



# **Astra Data Store 文档**

## **Astra Data Store**

NetApp  
July 19, 2022

This PDF was generated from <https://docs.netapp.com/zh-cn/astra-data-store/index.html> on July 19, 2022. Always check docs.netapp.com for the latest.

# 目录

Astra Data Store 文档	1
发行说明	2
此版本的 Astra 数据存储中的新增功能	2
已知问题	3
已知限制	4
概念	5
了解 Astra 数据存储	5
Astra Data Store 部署模式	6
集群扩展	7
Astra 数据存储中的存储效率	7
Astra 数据存储中的安全性	8
入门	9
Astra 数据存储要求	9
Astra 数据存储快速入门	13
安装 Astra 数据存储	14
设置Astra数据存储组件	31
Astra数据存储早期访问计划(EAP)版本限制	42
有关Astra数据存储的常见问题	43
使用 Astra 数据存储	46
使用kubectI命令管理Astra Data Store资源	46
部署测试应用程序	52
管理Astra数据存储集群	56
监控Astra数据存储	67
安全的Astra数据存储	80
更新Astra Data Store许可证	85
升级Astra数据存储	86
使用自动化脚本卸载Astra Data Store	88
将Astra数据存储与VMware结合使用	104
了解采用VMware的Astra Data Store	104
采用VMware的Astra Data Store要求	104
使用VMware设置Astra数据存储	105
监控VMware安装的组件	112
管理VMware安装的Astra Data Store组件	114
从VMware集成环境中卸载Astra Data Store	117
知识和支持	118
故障排除	118
获取帮助	118
自动支持监控	118
早期版本的Astra数据存储	123

法律声明 ..... 124

    版权 ..... 124

    商标 ..... 124

    专利 ..... 124

    隐私政策 ..... 124

    开放源代码 ..... 124

# Astra Data Store 文档

# 发行说明

我们很高兴地宣布发布了2022.05.001早期访问计划(EAP)版本的Astra数据存储。

- "此版本的 [Astra 数据存储中的新增功能](#)"
- "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](mailto:doccomments@netapp.com)。

## 此版本的 **Astra** 数据存储中的新增功能

我们很高兴地宣布发布了2022.05.001早期访问计划(EAP)版本的Astra数据存储。

### 2022年6月28日(2022.05.001版"EAP")

适用于Astra Data Store "EAP"版本(22.05)的此修补程序版本(2022.05.001)包括以下改进：

- Astra数据存储和Astra控制中心现在可以在同一集群中运行。
- 从vSphere中删除VVol数据存储库工作流现已可用。



此版本要求更新随kubectl-Astra插件附带的Astra控制中心版本。

### 2022年5月31日(2022.05 "EAP版本")

Astra Data Store 2022.05版本包含以下功能：

- VMware环境集成：
  - NFS VVol数据存储库工作流和基于存储策略的管理存储配置。
  - 适用于VMware vSphere的Astra插件、用于从vCenter管理和监控原生 存储。
- 增强的集群管理功能、包括 "[从Astra控制中心部署基于图形用户界面的集群](#)"。
- 支持 "[更大的集群规模](#)" (节点数量增加、CPU和容量增加)。
- 安全增强功能(密钥和证书以及对外部密钥管理(KMIP)的支持)。

### 2022 年 2 月 8 日 ( 2021.12.1 )

适用于 Astra Data Store 预览版 ( 21.12 ) 的修补版本 ( 2021.12.1 ) 。

- 在此版本中，Astra Data Store preview 支持使用 Calico CNI 的 VXLAN 配置。
- Astra Data Store 预览版现在支持启用了 Calico WireGuard 的网络配置。
- 更新的 AstraDSCluster.YAML adsNetworkInterfaces 一节提供了经过改进的描述性注释。
- 现在，我们提供了新的 Astra Data Store 卸载脚本，作为 kubectl 命令卸载过程的一种更简单的替代方法。

2021 年 12 月 21 日 ( 21.12 )

初始版本的 Astra Data Store 预览版。

- ["它是什么"](#)
- ["部署模式和组件"](#)
- ["入门所需的资源"](#)
- ["安装" 和 "设置"](#)
- ["管理" 和 "监控" 性能](#)
- ["使用 Cloud Insights 进行监控"](#)
- ["获取帮助" 和 "使用自动支持监控"](#)

了解更多信息

- ["已知问题"](#)
- ["已知限制"](#)
- ["NetApp 知识库文章"](#)

## 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。

### Astra Data Store 2202.05早期访问计划(EAP)版本

此版本包含这些已知问题。请仔细查看这些已知问题。

存储后端任务未显示在vSphere中

提交使用现有存储后端或删除存储后端的操作时、此任务不会显示在vSphere客户端的近期任务面板中。

创建VM时会创建额外的文件夹

在vSphere中创建VM时、有时会创建其他文件夹、而不是在数据存储库下为每个VM创建一个文件夹。

适用于VMware vSphere的Astra插件中经过筛选的视图间歇性不显示数据

虚拟机的表列筛选器有时会省略经过筛选的数据。而筛选后的视图则显示正在加载图标。要解决此问题、您可以转到另一个视图并重置筛选器以再次显示数据。

Kubernetes工作节点上的IP地址限制

在以下情况下、创建Astra Data Store集群失败：

1. Kubernetes工作节点的IP地址托管在未分配给Astra数据存储的接口上。
2. 此IP地址与分配给Astra数据存储集群的集群/数据IP地址位于同一子网中。

## 了解更多信息

- ["已知限制"](#)

## 已知限制

已知限制用于确定此产品的早期访问计划(EAP)版本不支持或无法与之正确交互的平台、设备或功能。

### **Astra Data Store 2022.05.001早期访问计划(EAP)版本**

此版本包含这些已知限制。请仔细查看这些已知限制。

#### **VMware集成限制**

不支持以下功能：

- VM工作流：
  - 克隆VM、包括从模板克隆
  - 在先前删除VM磁盘后连接该磁盘
- VVol复制
- 一流磁盘(FCD)
- VMware vMotion
- 链接模式vCenter服务器
- 合规性警报和通知
- 多vCenter和多VASA Provider配置
  - 支持每个vCenter配置一个VASA Provider
- 基于存储策略的管理仅适用于VM级别、在创建VM后无法修改。

无法同时调整多个**VM**磁盘的大小

对于此版本、一次只能调整一个磁盘的大小。如果您尝试一次调整多个磁盘的大小、则调整大小操作仅适用于第一个磁盘。

## 了解更多信息

- ["已知问题"](#)

# 概念

## 了解 Astra 数据存储

Astra 数据存储是 Kubernetes 原生的共享文件软件定义存储（SDS）解决方案，适用于内部数据中心，可帮助客户管理其云原生应用程序。Astra 数据存储为容器和 VM 工作负载提供原生共享文件服务，并提供 NetApp 企业数据管理。

借助 Astra 数据存储，您可以执行以下操作：

- \* 支持 Kubernetes 容器化工作负载 \*：使用您习惯使用的企业级数据管理服务和工具。
- \* 将 Kubernetes 的 "应用程序即服务" 平台用于开发运营 \*：创建弹性的软件定义自助式平台，以提供自动化，可重复的服务，消除 developers 带来的复杂性。

Astra 数据存储库是 Astra 产品系列的一部分。了解 ["Astra 系列"](#)。

## Astra 数据存储功能

Astra Data Store 可通过以下功能为您的云原生 applications 提供端到端 Kubernetes 本机存储和数据管理：

- \* Kubernetes 本机共享文件服务 \*：使用标准 NFS 客户端作为容器和 VM 的统一数据存储库，为 Kubernetes 提供共享文件服务原生。
- \* 云扩展 \*：在同一资源池中提供 Kubernetes 本机多个并行文件系统，以实现类似于云的扩展和利用率，从而无需在集群之外单独管理存储。
- \* API 优先方法 \*：将基础架构作为代码交付用于自动化集群和工作负载 management。
- \* 企业级数据管理 \*：提供自动化应用程序感知型数据保护和灾难恢复：
  - \* NetApp 技术 \*：利用 NetApp 数据管理技术执行快照，备份，复制和克隆，以便用户可以在 Kubernetes 中构建和部署企业级应用程序。
  - \* 故障恢复能力 \*：对 Kubernetes 本机工作负载使用复制和擦除编码技术，以提高故障恢复能力。
  - \* 数据效率 \*：通过实时重复数据删除和数据压缩功能进行扩展时控制成本。
- \* 适合您的现有环境 \*：支持基于微服务的工作负载和传统工作负载，为主要 Kubernetes 分发版提供服务，提供文件存储，并在您选择的 hardware 上运行。
- \* 与 NetApp Cloud Insights \* 集成：为持续 optimization 提供可观察性，分析和监控功能。

## 开始使用 Astra Data Store

首先，["了解 Astra 数据存储的要求"](#)。

然后，["开始使用"](#)。

## 有关详细信息 ...

- ["Astra 系列简介"](#)
- ["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"](#)

## Astra Data Store 部署模式

Astra数据存储使用与Kubernetes部署和协调的应用程序直接管理主机上的存储驱动器。

您可以使用以下选项之一在裸机或虚拟服务器上安装Astra数据存储：

- 部署在一个独立的专用 Kubernetes 集群上，为在单独集群（独立集群）中运行的 Kubernetes 应用程序提供永久性卷。
- 在 Kubernetes 集群上部署，同时在同一节点池（融合集群）上托管其他工作负载应用程序。
- 在 Kubernetes 集群上部署，同时在其他节点池（离散集群）上托管其他工作负载应用程序。

["了解有关Astra Data Store硬件要求的更多信息"](#)。

Astra数据存储库是Astra产品系列的一部分。有关整个Astra系列的视角、请参见 ["Astra 系列简介"](#)。

## Astra Data Store 生态系统

Astra数据存储可与以下各项配合使用：

- **\* Astra 控制中心 \***：使用 Astra 控制中心软件对内部环境中的 Kubernetes 集群进行应用程序感知型数据管理。轻松备份 Kubernetes 应用程序，将数据迁移到其他集群以及即时创建有效的应用程序克隆。

Astra控制中心支持具有ONTAP 的Astra Trident存储后端或Astra数据存储后端的Kubernetes集群。

- **\* Astra Trident \***：作为由 NetApp 维护的完全受支持的开源存储配置程序和编排程序，Astra Trident 使您能够为 Docker 和 Kubernetes 管理的容器化应用程序创建存储卷。

您可以使用Astra Trident在Astra数据存储库上创建卷。

- **\* Cloud Insights \***：Cloud Insights 是一款 NetApp 云基础架构监控工具，可用于监控由 Control 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 Control 中启用 Cloud Insights 连接后，遥测信息将显示在 Astra Control UI 页面中。Cloud Insights 显示有关在Astra数据存储库中管理的资源的信息。

## Astra 数据存储接口

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UI还会显示有关Astra数据存储卷的信息。

- **\* 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。您还可以使用Astra Control API检索有关Astra数据存储的信息。
- **\* kubectl commands**：要使用Astra Data Store、您可以直接使用kubectl命令。
- **\* Kubernetes扩展\***：此外、您还可以使用Astra Data Store Kubernetes API扩展。

自定义资源定义(CRD)是对Kubernetes REST API的扩展、该API是在部署Astra Data Store操作员时创建的。外部实体通过调用 Kubernetes API 服务器与 CRD 进行交互。Astra Data Store会监控特定CRD的更新、然后调用内部REST API。

## 有关详细信息 ...

- ["Astra 系列简介"](#)
- ["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"](#)

## 集群扩展

Astra数据存储支持集群中具有不同类型和功能的节点。如果要扩展集群、则Astra数据存储支持添加具有任何性能功能的节点、只要这些节点不低于集群中性能最低的节点即可。新节点的存储容量必须与现有节点相同。所有节点（包括扩展期间的新节点）至少需要满足中的最低要求 ["Astra 数据存储要求"](#)。

## 有关详细信息 ...

- ["Astra 系列简介"](#)
- ["Astra Control Service 文档"](#)
- ["Astra 控制中心文档"](#)
- ["Astra Trident 文档"](#)
- ["使用 Astra API"](#)
- ["Cloud Insights 文档"](#)
- ["ONTAP 文档"](#)

