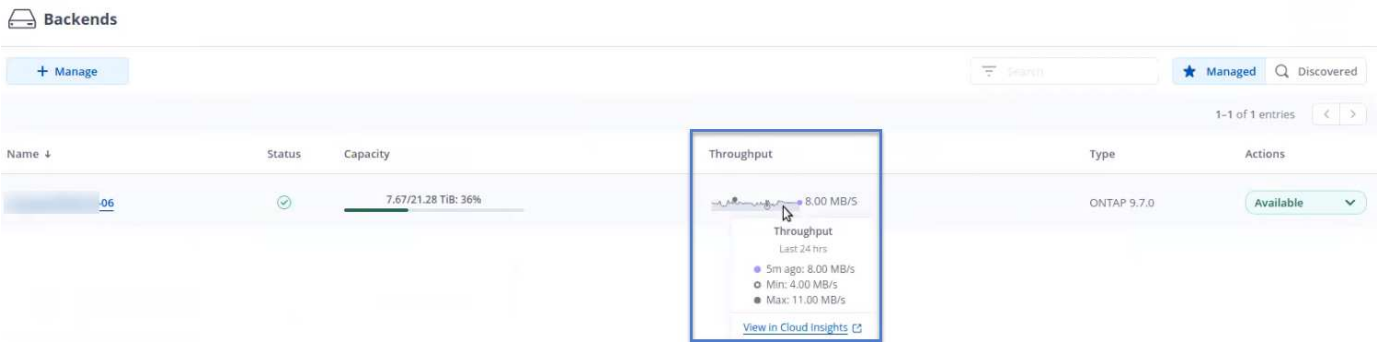


如果连接因某种原因失败，则状态将显示 * 失败 * 。您可以在用户界面右上角的 * 通知 * 下找到失败的原因。



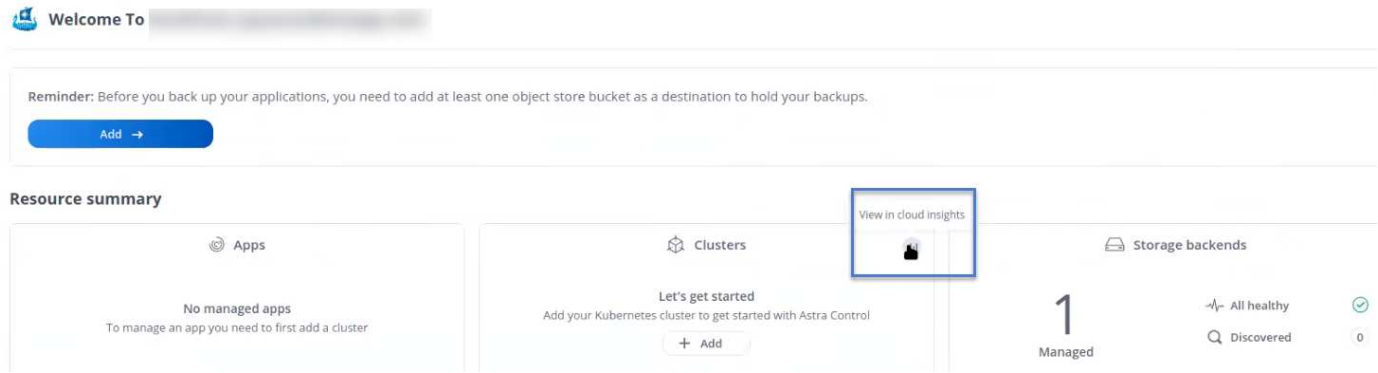
您还可以在 * 帐户 * > * 通知 * 下找到相同的信息。

在 Astra 控制中心中，您可以在 * 后端 * 页面上查看吞吐量信息，并在选择存储后端后从此处连接到 Cloud Insights
。



要直接转到 Cloud Insights ，请选择指标图像旁边的 * Cloud Insights * 图标。

您还可以在 * 信息板 * 上找到相关信息。



> 启用 Cloud Insights 连接后，如果删除在 Astra 控制中心添加的后端，后端将停止向 Cloud Insights 报告。

编辑 **Cloud Insights** 连接

您可以编辑 Cloud Insights 连接。

> 您只能编辑 API 密钥。要更改 Cloud Insights 租户 URL ，我们建议您断开 Cloud Insights 连接并使用新 URL 进行连接。