## Astra 数据存储中的存储效率

Astra数据存储使用基于NetApp ONTAP 和SolidFire 技术的存储效率技术、包括：

- **\* 精简配置 \***：精简配置卷是指不预先预留存储的卷。而是根据需要动态分配存储。删除卷或 LUN 中的数据后，可用空间将释放回存储系统
- **零块检测和消除**：采用精简配置的Astra Data Store存储系统能够检测卷中已置零的区域、以便回收该空间

并将其用于其他位置。

- \* 数据压缩 \*：数据压缩通过将数据块组合到数据压缩组中来减少卷所需的物理存储量，每个数据块都存储为一个块。与传统压缩方法相比、压缩数据的读取速度更快、因为Astra Data Store仅解压缩包含所请求数据的压缩组、而不是解压缩整个文件。
- \* 重复数据删除 \*：重复数据删除可通过丢弃重复块并将其替换为对单个共享块的引用来减少卷（或 AFF 聚合中的所有卷）所需的存储量。读取经过重复数据删除的数据通常不会对性能产生任何影响。除了过载的节点之外，写入所产生的费用可以忽略不计。

默认情况下，所有这些功能均处于启用状态。

请参见 ["存储效率详细信息"](#)。

## 有关详细信息 ...

- ["Astra 系列简介"](#)
- ["Astra Control Service 文档"](#)
- ["Astra 控制中心文档"](#)
- ["Astra Trident 文档"](#)
- ["使用 Astra Control API"](#)
- ["ONTAP 文档"](#)

## Astra 数据存储中的安全性

Astra 数据存储可通过多种方法确保客户端和管理员对存储的访问安全，保护通信和数据以及防范病毒。

Astra数据存储使用以下安全方法：

- 使用相互传输层安全（MTLS）进行通信加密
- 基于角色的访问控制，用于控制对功能的访问
- 部署安全性
- 证书管理
- 空闲软件加密、包括内部和外部密钥管理

## 有关详细信息 ...

- ["Astra 系列简介"](#)
- ["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"](#)

# 入门

## Astra 数据存储要求

首先验证您的环境是否满足Astra Data Store要求。

Astra数据存储既支持裸机部署、也支持基于VM的部署。Astra数据存储集群可以在具有四个或更多工作节点的Kubernetes集群上运行。Astra Data Store软件可以与在同一Kubernetes集群上运行的其他应用程序共存。

- [\[Kubernetes worker node resources\]](#)
- [\[Hardware and software\]](#)
- [\[Networking\]](#)
- [Astra Trident](#)
- [\[Container Network Interfaces\]](#)
- [\[Licensing\]](#)



如果您计划从Astra控制中心管理Astra数据存储集群、请确保您的Astra数据存储集群符合 "[由 Astra 控制中心管理的集群的要求](#)" 除了此处概述的要求之外。

### Kubernetes工作节点资源

以下是在Kubernetes集群中每个工作节点上分配给Astra Data Store软件所需的资源要求：

资源	最小值	最大值
Astra数据存储集群中的节点数	4.	16.
数据驱动器的数量	<ul style="list-style-type: none"><li>• 3 （存在单独的缓存设备）</li><li>• 4 （如果不存在缓存设备）</li></ul>	14
数据驱动器大小	100 GiB	4 TiB
可选缓存设备的数量	1 (8 GiB大小)	不适用

Astra数据存储支持每个Kubernetes工作节点的以下vCPU和RAM组合：

- 9个vCPU、RAM为38 GiB
- 23个vCPU、94 GiB RAM



为了获得最佳写入性能，您应配置一个专用的高持久性，低延迟，低容量缓存设备。

每个工作节点都有以下附加要求：

- 主机磁盘(启动)上要存储Astra Data Store日志文件的可用空间为100 GiB或更大。
- 至少为集群，数据和管理流量提供一个 10GbE 或更快的网络接口。此外，还可以使用一个额外的 1GbE 或更快的接口来隔离管理流量。

## 硬件和软件：

已在以下硬件平台、软件和存储配置上验证Astra Data Store软件。请访问 ["NetApp 社区支持"](#) 如果您的 Kubernetes 集群配置不同。

### 硬件平台

- Dell R640
- Dell R740
- HPE DL360
- HPE DL380
- 联想SR630
- 联想SR650



基于VM的部署还可以使用上列为兼容的服务器 ["VMware 兼容性指南"](#)。

### 存储

Astra数据存储已通过SATA、SAS和NVMe TLC SSD的验证。

对于基于VM的部署、您可以使用以虚拟磁盘或直通驱动器形式提供的驱动器。



- 如果主机在硬件 RAID 控制器后使用 SSD ， 请将硬件 RAID 控制器配置为使用 " 直通 " 模式。
- 每个驱动器都应具有唯一的序列号。在虚拟机创建期间，在虚拟机高级设置中添加属性 `disk.enableid=true` 。

### 软件

- 虚拟机管理程序：Astra Data Store已通过使用vSphere 7.0部署基于VMware的虚拟机的验证。Astra数据存储不支持基于KVM的部署。
- 已在以下主机操作系统上验证Astra数据存储：
  - Red Hat Enterprise Linux 8.4
  - Red Hat Enterprise Linux 8.2
  - Red Hat Enterprise Linux 7.9
  - Red Hat Enterprise Linux CoreOS （ RHCOS ）
  - CentOS 8
  - Ubuntu 20.04
- Astra数据存储已通过以下Kubernetes分发版的验证：
  - Red Hat OpenShift 4.6到4.9
  - Google Anthos 1.8到1.10
  - Kubernetes 1.20至1.23



Astra数据存储需要使用Astra Trident进行存储配置和编排。支持Astra Trident 21.10.1至22.04版。请参见 "[Astra Trident 安装说明](#)"。

## 网络

Astra数据存储要求每个集群为MVIP配置一个IP地址。它必须是与 MIP 位于同一子网中的未使用或未配置的 IP 地址。Astra数据存储管理接口应与Kubernetes节点的管理接口相同。

此外，还可以按下表所述配置每个节点：



此表使用了以下缩写词： mip ： management IP address cip ： cluster ip address MVIP ： management virtual IP address

Configuration	所需的 IP 地址
每个节点一个网络接口	<ul style="list-style-type: none"><li>• 每个节点两（2）个：<ul style="list-style-type: none"><li>◦ MIP/CIP：每个节点的管理接口上有一（1）个预配置的 IP 地址</li><li>◦ Data IP：在与 MIP 相同的子网中，每个节点一（1）个未使用或未配置的 IP 地址</li></ul></li></ul>
每个节点两个网络接口	<ul style="list-style-type: none"><li>• 每个节点三个：<ul style="list-style-type: none"><li>◦ MIP：每个节点的管理接口上有一（1）个预配置的 IP 地址</li><li>◦ CIP：在与 MIP 不同的子网中，每个节点的数据接口上有一（1）个预先配置的 IP 地址</li><li>◦ Data IP：在与 CIP 相同的子网中，每个节点一（1）个未使用或未配置的 IP 地址</li></ul></li></ul>



这些配置中不使用任何 VLAN 标记。

## 防火墙注意事项

在强制实施网络防火墙流量筛选的环境中、必须将防火墙配置为允许传入下表中列出的端口和协议的流量。"IP Address"列使用以下缩写：

- mip：每个节点的管理接口上的主IP地址
- cip：每个节点的集群接口上的主IP地址
- DIP：在节点上配置的一个或多个数据IP地址
- MVIP：在一个集群节点上配置的管理虚拟IP地址

端口/协议	IP 地址	目的	注释:
111 TCP	DIP	NFS	数据IP在运行时在集群节点之间移动。每个节点应允许所有数据IP (或整个子网)使用此流量。
442/TCP	MIP	API	
635 TCP	DIP	NFS	数据IP在运行时在集群节点之间移动。每个节点应允许所有数据IP (或整个子网)使用此流量。
2010/UDP	CIP	集群/节点发现	包括与UDP端口2010之间的单播和广播流量、包括回复。
2049 TCP	DIP	NFS	数据IP在运行时在集群节点之间移动。每个节点应允许所有数据IP (或整个子网)使用此流量。
2181-2183/TCP	CIP	分布式通信	
2443/TCP	MIP	API	
2443/TCP	MVIP	API	MVIP地址可由任何集群节点托管、并可在运行时根据需要重新定位。
4000-4006/TCP	CIP	集群内RPC	
4045-4046/ TCP	DIP	NFS	数据IP在运行时在集群节点之间移动。每个节点应允许所有数据IP (或整个子网)使用此流量。
7700/TCP	CIP	会话管理器	
9919/TCP	MIP	DMS-API	
9920/TCP	DIP	DMS-REST服务器	
ICMP	CIP + DIP	节点内通信、运行状况检查	数据IP在运行时在集群节点之间移动。每个节点应允许所有数据IP (或整个子网)使用此流量。

## Astra Trident

Astra数据存储要求应用程序Kubernetes集群运行Astra Trident进行存储配置和编排。支持Astra Trident 21.10.1至22.04版。Astra数据存储可配置为 ["存储后端"](#) 使用 Astra Trident 配置永久性卷。

## 容器网络接口

Astra数据存储已通过以下容器网络接口(CNI)的验证。

- 适用于RKE集群的Calico

- 适用于 Vanilla Kubernetes 集群的 Calico 和 Weave Net CNI
- 适用于 Red Hat OpenShift 容器平台（ OCP ）的 OpenShift SDN
- Cilium for Google Anthos



- 随Cilium CNI部署的Astra数据存储需要portmap插件才能支持hostPort。您可以通过将`cnis-canding-mode: portmap`添加到cilial-config configMap并重新启动Cilium Pod来启用CNI链模式。
- 只有Calico和OpenShift SDN CNI才支持启用了防火墙的配置。

## 许可

要启用全部功能、Astra数据存储需要有效的许可证。

["请在此处注册"](#) 以获取Astra Data Store许可证。注册后，系统将向您发送许可证下载说明。

## 下一步行动

查看 ["快速入门"](#) 概述。

有关详细信息 ...

["Astra数据存储限制"](#)

# Astra 数据存储快速入门

此页面简要概述了开始使用Astra数据存储所需的步骤。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！如果要试用Astra数据存储、可以使用90天评估许可证。 ["请在此处注册"](#) 获取Astra Data Store许可证。

跨度 class="image">&lt;img src="<a href="https://raw.githubusercontent.com/NetAppDocs/common/main/media/number-1.png"" class="bare">https://raw.githubusercontent.com/NetAppDocs/common/main/media/number-1.png"</a> Alt-one">&lt;span>&lt;查看 Kubernetes 集群要求

- 集群必须以运行状况良好的状态运行，并且至少具有四个或更多工作节点。
- 属于Astra Data Store部署的每个Kubernetes工作节点都应具有相同接口类型(SATA、 SAS或NVMe)的 SSD、并具有相同数量的驱动器、这些驱动器将分配给Astra Data Store集群。
- 每个 SSD 都应具有唯一的序列号。

了解更多信息 ["Astra 数据存储要求"](#)。

跨度 class="image">&lt;img src="<a href="https://raw.githubusercontent.com/NetAppDocs/common/main/media/number-2.png"" class="bare">https://raw.githubusercontent.com/NetAppDocs/common/main/media/number-2.png"</a> Alt-twe">&lt;span>&lt;下载并安装Astra数据存储

- 从下载Astra数据存储 ["NetApp 支持站点"](#)。
- 在本地环境中安装Astra数据存储。



- 应用Astra Data Store许可证。
- 安装Astra数据存储集群。

了解更多信息 ["安装Astra数据存储"](#)。

跨度 `class="image">https://raw.githubusercontent.com/NetAppDocs/common/main/media/number-3.png"</a> Alt-Three </span>`; 完成一些初始设置任务

- 安装 Astra Trident 。
- 安装 Kubernetes Snapshot 自定义资源定义（CRD）和控制器。
- 将Astra数据存储设置为存储后端。
- 创建默认的Astra Data Store存储类。
- 为遥测服务配置Astra数据存储。