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 * 编辑 * 以编辑连接。
4. 编辑 Cloud Insights 连接设置。
5. 选择 * 保存 * 。

禁用 **Cloud Insights** 连接

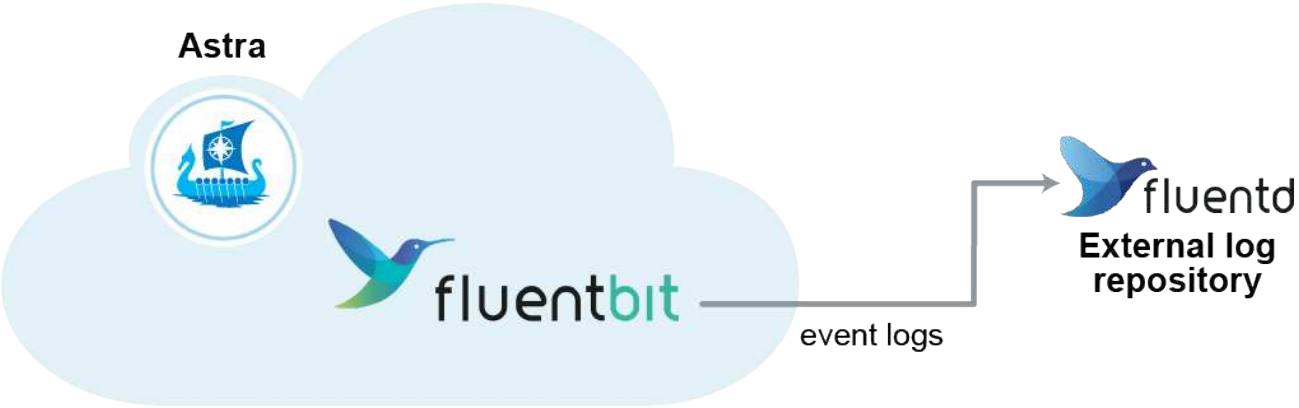
您可以为由 Astra 控制中心管理的 Kubernetes 集群禁用 Cloud Insights 连接。禁用 Cloud Insights 连接不会删除已上传到 Cloud Insights 的遥测数据。

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。确认此操作后，在 * 帐户 * > * 连接 * 页面上， Cloud Insights 状态将更改为 * 待定 * 。要将状态更改为 * 已断开连接 * ，需要几分钟的时间。

连接到 Fluentd

您可以将日志（ Kubernetes 事件）从 Astra 控制中心发送到 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 。

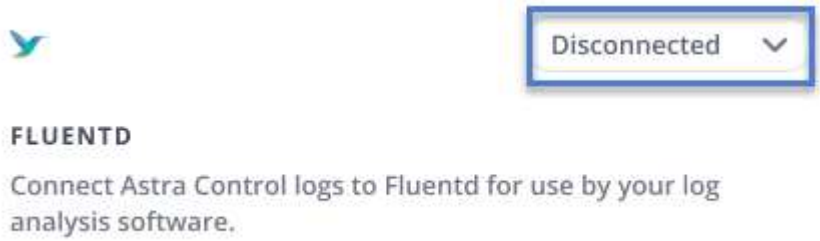
您需要的内容

- 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 * 编辑 * 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。

- 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. 选中不再需要管理的应用程序对应的复选框。
3. 从 * 操作 * 菜单中，选择 * 取消管理 * 。
4. 键入 "unmanage" 进行确认。
5. 确认要取消管理这些应用程序，然后选择 * 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

从 Astra 控制中心取消管理不再需要管理的集群。

- 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。
- 不会从集群中卸载 Trident 。 "了解如何卸载 Trident"。