详细了解 ["初始设置过程"](#)。

完成Astra数据存储的设置后、您接下来可能会执行以下操作：

- 使用 kubectl 命令和 kubectl astrad 扩展来管理集群，包括将节点置于维护模式，更换驱动器或更换节点等任务。了解更多信息 ["如何对Astra数据存储使用kubectl命令"](#)。
- 配置监控端点。了解更多信息 ["配置监控端点"](#)。
- ["将Astra数据存储与VMware结合使用"](#)。

["安装 Astra 数据存储"](#)。

## 安装 Astra 数据存储

您可以使用Kubernetes本机命令或使用Astra控制中心中的UI安装Astra数据存储。

### 安装选项

- 使用**Kubernetes**本机命令：要使用Kubernetes本机命令安装Astra Data Store、请完成中所述的安装步骤 [此操作步骤](#)。
- 使用**Astra**控制中心：要使用Astra控制中心安装Astra数据存储库、请完成中所述的安装步骤 [此操作步骤](#)。

### 您将需要什么

- ["开始安装之前、请为部署Astra Data Store准备您的环境"](#)。
- 访问 ["NetApp 支持站点"](#)。 ["注册"](#) 如果您还没有完全访问的NetApp支持站点帐户、请下载评估版。
- 答 ["NetApp 许可证文件（NLF）"](#) 用于Astra数据存储。下载许可证的说明将在您之后发送给您 ["注册"](#)。
- 具有活动上下文集群管理员权限的活动 kubeconfig 。
- 了解 ["角色和权限"](#) 由Astra数据存储使用。

## 安装 Astra 数据存储

标准Kubernetes集群的Astra数据存储安装过程将指导您完成以下高级步骤。此外、还介绍了适用于Red Hat OpenShift容器平台(OCP)环境的其他安装步骤。

- [\[Download the Astra Data Store bundle and extract the images\]](#)
- [\[Copy the binary and push images to your local registry\]](#)
- [\[OpenShift procedure\]](#)
- [\[Install the Astra Data Store operator\]](#)
- [\[Deploy the Astra Data Store version YAML\]](#)
- [\[Apply the Astra Data Store license\]](#)
- [\[Install the Astra Data Store cluster\]](#)
- [\[Understand deployment-related events\]](#)



要将Astra Data Store与Google Anthos结合使用、请完成以下安装步骤、然后将Astra Data Store集群添加到您的Anthos环境中。



要在Rancher RKE环境中安装Astra数据存储、请完成以下安装步骤、并将Rancher命令替换为kubectl命令。

### 下载Astra Data Store软件包并提取映像

1. 登录到 ["NetApp 支持站点"](#) 并下载Astra数据存储捆绑包(Astra\_Data\_Store\_2022.05.01.tar)。



如果要查找早期版本的捆绑包的说明、请参见 ["该版本的文档"](#)。

2. (可选) 使用以下命令验证捆绑包的签名：

```
openssl dgst -sha256 -verify Astra_Data_Store_2022.05.01.pub -signature  
Astra_Data_Store_2022.05.01.sig 2022.12.01_ads.tar
```

3. 创建目录：

```
mkdir Astra_Data_Store_2022.05.01  
cd Astra_Data_Store_2022.05.01
```

4. 提取映像：

```
tar -xvf <path to tar file>/Astra_Data_Store_2022.05.01.tar
```



这些图像将提取到工作目录中创建的`astrads/images`目录中。

## 将二进制文件和推送映像复制到本地注册表

1. 将kubectl-astrad二进制文件从用于提取映像的目录复制到安装了K8s kubectl二进制文件的标准路径(在以下示例中使用`/usr/bin/`作为路径)。kubectl-astrad是一个自定义kubectl扩展、用于安装和管理Astra数据存储集群。



使用`which kubectl`命令查找安装kubectl二进制文件的路径。

```
cp -p ./astrads/bin/kubectl-astrads /usr/bin/.
```

2. 将Astra Data Store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 login [your_registry_path]
```

- b. 将环境变量设置为要推送Astra数据存储映像的注册表路径；例如、repo.company.com。

```
export REGISTRY=repo.company.com/astrads
```

- c. 运行脚本将映像加载到 Docker ， 标记映像，并将这些映像推送到本地注册表：

```
for astraImageFile in $(ls astrads/images/*.tar) ; do
    astraImage=$(docker load --input ${astraImageFile} | sed 's~Loaded
image: ~~')
    astraImageShort=`echo $astraImage | sed 's~.*/~~'`
    docker tag ${astraImage} ${REGISTRY}/${astraImageShort}
    docker push ${REGISTRY}/${astraImageShort}
done
sed -i 's~\[YOUR_REGISTRY\]~'\${REGISTRY}'~'
./astrads/manifests/*.yaml
```

## OpenShift 操作步骤

以下操作步骤 仅用于在Red Hat OpenShift容器平台(OCP)上部署。跳过此操作步骤 以在非OCP Kubernetes集群上部署。

- 创建命名空间以部署Astra数据存储
- 创建自定义 SCC
- 创建角色和角色绑定

## 示例 1. 详细信息

创建一个`astrads-system`命名空间、其中将安装所有Astra Data Store组件。

只有在Red Hat OpenShift容器平台(OCP)上部署时、才需要执行以下步骤。

### 1. 创建命名空间：

```
kubectl create -f ads_namespace.yaml
```

示例： ads\_namespace.yaml

```
apiVersion: v1
kind: Namespace
metadata:
  labels:
    control-plane: operator
  name: astrads-system
```

OpenShift 使用安全上下文约束（ SCC ）来控制 Pod 可以执行的操作。默认情况下，任何容器的执行都将获得受限的 SCC ，并且仅获得该 SCC 定义的功能。

受限SCC不提供Astra Data Store集群Pod所需的权限。使用此操作步骤 为Astra数据存储提供所需的权限(在示例中列出)。

将自定义SCC分配给Astra Data Store命名空间的默认服务帐户。

只有在 Red Hat OpenShift 容器平台（ OCP ） 上部署时，才需要执行以下步骤。

### 1. 创建自定义 SCC：

```
kubectl create -f ads_privileged_scc.yaml
```

示例： ads\_privileged\_scc.yaml

```

allowHostDirVolumePlugin: true
allowHostIPC: true
allowHostNetwork: true
allowHostPID: true
allowHostPorts: true
allowPrivilegeEscalation: true
allowPrivilegedContainer: true
allowedCapabilities:
- '*'
allowedUnsafeSysctls:
- '*'
apiVersion: security.openshift.io/v1
defaultAddCapabilities: null
fsGroup:
  type: RunAsAny
groups: []
kind: SecurityContextConstraints
metadata:
  annotations:
    kubernetes.io/description: 'ADS privileged. Grant with caution.'
    release.openshift.io/create-only: "true"
  name: ads-privileged
priority: null
readOnlyRootFilesystem: false
requiredDropCapabilities: null
runAsUser:
  type: RunAsAny
seLinuxContext:
  type: RunAsAny
seccompProfiles:
- '*'
supplementalGroups:
  type: RunAsAny
users:
- system:serviceaccount:astrads-system:default
volumes:
- '*'

```

## 2. 使用 `oc get SCC` 命令显示新添加的 SCC :

```
# oc get scc/ads-privileged
NAME          PRIV  CAPS  SELINUX  RUNASUSER  FSGROUP  SUPGROUP
PRIORITY      READONLYROOTFS  VOLUMES
ads-privileged true  ["*"] RunAsAny RunAsAny  RunAsAny  RunAsAny
<no value> false                ["*"]
#
```

创建Astra数据存储的默认服务帐户要使用的所需角色和角色绑定。

以下YAML定义可分配`astrads.netapp.io` API组中的Astra Data Store资源所需的各种角色(通过绑定)。

只有在 Red Hat OpenShift 容器平台（ OCP ） 上部署时，才需要执行以下步骤。

#### 1. 创建定义的角色和角色绑定：

```
kubectl create -f oc_role_bindings.yaml
```

示例： oc\_role\_Bindings.yaml

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: privcrole
rules:
- apiGroups:
  - security.openshift.io
  resourceNames:
  - ads-privileged
  resources:
  - securitycontextconstraints
  verbs:
  - use
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: default-scc-rolebinding
  namespace: astrads-system
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: privcrole
subjects:
- kind: ServiceAccount
```

```

    name: default
    namespace: astrads-system
---

apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: ownerref
  namespace: astrads-system
rules:
- apiGroups:
  - astrads.netapp.io
  resources:
  - '*/finalizers'
  verbs:
  - update
---

apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: or-rb
  namespace: astrads-system
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: ownerref
subjects:
- kind: ServiceAccount
  name: default
  namespace: astrads-system

```

## 配置私有映像注册表

对于某些环境、您可以将配置更改为从使用机密的专用注册表中提取映像。

1. 创建`astrads-system`命名空间、除非您已在上一步中创建了此命名空间：

```
kubectl create namespace astrads-system
```

2. 创建密钥：

```
kubectl create secret docker-registry <secret-name> -n astrads-system
--docker-server=<registry name> --docker-username= <registry username>
--docker-password=<registry user password>
```

### 3. 向服务帐户添加机密配置信息：

```
kubectl patch serviceaccount default -p '{"imagePullSecrets": [{"name":
"<secret-name>"}]}' -n astrads-system
```



这些更改将在您使用时应用 [安装Astra数据存储操作员](#)。

## 安装Astra数据存储操作员

### 1. 列出Astra数据存储清单：

```
ls astrads/manifests/*.yaml
```

响应：

```
astrads/manifests/monitoring_operator.yaml
astrads/manifests/astradscluster.yaml
astrads/manifests/astradsversion.yaml
astrads/manifests/astradsoperator.yaml
astrads/manifests/vasa_asup_certs.yaml
astrads/manifests/manifest.yaml
astrads/manifests/configuration.yaml
```

### 2. 使用`kubectl apply`部署操作员：

```
kubectl apply -f ./astrads/manifests/astradsoperator.yaml
```

响应：



根据您执行的是标准安装还是、命名空间响应可能会有所不同 ["OpenShift容器平台安装"](#)。

```
namespace/astrads-system created
customresourcedefinition.apiextensions.k8s.io/astradsaddrives.astrads.n
etapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsautosupports.astrad
s.netapp.io created
```



```
customresourcedefinition.apiextensions.k8s.io/astradscLOUDsnapshots.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsclusters.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsexportpolicies.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsfaileddrives.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradskeyproviders.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradslICENSES.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsnfsOPTIONS.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsnodeinfoes.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsnodemANAGEMENTS.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsSQSPolicies.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradssearkeyrotaterequests.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsversions.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsvolumefiles.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsvolumes.astrads.netapp.io created
customresourcedefinition.apiextensions.k8s.io/astradsvolumesnapshots.astrads.netapp.io created
role.rbac.authorization.k8s.io/astrads-astrads-system-admin-role created
role.rbac.authorization.k8s.io/astrads-astrads-system-reader-role created
role.rbac.authorization.k8s.io/astrads-astrads-system-writer-role created
role.rbac.authorization.k8s.io/astrads-leader-election-role created
role.rbac.authorization.k8s.io/astrads-manager-role created
clusterrole.rbac.authorization.k8s.io/astrads-astrads-admin-clusterrole created
clusterrole.rbac.authorization.k8s.io/astrads-astrads-reader-clusterrole created
clusterrole.rbac.authorization.k8s.io/astrads-astrads-writer-clusterrole created
clusterrole.rbac.authorization.k8s.io/astrads-astradsautosupport-editor-role created
clusterrole.rbac.authorization.k8s.io/astrads-astradsautosupport-viewer-
```

```
role created
clusterrole.rbac.authorization.k8s.io/astrads-astradscloudsnapshot-
editor-role created
clusterrole.rbac.authorization.k8s.io/astrads-astradscloudsnapshot-
viewer-role created
clusterrole.rbac.authorization.k8s.io/astrads-astradscluster-editor-role
created
clusterrole.rbac.authorization.k8s.io/astrads-astradscluster-viewer-role
created
clusterrole.rbac.authorization.k8s.io/astrads-astradsexportpolicy-
editor-role created
clusterrole.rbac.authorization.k8s.io/astrads-astradsexportpolicy-
viewer-role created
clusterrole.rbac.authorization.k8s.io/astrads-astradsfaileddrive-editor-
role created
clusterrole.rbac.authorization.k8s.io/astrads-astradsfaileddrive-viewer-
role created
clusterrole.rbac.authorization.k8s.io/astrads-astradslicense-editor-role
created
clusterrole.rbac.authorization.k8s.io/astrads-astradslicense-viewer-role
created
clusterrole.rbac.authorization.k8s.io/astrads-astradsnfsoption-editor-
role created
clusterrole.rbac.authorization.k8s.io/astrads-astradsnfsoption-viewer-
role created
clusterrole.rbac.authorization.k8s.io/astrads-astradsnodeinfo-editor-
role created
clusterrole.rbac.authorization.k8s.io/astrads-astradsnodeinfo-viewer-
role created
clusterrole.rbac.authorization.k8s.io/astrads-astradsnodemangement-
editor-role created
clusterrole.rbac.authorization.k8s.io/astrads-astradsnodemangement-
viewer-role created
clusterrole.rbac.authorization.k8s.io/astrads-astradsqospolicy-viewer-
role created
clusterrole.rbac.authorization.k8s.io/astrads-astradsversion-editor-role
created
clusterrole.rbac.authorization.k8s.io/astrads-astradsversion-viewer-role
created
clusterrole.rbac.authorization.k8s.io/astrads-astradsvolume-editor-role
created
clusterrole.rbac.authorization.k8s.io/astrads-astradsvolume-viewer-role
created
clusterrole.rbac.authorization.k8s.io/astrads-astradsvolumeeditor-
role created
clusterrole.rbac.authorization.k8s.io/astrads-astradsvolumeeditor-
viewer-
```

```

role created
clusterrole.rbac.authorization.k8s.io/astrads-astradsvolumesnapshot-
editor-role created
clusterrole.rbac.authorization.k8s.io/astrads-astradsvolumesnapshot-
viewer-role created
clusterrole.rbac.authorization.k8s.io/astrads-manager-role created
rolebinding.rbac.authorization.k8s.io/astrads-astrads-admin-rolebinding
created
rolebinding.rbac.authorization.k8s.io/astrads-astrads-reader-rolebinding
created
rolebinding.rbac.authorization.k8s.io/astrads-astrads-writer-rolebinding
created
rolebinding.rbac.authorization.k8s.io/astrads-leader-election-
rolebinding created
rolebinding.rbac.authorization.k8s.io/astrads-manager-rolebinding
created
clusterrolebinding.rbac.authorization.k8s.io/astrads-astrads-admin-
rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/astrads-astrads-reader-
rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/astrads-astrads-writer-
rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/astrads-manager-rolebinding
created
configmap/astrads-autosupport-cm created
configmap/astrads-firetap-cm created
configmap/astrads-kevents-asup created
configmap/astrads-metrics-cm created
secret/astrads-autosupport-certs created
secret/astrads-webhook-server-cert created
service/astrads-webhook-service created
deployment.apps/astrads-operator created

```

### 3. 验证 Astra 数据存储操作员 POD 是否已启动且正在运行：

```
kubectl get pods -n astrads-system
```

响应：

NAME	READY	STATUS	RESTARTS	AGE
astrads-operator-5ffb94fbf-7ln4h	1/1	Running	0	17m

## 部署Astra Data Store版本YAML

1. 使用`kubectl apply`进行部署：

```
kubectl apply -f ./astrads/manifests/astradsversion.yaml
```

2. 验证 Pod 是否正在运行：

```
kubectl get pods -n astrads-system
```

响应：

NAME	READY	STATUS	RESTARTS
AGE			
astrads-cluster-controller-7f6f884645-xxf2n	1/1	Running	0
117s			
astrads-ds-nodeinfo-2jqnk	1/1	Running	0
2m7s			
astrads-ds-nodeinfo-dbk7v	1/1	Running	0
2m7s			
astrads-ds-nodeinfo-rn9tt	1/1	Running	0
2m7s			
astrads-ds-nodeinfo-vsmhv	1/1	Running	0
2m7s			
astrads-license-controller-fb8fd56bc-bxq7j	1/1	Running	0
2m2s			
astrads-operator-5ffb94fbf-7ln4h	1/1	Running	0
2m10s			

## 应用Astra Data Store许可证

1. 应用从NetApp获得的NetApp许可证文件(NLF)。运行命令之前，请输入您所在集群的名称（`<Astra-Data-Store-cluster-name>`） [即将部署](#) 或已部署许可证文件的路径（`<file\_path/file.txt>`）：

```
kubectl astrads license add --license-file-path <file_path/file.txt>  
--ads-cluster-name <Astra-Data-Store-cluster-name> -n astrads-system
```

2. 验证是否已添加此许可证：

```
kubectl astrads license list
```

响应：

NAME	ADSCUSTER	VALID	PRODUCT
EVALUATION ENDDATE	VALIDATED		
e100000006-ads-capacity	astrads-example-cluster	true	Astra Data Store
true	2023-01-23	2022-04-04T14:38:54Z	

## 安装Astra数据存储集群

### 1. 打开 YAML 文件：

```
vim ./astrads/manifests/astradscluster.yaml
```

### 2. 编辑 YAML 文件中的以下值。



以下步骤将提供一个简化的 YAML 文件示例。

- (必需) \* 元数据 \*：在 metadata 中，将 name string 更改为集群名称。此集群名称必须与您在使用时使用的集群名称相同 [应用许可证](#)。
- (必需) \* 规格 \*：在 sPec 中更改以下必需值：
  - 根据您的许可证和Astra Data Store安装大小、将`adsNodeConfig`值更改为您的安装所需的值：
    - 小型：9个CPU和38个内存
    - 中：23个CPU和94个内存
  - (可选)删除围绕`adsNodeSelector`部分的注释。如果要将Astra数据存储限制为仅安装在选定的工作节点池上、请配置此设置。
  - (可选)指定在4-16之间应在Astra数据存储集群中使用的特定节点数。
  - 将 mVIP 字符串更改为可从集群中的任何工作节点路由的浮动管理 IP 的 IP 地址。
  - 在 adsDataNetworks 中，添加一个逗号分隔的浮动 IP 地址列表 (addresses)，这些地址可从要挂载 NetApp 卷的任何主机路由。每个节点使用一个浮动 IP 地址。数据网络IP地址的数量应至少与Astra数据存储节点的数量相同。对于Astra数据存储、这意味着至少需要4个地址、如果您计划稍后扩展集群、则最多需要16个地址。
  - 在 adsDataNetworks 中，指定数据网络使用的网络掩码。
  - 在 adsNetworkInterfaces 中，将`<mgmt\_interface\_name>`和`<cluster\_and\_storage\_interface\_name>`值替换为要用于管理，集群和存储的网络接口名称。如果未指定名称，则节点的主接口将用于管理，集群和存储网络连接。此外、请务必删除`adsNetworkInterfaces`部分的注释。



集群和存储网络必须位于同一接口上。Astra数据存储管理接口应与Kubernetes节点的管理接口相同。

- (可选) \* 显示器配置 \*：如果要配置 [监控操作员](#)（如果您不使用 Astra Control Center 进行监控，则可选），从部分中删除注释，添加应用代理 CR（监控操作员资源）的命名空间（默认值为 netapp-monitoring），然后添加您在先前步骤中使用的注册表的 repo 路径（yor\_registry\_path）。

d. (可选) \* 自动支持配置 \* : 保留 "AutoSupport" 默认值, 除非您需要配置代理:

- 对于 proxyURL, 使用要用于 AutoSupport 捆绑包传输的端口设置代理的 URL。



为了简明起见, 我们从以下YAML示例中删除了一些注释。

```
apiVersion: astrads.netapp.io/v1beta1
kind: AstraDSCluster
metadata:
  name: astrads-cluster-name
  namespace: astrads-system
spec:
  adsNodeConfig:
    cpu: 9
    memory: 38
    # [Optional] Specify node selector labels to select the nodes for
    # creating ADS cluster
    # adsNodeSelector:
    #   matchLabels:
    #     customLabelKey: customLabelValue
    adsNodeCount: 4
    mvip: ""
  adsDataNetworks:
    - addresses: ""
      netmask:
        # Specify the network interface names to use for management, cluster
        # and storage networks.
        # If none are specified, the node's primary interface will be used for
        # management, cluster and storage networking.
        # To move the cluster and storage networks to a different interface
        # than management, specify all three interfaces to use here.
        # NOTE: The cluster and storage networks need to be on the same
        # interface.
  adsNetworkInterfaces:
    managementInterface: "<mgmt_interface_name>"
    clusterInterface: "<cluster_and_storage_interface_name>"
    storageInterface: "<cluster_and_storage_interface_name>"
    # [Optional] Provide a monitoring config to be used to setup/configure
    # a monitoring agent.
  # monitoringConfig:
  #   # namespace: "netapp-monitoring"
  #   # repo: "[YOUR REGISTRY]"
  autoSupportConfig:
    autoUpload: true
    enabled: true
    coredumpUpload: false
```

```
historyRetentionCount: 25
destinationURL: "https://support.netapp.com/put/AsupPut"
# ProxyURL defines the URL of the proxy with port to be used for
AutoSupport bundle transfer
# proxyURL:
periodic:
  - schedule: "0 0 * * *"
    periodicconfig:
      - component:
          name: storage
          event: dailyMonitoring
          userMessage: Daily Monitoring Storage AutoSupport bundle
          nodes: all
      - component:
          name: controlplane
          event: daily
          userMessage: Daily Control Plane AutoSupport bundle
```

3. 使用 `kubectl apply` 部署集群:

```
kubectl apply -f ./astrads/manifests/astradscluster.yaml
```

4. 等待几分钟, 以完成集群创建操作, 然后验证 Pod 是否正在运行:

```
kubectl get pods -n astrads-system
```

响应示例:

NAME	READY	STATUS	
RESTARTS      AGE			
astrads-cluster-controller-7c67cc7f7b-2jww2 7h31m	1/1	Running	0
astrads-deployment-support-788b859c65-2qjkn 12d	3/3	Running	19
astrads-ds-astrads-cluster-1ab0dbc-j9jzc 5d2h	1/1	Running	0
astrads-ds-astrads-cluster-1ab0dbc-k9wp8 5d1h	1/1	Running	0
astrads-ds-astrads-cluster-1ab0dbc-pwk42 5d2h	1/1	Running	0
astrads-ds-astrads-cluster-1ab0dbc-qhvc6 8h	1/1	Running	0
astrads-ds-nodeinfo-gcmj8 12d	1/1	Running	1
astrads-ds-nodeinfo-j826x 12d	1/1	Running	3
astrads-ds-nodeinfo-vdthh 12d	1/1	Running	3
astrads-ds-nodeinfo-xwgsf 12d	1/1	Running	0
astrads-ds-support-828vw 5d2h	2/2	Running	2
astrads-ds-support-astrads-example-cluster-cfzts 8h	2/2	Running	0
astrads-ds-support-astrads-example-cluster-nzkkr 7h49m	2/2	Running	15
astrads-ds-support-astrads-example-cluster-xxbnp 5d2h	2/2	Running	1
astrads-license-controller-86c69f76bb-s6fb7 8h	1/1	Running	0
astrads-operator-79ff8fbb6d-vpz9m 8h	1/1	Running	0

## 5. 验证集群部署进度：

```
kubectl get astradscluster -n astrads-system
```

响应示例：



NAME	STATUS	VERSION	SERIAL NUMBER	MVIP
AGE				
astrads-example-cluster	created	2022.05.0-X	e100000006	
10.x.x.x	13m			

## 了解与部署相关的事件

在集群部署期间，操作状态应从 `blank` 更改为 `in progress` 更改为 `created`。集群部署将持续大约 8 到 10 分钟。要在部署期间监控集群事件，您可以运行以下命令之一：

```
kubectl get events --field-selector involvedObject.kind=AstraDSCluster -n astrads-system
```

```
kubectl describe astradscluster <cluster name> -n astrads-system
```