> 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。

步骤

1. 从左侧导航栏中，选择 * 集群 * 。
2. 选中不再希望在 Astra 控制中心中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，并且不再由 Astra 控制中心管理。

> 如果 Astra 控制中心和 Cloud Insights 未连接 * ，则取消管理集群将删除为发送遥测数据而安装的所有资源。如果已连接 Astra 控制中心和 Cloud Insights * ，则取消管理集群将仅删除 `fluentbit` 和 `event-exporters` Pod 。

升级 Astra 控制中心

要升级 Astra 控制中心，请从 NetApp 支持站点下载安装包，然后按照以下说明升级环境中的 Astra 控制中心组件。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。

您需要的内容

- "开始升级之前，请确保您的环境仍满足 Astra Control Center 部署的最低要求"。
- 确保所有集群操作员均处于运行状况良好且可用。

  OpenShift 示例：

```
oc get clusteroperators
```

- 确保所有 API 服务均处于运行状况良好且可用。

  OpenShift 示例：

```
oc get apiservices
```

- 从 Astra 控制中心注销。

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：

- 下载 Astra Control Center 捆绑包
- [打开软件包的包装并更改目录]
- [将映像添加到本地注册表]
- 安装更新后的 Astra 控制中心操作员
- 升级 Astra 控制中心
- [升级第三方服务（可选）]
- [验证系统状态]
- [设置传入以进行负载平衡]

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

如果计划，备份和快照未运行，请在维护窗口中执行升级。

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

下载 Astra Control Center 捆绑包

1. 从下载 Astra Control Center 升级包（`Astra-control-center-[version].tar.gz`） "NetApp 支持站点"。
2. （可选）使用以下命令验证捆绑包的签名：

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

打开软件包的包装并更改目录

1. 提取映像：

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

2. 更改为 Astra 目录。

```
cd astra-control-center-[version]
```

将映像添加到本地注册表

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

> 有关自动加载映像的信息，请参见下面的示例脚本。

   a. 登录到 Docker 注册表：

```
docker login [your_registry_path]
```

   b. 将映像加载到 Docker 中。
   c. 标记图像。
   d. [substep_image_local_registry_push]] 将映像推送到本地注册表。

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

安装更新后的 **Astra** 控制中心操作员

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

```
vim astra_control_center_operator_deploy.yaml
```

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

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

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

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

d. 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
  template:
    metadata:
      labels:
        control-plane: controller-manager
    spec:
      containers:
      - args:
        - --secure-listen-address=0.0.0.0:8443
        - --upstream=http://127.0.0.1:8080/
        - --logtostderr=true
        - --v=10
        image: [your_registry_path]/kube-rbac-proxy:v4.8.0
        name: kube-rbac-proxy
        ports:
        - containerPort: 8443
          name: https
      - args:
        - --health-probe-bind-address=:8081
        - --metrics-bind-address=127.0.0.1:8080
        - --leader-elect
        command:
        - /manager
        env:
        - name: ACCOP_LOG_LEVEL
          value: "2"
        - name: ACCOP_HELM_UPGRADETIMEOUT
          value: 300m
        image: [your_registry_path]/acc-operator:[version x.y.z]
        imagePullPolicy: IfNotPresent
      imagePullSecrets: []
```

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.astra.
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
```

升级 Astra 控制中心

1. 编辑 Astra 控制中心自定义资源（CR）（`Astra_control_center_min.yaml`），并将 Astra 版本（`AstraVersion` Insidem of `SPec`）编号更改为最新：

```
kubectl edit acc -n [netapp-acc or custom namespace]
```

> 注册表路径必须与中推送映像的注册表路径匹配 上一步。

2. 在 Astra 控制中心 CR 的 `SPec` 内的 `addtionalValues` 中添加以下行：

```
additionalValues:
    nautilus:
      startupProbe:
        periodSeconds: 30
        failureThreshold: 600
```