以下是部署期间的关键事件：

事件	消息和重要性
已选择ControlPlaneNodesS	已成功选择【编号】控制平面节点以加入ADS集群。Astra数据存储操作员确定了具有CPU、内存、存储和网络的足够多节点、以创建Astra数据存储集群。
ADSClusterCreateInProProgress	Astra Data Store集群控制器已启动集群创建操作。
ADSClusterCreateSuccessess	已成功创建集群。

如果集群的状态未更改为 `in progress`，请查看操作员日志，了解有关节点选择的更多详细信息：

```
kubectl logs -n astrads-system <astrads operator pod name>
```

如果集群状态停留在 `in progress`、请检查集群控制器的日志：

```
kubectl logs -n astrads-system <astrads cluster controller pod name>
```

## 使用Astra控制中心安装Astra数据存储

要在Astra控制中心中部署和使用Astra数据存储、请执行以下操作。

您需要什么？ **#8217** ；将需要什么

- 您已查看 [Astra数据存储的一般前提条件](#)。
- 您已安装Astra控制中心。

## 步骤

1. "使用Astra控制中心部署Astra数据存储"。

## 下一步行动

- \* Kubernetes本机部署和第三方分发版\*：通过执行其他操作完成Astra Data Store部署 ["设置任务"](#)。
- \* Astra控制中心\*：如果您已使用Astra控制中心部署Astra数据存储、则无需遵循这些原则 ["设置任务"](#) 除非您要配置任何其他监控选项。部署Astra数据存储后、您可以使用Astra控制中心UI完成以下任务：
  - ["监控Astra数据存储资产的运行状况"](#)。
  - ["管理Astra Data Store后端存储"](#)。
  - ["监控节点，磁盘和永久性卷声明（PVC）"](#)。

## 设置Astra数据存储组件

你先请 ["安装了独立的Astra数据存储库"](#) 并解决了一些问题 ["环境前提条件"](#)、您将安装Astra Trident、配置Kubernetes快照功能、设置存储后端并创建默认存储类：



如果您使用Astra控制中心部署Astra数据存储、则无需执行这些步骤、除非您打算进行设置 [其他监控选项](#)。

- [\[Install Astra Trident\]](#)
- [\[Install Kubernetes snapshot CRDs and Controller\]](#)
- [\[Set up Astra Data Store as storage backend\]](#)
- [\[Create a default Astra Data Store storage class\]](#)
- [\[Configure Astra Data Store monitoring\]](#)

## 安装 Astra Trident

对于Astra数据存储、您需要安装Astra Trident 21.10.1或更高版本。您可以使用以下选项之一安装 Astra Trident：

- ["使用 tridentctl 安装 Astra Trident"](#)。
- ["使用 Trident 操作员安装 Astra Trident"](#)。



您可以手动或使用 Helm 部署 Trident 操作员。

## 安装 Kubernetes Snapshot CRD 和控制器

要创建永久性卷声明（PVC）快照，需要使用 Kubernetes Snapshot CRD 和控制器。如果尚未为您的环境安装 CRD 和控制器，请运行以下命令进行安装。



以下命令示例假设 `/trident` 作为目录；但是，您使用的目录可以是用于下载 YAML 文件的任何目录。

您需要什么？ #8217 ；将需要什么

- "开始安装之前、请为部署Astra Data Store准备您的环境"。
- 下载 "Kubernetes 快照控制器 YAML 文件":
  - Setup-snapshot-controller.yaml
  - rbac 快照控制器 .yaml
- 下载 "YAML CRD":
  - snapshot.storage.k8s.io\_volumesnapshotclasses.yaml
  - snapshot.storage.k8s.io\_volumesnapshotcontents.yaml
  - snapshot.storage.k8s.io\_volumesnapshots.yaml

## 步骤

1. 应用 snapshot.storage.k8s.io\_volumesnapshotclasses.yaml :

```
kubectl apply -f
trident/snapshot.storage.k8s.io_volumesnapshotclasses.yaml
```

响应:

```
customresourcedefinition.apiextensions.k8s.io/volumesnapshotclasses.snapshot.storage.k8s.io configured
```

2. 应用 snapshot.storage.k8s.io\_volumesnapshotcontents.yaml :

```
kubectl apply -f
trident/snapshot.storage.k8s.io_volumesnapshotcontents.yaml
```

响应:

```
customresourcedefinition.apiextensions.k8s.io/volumesnapshotcontents.snapshot.storage.k8s.io configured
```

3. 应用 snapshot.storage.k8s.io\_volumesnapshots.yaml :

```
kubectl apply -f trident/snapshot.storage.k8s.io_volumesnapshots.yaml
```

响应:

```
customresourcedefinition.apiextensions.k8s.io/volumesnapshots.snapshot.s  
torage.k8s.io configured
```

4. 应用 setup-snapshot-controller.yaml :

```
kubectl apply -f trident/setup-snapshot-controller.yaml
```

响应:

```
deployment.apps/snapshot-controller configured
```

5. 应用 rbac 快照控制器 .yaml :

```
kubectl apply -f trident/rbac-snapshot-controller.yaml
```

响应:

```
serviceaccount/snapshot-controller configured  
clusterrole.rbac.authorization.k8s.io/snapshot-controller-runner  
configured  
clusterrolebinding.rbac.authorization.k8s.io/snapshot-controller-role  
configured  
role.rbac.authorization.k8s.io/snapshot-controller-leaderelection  
configured  
rolebinding.rbac.authorization.k8s.io/snapshot-controller-leaderelection  
configured
```

6. 验证是否已应用 CRD YAML 文件:

```
kubectl get crd | grep volumesnapshot
```

响应示例:

```
astradsvolumesnapshots.astrads.netapp.io      2021-08-04T17:48:21Z
volumesnapshotclasses.snapshot.storage.k8s.io 2021-08-04T22:05:49Z
volumesnapshotcontents.snapshot.storage.k8s.io 2021-08-04T22:05:59Z
volumesnapshots.snapshot.storage.k8s.io       2021-08-04T22:06:17Z
```

## 7. 验证是否已应用快照控制器文件：

```
kubectl get pods -n kube-system | grep snapshot
```

响应示例：

```
snapshot-controller-7f58886ff4-cdh78
1/1      Running    0          13s
snapshot-controller-7f58886ff4-tmrd9
1/1      Running    0          32s
```

## 将 **Astra** 数据存储设置为存储后端

在 `ads_backend.json` 文件中配置存储后端参数，并创建 Astra Data Store 存储后端。

### 步骤

#### 1. 使用安全终端创建 `ads_backend.json`：

```
vi ads_backend.json
```

#### 2. 配置 JSON 文件：



以下步骤将提供一个JSON示例。

- a. 将 `"cluster"` 值更改为 Astra Data Store 集群的集群名称。
- b. 将 `"namespace"` 值更改为要用于创建卷的命名空间。
- c. 将 `"autoExportPolicy"` 值更改为 `true`，除非为此后端设置了 `exportpolicy` CR。
- d. 使用要授予访问权限的 IP 地址填充 `"autosExportCIDRs"` 列表。使用 `0.0.0.0/0` 允许所有。
- e. 对于 `"kubeconfig"` 值，请执行以下操作：
  - i. 将 `.Kube/config` YAML 文件转换为不含空格的 JSON 格式并将其最小化：

转换示例：

```
python3 -c 'import sys, yaml, json;
json.dump(yaml.load(sys.stdin), sys.stdout, indent=None)' <
~/.kube/config > kubeconf.json
```

ii. 编码为 base64 ，并使用 base64 输出作为 ` "kubeconfig" ` 值：

示例编码：

```
cat kubeconf.json | base64 | tr -d '\n'
```

```

{
  "version": 1,
  "storageDriverName": "astrads-nas",
  "storagePrefix": "",
  "cluster": "example-1234584",
  "namespace": "astrads-system",
  "autoExportPolicy": true,
  "autoExportCIDRs": ["0.0.0.0/0"],
  "kubeconfig": "<base64_output_of_kubeconf_json>",
  "debugTraceFlags": {"method": true, "api": true},
  "labels": {"cloud": "on-prem", "creator": "trident-dev"},
  "defaults": {
    "qosPolicy": "silver"
  },
  "storage": [
    {
      "labels": {
        "performance": "extreme"
      },
      "defaults": {
        "qosPolicy": "gold"
      }
    },
    {
      "labels": {
        "performance": "premium"
      },
      "defaults": {
        "qosPolicy": "silver"
      }
    },
    {
      "labels": {
        "performance": "standard"
      },
      "defaults": {
        "qosPolicy": "bronze"
      }
    }
  ]
}

```

3. 更改为下载 Trident 安装程序的目录：

```
cd <trident-installer or path to folder containing tridentctl>
```

#### 4. 创建存储后端：

```
./tridentctl create backend -f ads_backend.json -n trident
```

响应示例：

```
+-----+-----+
+-----+-----+-----+
|      NAME      | STORAGE DRIVER |                      UUID
| STATE  | VOLUMES |
+-----+-----+
+-----+-----+-----+
| example-1234584 | astrads-nas    | 2125fa7a-730e-43c8-873b-
6012fcc3b527 | online |          0 |
+-----+-----+
+-----+-----+-----+
```

## 创建默认的 **Astra Data Store** 存储类

创建 Astra Trident 默认存储类并将其应用于存储后端。

### 步骤

#### 1. 创建 trident CSI 存储类：

##### a. 创建 ads\_sc\_example.yaml：

```
vi ads_sc_example.yaml
```

示例



```

allowVolumeExpansion: true
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  creationTimestamp: "2022-05-09T18:05:21Z"
  name: ads-silver
  resourceVersion: "3361772"
  uid: 1o023456-da4b-51e3-b430-3aa1e3bg111a
mountOptions:
- vers=4
parameters:
  backendType: astrads-nas
  selector: performance=premium
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate

```

#### b. 创建 trident CSI :

```
kubectl create -f ads_sc_example.yaml
```

响应:

```
storageclass.storage.k8s.io/trident-csi created
```

#### 2. 验证是否已添加存储类:

```
kubectl get storageclass
```

响应:

NAME	PROVISIONER	RECLAIMPOLICY	VOLUMEBINDINGMODE
ALLOWVOLUMEEXPANSION	AGE		
ads-silver	csi.trident.netapp.io	Delete	Immediate
true	6h29m		

#### 3. 更改为下载 Trident 安装程序的目录:

```
cd <trident-installer or path to folder containing tridentctl>
```

#### 4. 验证 Astra Trident 后端是否已使用默认存储类参数进行更新：

```
./tridentctl get backend -n trident -o yaml
```

响应示例：

```
items:
- backendUUID: 2125fa7a-730e-43c8-873b-6012fcc3b527
  config:
    autoExportCIDRs:
    - 0.0.0.0/0
    autoExportPolicy: true
    backendName: ""
    cluster: example-1234584
    credentials: null
    debug: false
    debugTraceFlags:
      api: true
      method: true
    defaults:
      exportPolicy: default
      qosPolicy: bronze
      size: 1G
      snapshotDir: "false"
      snapshotPolicy: none
    disableDelete: false
    kubeconfig: <ID>
    labels:
      cloud: on-prem
      creator: trident-dev
    limitVolumeSize: ""
    namespace: astrads-system
    nfsMountOptions: ""
    region: ""
    serialNumbers: null
    storage:
    - defaults:
        exportPolicy: ""
        qosPolicy: gold
        size: ""
        snapshotDir: ""
        snapshotPolicy: ""
      labels:
        performance: extreme
      region: ""
```

```

    supportedTopologies: null
    zone: ""
  - defaults:
      exportPolicy: ""
      qosPolicy: silver
      size: ""
      snapshotDir: ""
      snapshotPolicy: ""
    labels:
      performance: premium
    region: ""
    supportedTopologies: null
    zone: ""
  - defaults:
      exportPolicy: ""
      qosPolicy: bronze
      size: ""
      snapshotDir: ""
      snapshotPolicy: ""
    labels:
      performance: standard
    region: ""
    supportedTopologies: null
    zone: ""
  storageDriverName: astrads-nas
  storagePrefix: ""
  supportedTopologies: null
  version: 1
  zone: ""
configRef: ""
name: example-1234584
online: true
protocol: file
state: online
storage:
  example-1234584_pool_0:
    name: example-1234584_pool_0
    storageAttributes:
      backendType:
        offer:
          - astrads-nas
      clones:
        offer: true
      encryption:
        offer: false
    labels:

```

```

    offer:
      cloud: on-prem
      creator: trident-dev
      performance: extreme
  snapshots:
    offer: true
storageClasses:
- trident-csi
supportedTopologies: null
example-1234584_pool_1:
  name: example-1234584_pool_1
  storageAttributes:
    backendType:
      offer:
        - astrads-nas
    clones:
      offer: true
    encryption:
      offer: false
    labels:
      offer:
        cloud: on-prem
        creator: trident-dev
        performance: premium
    snapshots:
      offer: true
storageClasses:
- trident-csi
supportedTopologies: null
example-1234584_pool_2:
  name: example-1234584_pool_2
  storageAttributes:
    backendType:
      offer:
        - astrads-nas
    clones:
      offer: true
    encryption:
      offer: false
    labels:
      offer:
        cloud: on-prem
        creator: trident-dev
        performance: standard
    snapshots:
      offer: true

```

```
storageClasses:
  - ads-silver
supportedTopologies: null
volumes: []
```

## 配置Astra数据存储监控

(可选)您可以配置Astra数据存储以供其他遥测服务监控。如果您不使用Astra控制中心进行Astra数据存储监控、或者希望将监控扩展到其他端点、则建议使用此操作步骤。

如果您的Astra数据存储实例是独立部署、使用Cloud Insights 监控遥测或将日志流式传输到Elastic等第三方端点、则可以安装监控操作员。



对于Astra控制中心部署、监控操作员会自动进行配置。您可以跳过以下操作步骤 的前两个命令。

在设置监控之前、您需要在`astrads-system`命名空间中有一个活动的Astra数据存储集群。

### 步骤

1. 运行此安装命令：

```
kubectl apply -f ./manifests/monitoring_operator.yaml
```

2. 配置用于监控的Astra数据存储：

```
kubectl astrads monitoring -n netapp-monitoring -r [YOUR REGISTRY] setup
```

3. 配置Astra数据存储以将EMS日志流式传输到弹性端点：

```
kubectl astrads monitoring es --port <portname> --host <hostname>
```

## Astra数据存储早期访问计划(EAP)版本限制

在早期访问计划期间、Astra数据存储具有以下资源限制。

资源	最小值	最大值
Astra数据存储集群中的节点数	4.	16.
每个节点的永久性卷数	不适用	50
卷大小	20 MiB	2 TiB
每个卷的快照数	0	1023
每个卷的克隆数	0	9

资源	最小值	最大值
每个节点的VM数	0	50

## 有关Astra数据存储的常见问题

查找有关Astra Data Store Early Access计划版本的安装、配置、升级和故障排除的常见问题解答。

### 一般问题

\*是否可以使用Astra数据存储早期访问计划版本进行生产？\*否虽然Astra数据存储库的设计和开发旨在提供企业级故障恢复能力、但Astra数据存储库的早期访问计划版本不适用于生产工作负载。

\*是否可以对虚拟机工作负载使用Astra数据存储？\*是。Astra数据存储同时支持Kubernetes和VMware VVol工作负载。

请参见 ["了解采用VMware的Astra Data Store"](#)。

\*是否可以使用VMware vSphere管理Astra数据存储？\*是的、可以使用适用于VMware vSphere的NetApp Astra插件在vCenter中本机管理Astra数据存储。请参见 ["管理VMware安装的Astra Data Store组件"](#)。

- Astra数据存储是否依赖于其他NetApp产品才能正常运行？\*

是的。Astra Data Store要求在工作负载Kubernetes集群上部署NetApp CSI驱动程序Astra Trident 21.10.1及更高版本。了解相关信息 ["Astra 数据存储要求"](#)。

要启用VMware工作流和适用于VMware vSphere的NetApp Astra插件、需要使用Astra控制中心。

使用Astra Data Store集群作为存储后端的应用程序可以使用 ["Astra 控制中心"](#) 利用应用程序感知型数据管理功能、包括数据保护、灾难恢复和Kubernetes工作负载迁移。

\*如何管理Astra数据存储集群？\*您可以使用kubectl命令和Kubernetes API扩展来管理Astra数据存储资源。

`kubectl astrad` 命令包含一个 `-h` 交换机，可为您提供使用情况和标志文档。

\*如何监控Astra数据存储集群指标？\*您可以使用Cloud Insights 监控Astra数据存储指标(请参见 ["使用 Cloud Insights 监控指标"](#))或Kubernetes原生 监控(请参见 ["使用Prometheus和Grafana进行监控"](#))。

您还可以监控日志。请参见 ["配置和监控事件日志"](#)。

\*我是否可以在Kubernetes集群中将Astra数据存储与ONTAP 或其他存储提供程序结合使用？\*是。Astra数据存储可与应用程序集群中的其他存储提供程序结合使用。

\*如果从Astra数据存储库中删除Kubernetes集群、则是否会卸载Astra Trident？\*如果卸载Astra数据存储库、则不会从集群中卸载Astra Trident。如果需要卸载 Astra Trident ，则必须单独执行此操作。

### 许可

- Astra数据存储是否需要许可证？\*是的、Astra数据存储需要早期访问计划的评估版NetApp许可证文件(NLF)。

请参见 ["Astra 数据存储要求"](#)。

- Astra Data Store评估许可证有效期有多长时间？\* Astra Data Store许可证的默认期限为自下载日期起90天。

## 在Kubernetes集群上安装和使用Astra数据存储

\*是否可以在运行于裸机或虚拟机上的Kubernetes集群上安装Astra数据存储？\*是。Astra数据存储可以安装在裸机上运行的Kubernetes集群或vSphere VM上。请参见 ["Astra 数据存储要求"](#)。

### 支持哪些版本的Kubernetes for Astra Data Store？

Astra数据存储可与v1.20及更高版本兼容的Kubernetes分发版配合使用。但是，目前尚未对所有 Kubernetes 分发版进行验证。了解相关信息 ["Astra 数据存储要求"](#)。

\*我的Kubernetes集群超过4个工作节点。是否可以在其中安装Astra数据存储？\*是。Astra数据存储集群最初必须部署在Kubernetes集群中的4个辅助节点上。部署完成后、您可以将集群最多扩展到16个辅助节点。

- Astra数据存储是否支持从专用注册表脱机安装？\*是。可以从本地注册表脱机安装Astra数据存储。请参见 ["安装 Astra 数据存储"](#)。

\*是否需要Internet连接才能使用Astra Data Store？\*否、Astra Data Store Early Access Program不需要Internet连接。但是、建议连接到NetApp AutoSupport 后端、以便定期发送遥测捆绑包。此连接可以是直接连接，也可以通过代理进行连接。

- Astra数据存储使用哪些角色和特权？\*您需要成为Kube管理员才能部署Astra数据存储操作员。

Astra数据存储具有一个名为`astrads-ds-nodeinfo`的特权取消命名集、用于发现在选择节点时使用的节点资源。

此外、操作员还将使用具有权限的Kubernetes作业在选定工作节点上安装存储集群的容器、以构建Astra Data Store存储集群。

从下载的Astra Data Store捆绑包中\*安装Astra Data Store需要更新哪些清单文件？\* ["NetApp 支持站点"](#)，您将获得以下清单：

- astradscluster.yaml
- astradsoperator.yaml
- astradsversion.yaml
- monitoring\_operator.yaml

您需要使用特定于部署的配置更新 `astradscluster.yaml` 清单。请参见 ["安装 Astra 数据存储"](#)。

## 故障排除和支持

借助Astra Data Store、您可以使用NetApp容器可宽延通道访问社区支持。此渠道由 NetApp 支持和我们的技术营销工程师监控。

["NetApp 容器 Slack 通道"](#)

请参见 ["Astra 数据存储支持操作"](#)。

- 如何提出支持案例或要求对快速问题进行澄清？ \* 要提出支持案例或获得有关快速问题的澄清，请在上报告您的问题描述或问题 ["NetApp 容器 Slack 通道"](#)。NetApp 支持部门将与您密切合作，尽最大努力提供帮助。
- 如何申请新功能？ \* 如果您对支持的配置或功能有任何疑问，请联系 [astra.feedback@netapp.com](mailto:astra.feedback@netapp.com)。
- 如何生成支持日志包？ \* 请参见 ["生成支持包"](#) 有关为Astra数据存储设置和下载支持日志包的说明。
- Astra数据存储无法找到我的Kubernetes节点。如何修复此问题？ \* 请参见 ["安装 Astra 数据存储"](#)。
- IPv6地址是否可用于管理、数据和集群网络？ \* 否、Astra数据存储仅支持IPv4地址。未来版本的Astra Data Store将增加对IPv6的支持。

\*在Astra Data Store上配置卷时使用的是什么NFS版本？ \*对于为Kubernetes应用程序配置的所有卷、Astra Data Store支持NFS v4.1、而对于为VMware工作负载配置的所有卷、则支持NFSv3。

请参见 ["Astra 数据存储要求"](#) 和 ["Astra数据存储限制"](#)。

## 升级**Astra**数据存储

\*是否可以从Astra Data Store预览版升级？ \*是。您可以从Astra数据存储早期访问计划版本升级到未来版本。



# 使用 **Astra** 数据存储

## 使用**kubectl**命令管理**Astra Data Store**资源

您可以使用kubectl命令和Kubernetes API扩展来管理Astra Data Store资源。

要了解如何部署示例应用程序，请参见 ["部署测试应用程序"](#)。

有关集群维护信息、请参见 ["管理集群"](#)。

您需要什么？ [#8217](#) ；将需要什么

- 您安装在中的Astra Data Store kubectl插件 ["安装 Astra 数据存储"](#)

## 列出适用于**Astra**数据存储的**Kubernetes**自定义**API**资源

您可以在Kubernetes中使用kubectl命令与Astra Data Store集群进行交互并观察其状态。

`api-resources`命令中列出的每个项都表示一个Kubernetes自定义资源定义(CRD)、Astra Data Store可在内部使用该定义定义来管理集群。

此列表对于获取每个Astra Data Store对象的短名称以减少键入内容特别有用、如后面所示。

1. 显示适用于Astra数据存储的Kubernetes自定义API资源列表：

```
kubectl api-resources --api-group astrads.netapp.io
```

响应：

NAME	SHORTNAMES	APIGROUP	NAMESPACED	KIND
astradsautosupports	adsas	astrads.netapp.io/v1alpha1	true	
AstraDSAutoSupport				
astradscLOUDsnapshots	adscs	astrads.netapp.io/v1alpha1	true	
AstraDSCloudSnapshot				
astradsclusters	adscl	astrads.netapp.io/v1alpha1	true	
AstraDSCluster				
astradsexportpolicies	adsep	astrads.netapp.io/v1alpha1	true	
AstraDSEExportPolicy				
astradsfaileddrives	adsfd	astrads.netapp.io/v1alpha1	true	
AstraDSFailedDrive				
astradslicenses	adsli	astrads.netapp.io/v1alpha1	true	
AstraDSLICENSE				
astradsnfsoptions	adsnf	astrads.netapp.io/v1alpha1	true	
AstraDSNfsOption				
astradsnodeinfoes	adsni	astrads.netapp.io/v1alpha1	true	
AstraDSNodeInfo				
astradsnodemanagements	adsnm	astrads.netapp.io/v1alpha1	true	
AstraDSNodeManagement				
astradsqospolicies	adsqp	astrads.netapp.io/v1alpha1	true	
AstraDSQosPolicy				
astradsversions	adsve	astrads.netapp.io/v1alpha1	true	
AstraDSVersion				
astradsvolumeFiles	adsvf	astrads.netapp.io/v1alpha1	true	
AstraDSVolumeFiles				
astradsvolumes	adsvo	astrads.netapp.io/v1alpha1	true	
AstraDSVolume				
astradsvolumesnapshots	adsvs	astrads.netapp.io/v1alpha1	true	
AstraDSVolumeSnapshot				