3. 执行以下操作之一：

   a. 如果您没有自己的 InvorsController 或入口，并且一直使用带有其 Traefik 网关的 Astra 控制中心作为负载平衡器类型的服务，并且希望继续进行此设置，请指定另一个字段 `ingressType` （如果尚未显示）并将其设置为 `AccTraefik` 。

   ```
   ingressType: AccTraefik
   ```

   b. 如果您要切换到默认的 Astra 控制中心通用传入部署，请提供您自己的内部控制器 / 传入设置（采用 TLS 终止等），打开通往 Astra 控制中心的路由，并将 `ingressType` 设置为 `Generic` 。

   ```
   ingressType: Generic
   ```

   > 如果省略此字段，则此过程将成为通用部署。如果您不希望使用通用部署，请务必添加此字段。

4. （可选）验证 Pod 是否终止并重新可用：

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

5. 等待 Astra 状态条件指示升级已完成且准备就绪：

   ```
   kubectl get -o yaml -n [netapp-acc or custom namespace]
   astracontrolcenters.astra.netapp.io astra
   ```

   响应：

   ```
   conditions:
     - lastTransitionTime: "2021-10-25T18:49:26Z"
       message: Astra is deployed
       reason: Complete
       status: "True"
       type: Ready
     - lastTransitionTime: "2021-10-25T18:49:26Z"
       message: Upgrading succeeded.
       reason: Complete
       status: "False"
       type: Upgrading
   ```

6. 重新登录并验证所有受管集群和应用程序是否仍然存在并受到保护。
7. 如果操作员未更新证书管理器，请接下来升级第三方服务。

升级第三方服务（可选）

在先前的升级步骤中，不会升级第三方服务 Traefik 和 Cert-manager 。您可以选择使用此处所述的操作步骤对其进行升级，也可以在系统需要时保留现有服务版本。

- * 任务期限 *： 默认情况下， Astra 控制中心负责管理任务期限部署的生命周期。如果将 `externalTraefik` 设置为 `false` （默认），则表示系统中不存在外部 Traefik ，并且 Astra 控制中心正在安装和管理 Traefik 。在这种情况下， `externalTraefik` 设置为 `false` 。

  另一方面，如果您有自己的 Traefik 部署，请将 `externalTraefik` 设置为 `true` 。在这种情况下，您将保持部署状态，并且 Astra 控制中心不会升级 CRD ，除非 `shouldUpgrade` 设置为 `true` 。

- * 证书管理器 *： 默认情况下， Astra 控制中心会安装证书管理器（和 CRD ），除非您将 `externalCertManager` 设置为 `true` 。将 `shoulldUpgrade` 设置为 `true` 让 Astra Control Center 升级 CRD 。

如果满足以下任一条件，则升级 Traefik：

- externalTraefik： false 或
- externalTraefik： true ， shouldUpgrade： true 。

步骤

1. 编辑 `Acc` CR：

```
kubectl edit acc -n [netapp-acc or custom namespace]
```

2. 根据需要将 `externalTraefik` 字段和 `shouldUpgrade` 字段更改为 `true` 或 `false` 。

```
crds:
    externalTraefik: false
    externalCertManager: false
    shouldUpgrade: false
```

验证系统状态

1. 登录到 Astra 控制中心。
2. 验证所有受管集群和应用程序是否仍存在并受到保护。

设置传入以进行负载平衡

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

- 默认升级使用通用传入部署。在这种情况下，您还需要设置入口控制器或入口资源。
- 如果您不需要入口控制器，但希望保留现有控制器，请将 `ingressType` 设置为 `AccTraefik` 。