2. 要获取Kubernetes集群中的所有当前Astra Data Store对象、请使用`kubectl get ADS -a`命令：

```
kubectl get ads -A
```

响应：

NAMESPACE	NAME	AGE
astrads-system	astradsqospolicy.astrads.netapp.io/bronze	45h
astrads-system	astradsqospolicy.astrads.netapp.io/gold	45h
astrads-system	astradsqospolicy.astrads.netapp.io/silver	45h

NAMESPACE	NAME			
STATUS	VERSION	SERIAL NUMBER	MVIP	AGE

```
astrads-system    astradscluster.astrads.netapp.io/astrads-cluster-9f1
created    arda-9.11.1    e000000009    10.224.8.146    46h
```

```
NAMESPACE    NAME
```

```
AGE
```

```
astrads-system    astradsnodeinfo.astrads.netapp.io/englab.netapp.com
46h
```

```
astrads-system    astradsnodeinfo.astrads.netapp.io/englab.netapp.com
46h
```

```
astrads-system    astradsnodeinfo.astrads.netapp.io/englab.netapp.com
46h
```

```
astrads-system    astradsnodeinfo.astrads.netapp.io/englab.netapp.com
46h
```

```
NAMESPACE    NAME    AGE
astrads-system    astradsversion.astrads.netapp.io/astradsversion    46h
```

```
NAMESPACE    NAME    AGE
astrads-system    astradsvolumefiles.astrads.netapp.io/test23    27h
astrads-system    astradsvolumefiles.astrads.netapp.io/test234    27h
astrads-system    astradsvolumefiles.astrads.netapp.io/test2345    4h22m
```

```
NAMESPACE    NAME    SIZE    IP
CLUSTER    CREATED
astrads-system    astradsvolume.astrads.netapp.io/test234    21Gi
172.25.123.123    astrads-cluster-9f1    true
astrads-system    astradsvolume.astrads.netapp.io/test2345    21Gi
172.25.123.123    astrads-cluster-9f1    true
```

```
NAMESPACE    NAME
SEQUENCE COMPONENT    EVENT    TRIGGER    PRIORITY    SIZE
STATE
astrads-system    astradsautosupport.astrads.netapp.io/controlplane-
adsclustercreatesuccess-20211214t 9    controlplane
adsclustercreatesuccess k8sEvent    notice    0    uploaded
astrads-system    astradsautosupport.astrads.netapp.io/controlplane-
daily-20211215t0    15    controlplane    daily
periodic    notice    0    uploaded
astrads-system    astradsautosupport.astrads.netapp.io/controlplane-
daily-20211216t0    20    controlplane    daily
periodic    notice    0    uploaded
astrads-system    astradsautosupport.astrads.netapp.io/storage-
callhome.dbs.cluster.cannot.sync.blocks 10    storage
callhome.dbs.cluster.cannot.sync.blocks    firetapEvent    emergency    0
uploaded
```

NAMESPACE	NAME	ADSCUSTER
VALID PRODUCT	EVALUATION ENDDATE	VALIDATED
astrads-system	astradslicense.astrads.netapp.io/e0	astrads-cluster-
9f1 true	Astra Data Store true	2022-02-07 2021-12-16T20:43:23Z

### 3. 使用以下短名称之一显示集群中卷的当前状态：

```
kubectl get adsvo -A
```

响应：

NAMESPACE	NAME	SIZE	IP	CLUSTER
CREATED				
astrads-system	test234	21Gi	172.25.138.109	astrads-cluster-
9f1c99f true				
astrads-system	test2345	21Gi	172.25.138.111	astrads-cluster-
9f1c99f true				

## 使用 **kubectl** 扩展上的 **help** 选项

NetApp为`kubectl`命令提供了`astrad`扩展、可用于执行Astra Data Store集群管理任务、例如添加许可证、管理节点、解决问题以及扩展Astra Data Store集群。`astrad`扩展包括一个`-h`选项、用于提供有关如何使用扩展以及执行哪些任务的信息。

### 1. 显示有关Astra Data Store `kubectl`扩展中所有命令的帮助：

```
kubectl astrads -h
```

响应：

```
A kubectl plugin for inspecting your AstraDS deployment

Usage:
  astrads [command]

Available Commands:
  asup          Manage AutoSupport
  clusters      Manage clusters
  drives        Manage drives in a cluster
  faileddrive   Manage drive replacement in a cluster
  help          Help about any command
  license       Manage license in the astrads cluster
```

maintenance Manage maintenance status of a node  
 monitoring Manage Monitoring Output  
 nodes Manage nodes in a cluster

#### Flags:

<code>--as</code> string	Username to impersonate for the operation
<code>--as-group</code> stringArray	Group to impersonate for the operation, this flag can be repeated to specify multiple groups.
<code>--cache-dir</code> string	Default HTTP cache directory (default "/u/arda/.kube/http-cache")
<code>--certificate-authority</code> string	Path to a cert file for the certificate authority
<code>--client-certificate</code> string	Path to a client certificate file for TLS
<code>--client-key</code> string	Path to a client key file for TLS
<code>--cluster</code> string	The name of the kubeconfig cluster to use
<code>--context</code> string	The name of the kubeconfig context to use
<code>-h, --help</code>	help for astrads
<code>--insecure-skip-tls-verify</code>	If true, the server's certificate will not be checked for validity. This will make your HTTPS connections insecure
<code>--kubeconfig</code> string	Path to the kubeconfig file to use for CLI requests.
<code>-n, --namespace</code> string	If present, the namespace scope for this CLI request
<code>--request-timeout</code> string	The length of time to wait before giving up on a single server request. Non-zero values should contain a
	corresponding time unit (e.g. 1s, 2m, 3h).
	A value of zero means don't timeout requests.
<code>-s, --server</code> string	(default "0") The address and port of the Kubernetes API server
<code>--token</code> string	Bearer token for authentication to the API server
<code>--user</code> string	The name of the kubeconfig user

to use

2. 有关命令的详细信息，请使用 `astrad [command] -help`。

```
kubectl astrads asup collect --help
```

响应：

Collect the autosupport bundle by specifying the component to collect.  
It will default to manual event.

Usage:

```
astrads asup collect [flags]
```

Examples:

```
# Control plane collection
```

```
kubectl astrads collect --component controlplane example1
```

```
# Storage collection for single node
```

```
kubectl astrads collect --component storage --nodes node1 example2
```

```
# Storage collection for all nodes
```

```
kubectl astrads collect --component storage --nodes all example3
```

```
# Collect but don't upload to support
```

```
kubectl astrads collect --component controlplane --local example4
```

NOTE:

```
--component storage and --nodes <name> are mutually inclusive.
```

```
--component controlplane and --nodes <name> are mutually  
exclusive.
```

Flags:

```
-c, --component string      Specify the component to collect:  
[storage , controlplane , vasaprovider, all]
```

```
-d, --duration int          Duration is the duration in hours from  
the startTime for collection  
of AutoSupport.
```

```
-e, --event string          Specify the callhome event to trigger.  
(default "manual")
```

```
-f, --forceUpload           Configure an AutoSupport to upload if  
it is in the compressed state  
and not
```

the 'local' option or if	uploading because it was created with
disabled	automatic uploads of AutoSupports is
	at the cluster level.
-h, --help	help for collect
-l, --local	Only collect and compress the
autosupport bundle. Do not upload	to support.
	Use 'download' to copy the collected
bundle after it is in	the 'compressed' state
	Specify nodes to collect for storage
--nodes string	
component. (default "all")	
-t, --startTime string	StartTime is the starting time for
collection of AutoSupport.	
	This should be in the ISO 8601 date
time format.	
	Example format accepted:
	2021-01-01T15:20:25Z, 2021-01-
01T15:20:25-05:00	
-u, --usermessage string	UserMessage is the additional message
to include in the	
	AutoSupport subject.
	(default "Manual event trigger from
CLI")	

## 部署测试应用程序

以下是部署可与 Astra Data Store 结合使用的测试应用程序的步骤。

在此示例中，我们使用 Helm 存储库从 BitNami 部署 MongoDB 图表。

您需要什么？ **#8217** ；将需要什么

- 已部署和配置 Astra Data Store 集群
- Trident 安装已完成

步骤

1. 从 BitNami 添加 Helm repo :

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

2. 部署 MongoDB :

```
helm install mongohelm4 --set persistence.storageClass=trident-csi
bitnami/mongodb --namespace=ns-mongodb --create-namespace
```

### 3. 检查 MongoDB POD 的状态:

```
~% kubectl get pods -n ns-mongodb
```

NAME	READY	STATUS	RESTARTS	AGE
mongodb-9846ff8b7-rfr4r	1/1	Running	0	67s

### 4. 验证 MongoDB 使用的永久性卷声明 (PVC) :

```
~% kubectl get pvc -n ns-mongodb
```

NAME	STATUS	VOLUME	CAPACITY	ACCESS MODES
STORAGECLASS	AGE			
mongodb	Bound	pvc-1133453a-e2f5-48a5	8Gi	RWO
trident-csi	97s			

### 5. 使用 kubectl 命令 get astradsvolume 列出卷:

```
~% kubectl get astradsvolume pvc-1133453a-e2f5-48a5 -n astrads-system
```

NAME	SIZE	IP	CLUSTER	CREATED
pvc-1133453a-e2f5-48a5	8830116Ki	10.192.2.192	jai-ads	true

### 6. 使用 kubectl 命令 describe astradsvolume 描述卷:

```
~% kubectl describe astradsvolume pvc-1133453a-e2f5-48a5 -n astrads-
system
```

Name: pvc-1133453a-e2f5-48a5-a06c-d14b8aa7be07

Namespace: astrads-system

Labels: astrads.netapp.io/cluster=jai-ads  
astrads.netapp.io/mip=10.192.1.39  
astrads.netapp.io/volumeUUID=cf33fd38-a451-596c-b656-61b8270d2b5e  
trident.netapp.io/cloud=on-prem  
trident.netapp.io/creator=trident-dev  
trident.netapp.io/performance=premium

Annotations: provisioning: {"provisioning":{"cloud":"on-prem","creator":"trident-dev","performance":"premium"}}  
trident:  
{"trident":{"version":"21.10.0-test.jenkins-trident-stable-v21.10-



```

2+e03219ce37294d9ba54ec476bbe788c1a7772548","backendUUID":"","platform":
...
API Version:  astrads.netapp.io/v1alpha1
Kind:          AstraDSVolume
Metadata:
  Creation Timestamp:  2021-12-08T19:35:26Z
  Finalizers:
    trident.netapp.io/astradsvolume-finalizer
    astrads.netapp.io/astradsvolume-finalizer
  Generation:  1
  Managed Fields:
    API Version:  astrads.netapp.io/v1alpha1
    Fields Type:  FieldsV1
    fieldsV1:
      f:metadata:
        f:labels:
          f:astrads.netapp.io/cluster:
          f:astrads.netapp.io/mip:
          f:astrads.netapp.io/volumeUUID:
        f:status:
          .:
          f:cluster:
          f:conditions:
          f:created:
          f:displayName:
          f:exportAddress:
          f:internalName:
          f:mip:
          f:permissions:
          f:qosPolicy:
          f:requestedSize:
          f:restoreCacheSize:
          f:size:
          f:snapshotReservePercent:
          f:state:
          f:volumePath:
          f:volumeUUID:
    Manager:      cluster-controller
    Operation:    Update
    Time:         2021-12-08T19:35:32Z
    API Version:  astrads.netapp.io/v1alpha1
    Fields Type:  FieldsV1
    fieldsV1:
      f:status:
        f:exportPolicy:
    Manager:      dms-controller