有关 "loadbalancer" 服务类型和入口的其他详细信息，请参见 "要求"。

根据您使用的入口控制器类型，步骤会有所不同：

- nginx 入口控制器
- OpenShift 入口控制器

您需要的内容

- 在 CR 规范中，
  - 如果存在 `crd.externalTraefik` ，则应将其设置为 `false` 或
  - 如果 `crd.externalTraefik` 为 `true` ，则 `crd.shouldUpgrade` 也应为 `true` 。
- 所需 "入口控制器" 应已部署。
- 。 "入口类" 应已创建与入口控制器对应的。
- 您使用的是介于 v1.19 和 v1.21 之间的 Kubernetes 版本，包括 v1.19 和 v1.21 。

nginx 入口控制器的步骤

1. 使用现有密钥 `secure-testing-cert` 或创建类型的密钥 "8a637503539b25b68130b6e8003579d9" 用于 `NetApp-Accc` （或自定义命名）命名空间中的 TLS 专用密钥和证书，如中所述 "TLS 密钥"。
2. 在 `NetApp-Accc` （或自定义命名）命名空间中为已弃用或新模式部署入站资源：
   a. 对于已弃用的模式，请遵循以下示例：

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

b. 对于新模式，请遵循以下示例：

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

OpenShift 入口控制器的步骤

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

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

验证入口设置

您可以先验证入口设置，然后再继续操作。

1. 确保已将负载平衡器中的 Traefik 更改为 `clusterIP`：

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

2. 验证 Traefik 中的路由：

```
Kubectl get ingressroute ingressroutetls -n [netapp-acc or custom
namespace]
-o yaml | grep "Host("
```

结果应为空。

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]。

您需要的内容

- 使用 Astra 控制中心 UI 取消全部管理 "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_min.yaml -n netapp-acc
```

结果

```
astracontrolcenter.astra.netapp.io "astra" deleted
```

2. 使用以下命令删除 `NetApp-Accc` 命名空间：

```
kubectl delete ns netapp-acc
```

结果

```
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 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 UI 或 Astra Control API 。

要了解更多信息，请转到 "Astra 自动化文档"。

部署应用程序

从 Helm 图表中部署 Jenkins

了解如何从部署 Jenkins "BitNami Helm 图表"。在集群上部署 Jenkins 后，您可以向 Astra Control 注册此应用程序。

Jenkins 是一款经过验证的适用于 Astra Control 的应用程序。

- "了解Astra Control中经过验证的应用程序与标准应用程序之间的区别"。

这些说明同时适用于 Astra 控制服务和 Astra 控制中心。

> 尚未验证从 Google Marketplace 部署的应用程序。一些用户报告了在 Google Marketplace 部署 Postgres ， MariaDB 和 MySQL 时发现和 / 或备份的问题。

要求

- 已添加到 Astra Control 的集群。

  > 对于 Astra 控制中心，您可以先将集群添加到 Astra 控制中心，或者先添加应用程序。

- 更新了安装在本地计算机上的 Helm （版本 3.2+ ）和 Kubectl 的版本，并为集群提供了正确的 kubeconfig

Astra Control 当前不支持 "适用于 Jenkins 的 Kubernetes 插件"。您可以在不使用插件的 Kubernetes 集群中运行 Jenkins 。该插件可为 Jenkins 集群提供可扩展性。

安装 Jenkins

有关此过程的两个重要注意事项：

- 您必须在将集群添加到 Astra Control Service 后部署应用程序，而不是在之前部署。在将集群添加到 Astra 控制中心之前或之后， Astra 控制中心将接受应用程序。
- 您必须将 Helm 图表部署在非默认命名空间中。

步骤

1. 添加 BitNami 图表 repo ：

```
helm repo add bitnami https://charts.bitnami.com/bitnami
```

2. 使用命令创建 `Jenkins` 命名空间并将 Jenkins 部署到其中：

```
helm install <name> bitnami/jenkins --namespace <namespace> --create
-namespace
--set global.storageClass=<storage_class_name>
```

> 如果卷大小发生更改，请使用 Kibibyte （KI）， mebibyte （Mi）或 Gibibyte （Gi）单位。

只有在以下情况下才需要定义存储类：

- 您正在使用 Astra 控制服务，并且不想使用默认存储类。
- 您正在使用 Astra 控制中心，但尚未将集群导入到 Astra 控制中心。或者，您已导入集群，但不想使用默认存储类。

结果

此操作将执行以下操作：

- 创建命名空间。
- 设置正确的存储类。

Pod 联机后，您可以使用 Astra Control 管理应用程序。使用 Astra Control ，您可以在命名空间级别或使用 Helm 标签管理应用程序。

从 Helm 图表部署 MariaDB

了解如何从部署 MariaDB "BitNami Helm 图表"。在集群上部署 MariaDB 后，您可以使用 Astra Control 管理应用程序。

MariaDB 是一款经过验证的适用于 Astra 的应用程序。

- "了解Astra Control中经过验证的应用程序与标准应用程序之间的区别"。

这些说明同时适用于 Astra 控制服务和 Astra 控制中心。

> 尚未验证从 Google Marketplace 部署的应用程序。一些用户报告了在 Google Marketplace 部署 Postgres， MariaDB 和 MySQL 时发现和 / 或备份的问题。

要求

- 已添加到 Astra Control 的集群。

  > 对于 Astra 控制中心，您可以先将集群添加到 Astra 控制中心，或者先添加应用程序。

- 更新了安装在本地计算机上的 Helm （版本 3.2+ ）和 Kubectl 的版本，并为集群提供了正确的 kubeconfig

安装 MariaDB

有关此过程的两个重要注意事项：

- 您必须在将集群添加到 Astra Control Service 后部署应用程序，而不是在之前部署。在将集群添加到 Astra 控制中心之前或之后， Astra 控制中心将接受应用程序。
- 您必须将 Helm 图表部署在非默认命名空间中。

步骤

1. 添加 BitNami 图表 repo：

```
helm repo add bitnami https://charts.bitnami.com/bitnami
```

2. 使用以下命令部署 MariaDB：

```
helm install <name> bitnami/MariaDB --namespace <namespace> --create
-namespace
--set global.storageClass=<storage_class_name>
```

> 如果卷大小发生更改，请使用 Kibibyte （KI），mebibyte （Mi）或 Gibibyte （Gi）单位。

只有在以下情况下才需要定义存储类：

- 您正在使用 Astra 控制服务，并且不想使用默认存储类。
- 您正在使用 Astra 控制中心，但尚未将集群导入到 Astra 控制中心。或者，您已导入集群，但不想使用默认存储类。

结果

此操作将执行以下操作：

- 创建命名空间。
- 在命名空间上部署 MariaDB 。
- 创建数据库。

> 这种在部署时设置密码的方法不安全。对于生产环境，我们不建议这样做。

Pod 联机后，您可以使用 Astra Control 管理应用程序。使用 Astra Control ，您可以在命名空间级别或使用 Helm 标签管理应用程序。

从 Helm 图表部署 MySQL

了解如何从部署 MySQL "BitNami Helm 图表"。在 Kubernetes 集群上部署 MySQL 后，您可以使用 Astra Control 管理此应用程序。

MySQL 是经过验证的适用于 Astra Control 的应用程序。

- "了解Astra Control中经过验证的应用程序与标准应用程序之间的区别"。

这些说明同时适用于 Astra 控制服务和 Astra 控制中心。

> 尚未验证从 Google Marketplace 部署的应用程序。一些用户报告了在 Google Marketplace 部署 Postgres ， MariaDB 和 MySQL 时发现和 / 或备份的问题。

要求

- 已添加到 Astra Control 的集群。

> 对于 Astra 控制中心，您可以先将集群添加到 Astra 控制中心，或者先添加应用程序。

- 更新了安装在本地计算机上的 Helm （版本 3.2+ ）和 Kubectl 的版本，并为集群提供了正确的 kubeconfig

安装 MySQL

有关此过程的两个重要注意事项：

- 您必须在将集群添加到 Astra Control Service 后部署应用程序，而不是在之前部署。在将集群添加到 Astra 控制中心之前或之后， Astra 控制中心将接受应用程序。
- 建议您将 Helm 图表部署在非默认命名空间中。

步骤

1. 添加 BitNami 图表 repo：

```
helm repo add bitnami https://charts.bitnami.com/bitnami
```

2. 使用以下命令部署 MySQL：

```
helm install <name> bitnami/mysql --namespace <namespace> --create
-namespace
--set global.storageClass=<storage_class_name>
```

> 如果卷大小发生更改，请使用 Kibibyte （ KI ）， mebibyte （ Mi ）或 Gibibyte （ Gi ）单位。

只有在以下情况下才需要定义存储类：

- 您正在使用 Astra 控制服务，并且不想使用默认存储类。
- 您正在使用 Astra 控制中心，但尚未将集群导入到 Astra 控制中心。或者，您已导入集群，但不想使用默认存储类。

结果

此操作将执行以下操作：

- 创建命名空间。
- 在命名空间上部署 MySQL 。

Pod 联机后，您可以使用 Astra Control 管理应用程序。使用 Astra Control ，您可以在命名空间级别或使用 Helm 标签管理应用程序及其名称。

从 Helm 图表部署 Postgres

了解如何从部署 Postgres "BitNami Helm 图表"。在集群上部署 Postgres 后，您可以向 Astra Control 注册该应用程序。

Postgres 是一款经过验证的适用于 Astra 的应用程序。

- "了解Astra Control中经过验证的应用程序与标准应用程序之间的区别"。

这些说明同时适用于 Astra 控制服务和 Astra 控制中心。

> 尚未验证从 Google Marketplace 部署的应用程序。一些用户报告了在 Google Marketplace 部署 Postgres ， MariaDB 和 MySQL 时发现和 / 或备份的问题。

要求

- 已添加到 Astra Control 的集群。

  > 对于 Astra 控制中心，您可以先将集群添加到 Astra 控制中心，或者先添加应用程序。

- 更新了安装在本地计算机上的 Helm （版本 3.2+ ）和 Kubectl 的版本，并为集群提供了正确的 kubeconfig

安装 Postgres

有关此过程的两个重要注意事项：

- 您必须在将集群添加到 Astra Control Service 后部署应用程序，而不是在之前部署。在将集群添加到 Astra 控制中心之前或之后， Astra 控制中心将接受应用程序。
- 您必须将 Helm 图表部署在非默认命名空间中。

步骤

1. 添加 BitNami 图表 repo ：

   ```
   helm repo add bitnami https://charts.bitnami.com/bitnami
   ```

2. 使用以下命令部署 Postgres ：

   ```
   helm install <name> bitnami/postgresql --namespace <namespace> --create
   -namespace
   --set global.storageClass=<storage_class_name>
   ```

   > 如果卷大小发生更改，请使用 Kibibyte （ KI ）， mebibyte （ Mi ）或 Gibibyte （ Gi ）单位。

   只有在以下情况下才需要定义存储类：

- 您正在使用 Astra 控制服务，并且不想使用默认存储类。
- 您正在使用 Astra 控制中心，但尚未将集群导入到 Astra 控制中心。或者，您已导入集群，但不想使用默认存储类。

结果

此操作将执行以下操作：

- 创建命名空间。
- 在命名空间上部署 Postgres 。

Pod 联机后，您可以使用 Astra Control 管理应用程序。使用 Astra Control ，您可以在命名空间级别或使用 Helm 标签管理应用程序。

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。

https://kb.netapp.com/Advice_and_Troubleshooting/Cloud_Services/Astra

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。全天候提供丰富的免费自助支持选项，例如知识库（ KB ）文章和可宽延的渠道。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 Slack 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：

- "* 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。
- * 文档中心 *：这是您当前正在查看的文档站点。
- "* 通过 Slack 获取帮助 *"：转到 Pub 工作空间中的容器通道与同行和专家进行联系。
- * 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 控制中心期间，如果在 Astra 控制中心自定义资源定义（ CRD ）文件（`Astra_control_center_min.yaml` ）中指定 `enrolled` ： `true` for `AutoSupport` ，则每日支持包将自动上传到 "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 21.12 文档"
- "Astra Control Center 21.08 文档"

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"
- "有关Astra数据存储的通知"

Astra Control API 许可证

https://docs.netapp.com/us-en/astra-automation-2204/media/astra-api-license.pdf