```

```

Operation:      Update
Subresource:    status
Time:           2021-12-08T19:35:32Z
API Version:    astrads.netapp.io/v1alpha1
Fields Type:    FieldsV1
fieldsV1:
  f:metadata:
    f:annotations:
      .:
    f:provisioning:
    f:trident:
  f:finalizers:
    v:"trident.netapp.io/astradsvolume-finalizer":
  f:labels:
    .:
    f:trident.netapp.io/cloud:
    f:trident.netapp.io/creator:
    f:trident.netapp.io/performance:
  f:spec:
    .:
    f:cluster:
    f:displayName:
    f:exportPolicy:
    f:noSnapDir:
    f:permissions:
    f:qosPolicy:
    f:size:
    f:snapshotReservePercent:
    f:type:
    f:volumePath:
Manager:        trident_orchestrator
Operation:      Update
Time:           2021-12-08T19:35:34Z
Resource Version: 12007115
UID:            d522ae4f-e793-49ed-bbe0-9112d7f9167b
Spec:
  Cluster:      jai-ads
  Display Name:  pvc-1133453a-e2f5-48a5-a06c-d14b8aa7be07
  Export Policy: pvc-1133453a-e2f5-48a5-a06c-d14b8aa7be07
  No Snap Dir:  true
  Permissions:  0777
  Qos Policy:   silver
  Size:         9042036412
  Snapshot Reserve Percent: 5
  Type:         ReadWrite
  Volume Path:  /pvc-1133453a-e2f5-48a5-a06c-d14b8aa7be07

```

```

Status:
Cluster:  jai-ads
Conditions:
  Last Transition Time:  2021-12-08T19:35:32Z
  Message:              Volume is online
  Reason:               VolumeOnline
  Status:               True
  Type:                 AstraDSVolumeOnline
  Last Transition Time:  2021-12-08T19:35:32Z
  Message:              Volume creation request was successful
  Reason:               VolumeCreated
  Status:               True
  Type:                 AstraDSVolumeCreated
Created:                true
Display Name:           pvc-1133453a-e2f5-48a5-a06c-d14b8aa7be07
Export Address:         10.192.2.192
Export Policy:          pvc-1133453a-e2f5-48a5-a06c-d14b8aa7be07
Internal Name:          pvc_1133453a_e2f5_48a5_a06c_d14b8aa7be07
Mip:                    10.192.1.192
Permissions:            777
Qos Policy:             silver
Requested Size:          9042036412
Restore Cache Size:     0
Size:                   8830116Ki
Snapshot Reserve Percent: 5
State:                  online
Volume Path:            /pvc-1133453a-e2f5-48a5-a06c-d14b8aa7be07
Volume UUID:            cf33fd38-a451-596c-b656-61b8270d2b5e
Events:
  Type      Reason          Age   From                      Message
  ----      -
  Normal    VolumeCreated  3m9s  ADSClusterController      Volume creation
request was successful

```

## 管理Astra数据存储集群

您可以在Astra Data Store中使用kubectl命令来管理集群。

- [\[Add a node\]](#)
- [\[Remove a node\]](#)
- [\[Place a node in maintenance mode\]](#)
- [\[Add drives to a node\]](#)
- [\[Replace a drive\]](#)

您需要什么？ **#8217** ； 将需要什么

- 安装了 kubectl 和 kubectl-astrad 插件的系统。请参见 ["安装 Astra 数据存储"](#)。

## 添加节点

要添加的节点应属于 Kubernetes 集群，并且其配置应与集群中的其他节点类似。



要使用Astra控制中心添加节点、请参见 ["将节点添加到存储后端集群"](#)。

### 步骤

1. 如果新节点的dataIP尚未加入Astra Data Store集群CR、请执行以下操作：
  - a. 编辑集群CR并在`adsDataNetworks`\*地址\*字段中添加额外的dataIP。将以大写字母表示的信息替换为适合您环境的值：

```
kubectl edit astradscluster CLUSTER_NAME -n astrads-system
```

- b. 保存 CR 。
- c. 将节点添加到Astra Data Store集群。将以大写字母表示的信息替换为适合您环境的值：

```
kubectl astrads nodes add --cluster CLUSTER_NAME
```

2. 否则、只需添加节点即可。将以大写字母表示的信息替换为适合您环境的值：

```
kubectl astrads nodes add --cluster CLUSTER_NAME
```

3. 验证是否已添加此节点：

```
kubectl astrads nodes list
```

## 删除节点

对Astra数据存储使用kubectl命令删除集群中的节点。

### 步骤

1. 列出所有节点：

```
kubectl astrads nodes list
```

响应：

NODE NAME	NODE STATUS	CLUSTER NAME
sti-rx2540-534d...	Added	cluster-multinodes-21209
sti-rx2540-535d...	Added	cluster-multinodes-21209
...		

2. 标记要删除的节点。将以大写字母表示的信息替换为适合您环境的值：

```
kubectl astrads nodes remove NODE_NAME
```

响应：

```
Removal label set on node sti-rx2540-534d.lab.org  
Successfully updated ADS cluster cluster-multinodes-21209 desired node  
count from 4 to 3
```

将节点标记为要删除后、节点状态应从`active`更改为`present`。

3. 验证已删除节点的`Present`状态：

```
kubectl get nodes --show-labels
```

响应：

NAME	STATUS	ROLES
sti-astramaster-050.lab.org v1.20.0 beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=sti-astramaster-050.lab.org,kubernetes.io/os=linux,node-role.kubernetes.io/control-plane=,node-role.kubernetes.io/master=	Ready	control-plane,master 3h39m
sti-rx2540-556a.lab.org v1.20.0 astrads.netapp.io/cluster=astrads-cluster-890c32c,astrads.netapp.io/storage-cluster-status=active,beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=sti-rx2540-556a.lab.org,kubernetes.io/os=linux,node-role.kubernetes.io/worker=true	Ready	worker 3h38m
sti-rx2540-556b.lab.org v1.20.0 astrads.netapp.io/cluster=astrads-cluster-890c32c,astrads.netapp.io/storage-cluster-status=active,beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=sti-rx2540-556b.lab.org,kubernetes.io/os=linux,node-role.kubernetes.io/worker=true	Ready	worker 3h38m
sti-rx2540-534d.lab.org v1.20.0 astrads.netapp.io/storage-cluster-status=present,astrads.netapp.io/storage-node-removal=,beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=sti-rx2540-557a.lab.org,kubernetes.io/os=linux,node-role.kubernetes.io/worker=true	Ready	worker 3h38m
sti-rx2540-557b.lab.org v1.20.0 astrads.netapp.io/cluster=astrads-cluster-890c32c,astrads.netapp.io/storage-cluster-status=active,beta.kubernetes.io/arch=amd64,beta.kubernetes.io/os=linux,kubernetes.io/arch=amd64,kubernetes.io/hostname=sti-rx2540-557b.lab.org,kubernetes.io/os=linux,node-role.kubernetes.io/worker=true	Ready	worker 3h38m

4. 从节点中卸载Astra数据存储。将以大写字母表示的信息替换为适合您环境的值：

```
kubectl astrads nodes uninstall NODE_NAME
```

5. 验证是否已从集群中删除此节点：

```
kubectl astrads nodes list
```

此节点将从Astra数据存储中删除。

## 将节点置于维护模式

需要执行主机维护或软件包升级时，应将节点置于维护模式。



此节点必须已属于Astra Data Store集群。

当节点处于维护模式时，您无法向集群添加节点。在此示例中、我们会将节点`nhcitjj1525`置于维护模式。

### 步骤

1. 显示节点详细信息：

```
kubectl get nodes
```

响应：

NAME	STATUS	ROLES	AGE	VERSION
nhcitjj1525	Ready	<none>	3d18h	v1.23.5
nhcitjj1526	Ready	<none>	3d18h	v1.23.5
nhcitjj1527	Ready	<none>	3d18h	v1.23.5
nhcitjj1528	Ready	<none>	3d18h	v1.23.5
scs000039783-1	Ready	control-plane,master	3d18h	v1.23.5

2. 确保节点尚未处于维护模式：

```
kubectl astrads maintenance list
```

响应（没有节点处于维护模式）：

NAME	NODE NAME	IN MAINTENANCE	MAINTENANCE STATE	MAINTENANCE VARIANT
------	-----------	----------------	-------------------	---------------------

3. 启用维护模式。将以大写字母表示的信息替换为适合您环境的值：

```
kubectl astrads maintenance create CR_NAME --node-name=NODE_NAME  
--variant=Node
```

例如：

```
kubectl astrads maintenance create maint1 --node-name="nhcitjj1525"  
--variant=Node
```

响应:

```
Maintenance mode astrads-system/maint1 created
```

#### 4. 列出节点:

```
kubectl astrads nodes list
```

响应:

NODE NAME	NODE STATUS	CLUSTER NAME
nhcitjj1525	Added	ftap-astra-012
...		

#### 5. 检查维护模式的状态:

```
kubectl astrads maintenance list
```

响应:

NAME	NODE NAME	IN MAINTENANCE	MAINTENANCE STATE
MAINTENANCE VARIANT			
node4	nhcitjj1525	true	ReadyForMaintenance Node

维护`模式下的 将以 `false 开头, 并更改为 true。M状态 从 PreparingForMaintenance 更改为 ReadyforMaintenance。

#### 6. 完成节点维护后, 禁用维护模式:

```
kubectl astrads maintenance update maint1 --node-name="nhcitjj1525"
--variant=None
```

#### 7. 确保节点不再处于维护模式:

```
kubectl astrads maintenance list
```

## 向节点添加驱动器

将kubectl命令与Astra Data Store结合使用、以便向Astra Data Store集群中的节点添加物理或虚拟驱动器。



您需要什么？ #8217 ；将需要什么

- 一个或多个满足以下条件的驱动器：
  - 已安装在节点(物理驱动器)中或已添加到节点VM (虚拟驱动器)中
  - 驱动器上无分区
  - 驱动器当前未被集群使用
  - 驱动器原始容量不会超过集群中已获得许可的原始容量(例如、如果许可证为每个CPU核心授予2 TB存储空间、则一个包含10个节点的集群的最大原始驱动器容量为20 TB)
  - 驱动器大小至少与节点中其他活动驱动器的大小相同



Astra数据存储每个节点所需的驱动器不超过16个。如果您尝试添加第17个驱动器、则驱动器添加请求将被拒绝。

## 步骤

### 1. 描述集群：

```
kubectl astrads clusters list
```

响应：

CLUSTER NAME	CLUSTER STATUS	NODE COUNT
cluster-multinodes-21209	created	4

### 2. 记下集群名称。

### 3. 显示可添加到集群中所有节点的驱动器。将cluster\_name替换为集群名称：

```
kubectl astrads adddrive show-available --cluster=CLUSTER_NAME
```

响应：

Current cluster drive add status  
Licensed cluster capacity: 72.0 TiB  
Cluster capacity used: 2.3 TiB  
Maximum node size without stranding: 800.0 GiB

Node: node1.name  
Current node size: 600.0 GiB  
Maximum licensed node size: 18.0 TiB  
Total size that can be added to this node without stranding: 200.0 GiB  
Add drive minimum/recommended size: 100.0 GiB. Larger disks will be constrained to this size  
NAME IDPATH SERIAL PARTITIONCOUNT SIZE ALREADYINCLUSTER  
sdg /dev/disk/by-id/scsi-3c290e16d52479a9af5eac c290e16d52479a9af5eac 0  
100 GiB false  
sdh /dev/disk/by-id/scsi-3c2935798df68355dee0be c2935798df68355dee0be 0  
100 GiB false

Node: node2.name  
Current node size: 600.0 GiB  
Maximum licensed node size: 18.0 TiB  
Total size that can be added to this node without stranding: 200.0 GiB  
Add drive minimum/recommended size: 100.0 GiB. Larger disks will be constrained to this size  
No suitable drives to add exist.

Node: node3.name  
Current node size: 600.0 GiB  
Maximum licensed node size: 18.0 TiB  
Total size that can be added to this node without stranding: 200.0 GiB  
Add drive minimum/recommended size: 100.0 GiB. Larger disks will be constrained to this size  
NAME IDPATH SERIAL PARTITIONCOUNT SIZE ALREADYINCLUSTER  
sdg /dev/disk/by-id/scsi-3c29ee82992ed7a36fc942 c29ee82992ed7a36fc942 0  
100 GiB false  
sdh /dev/disk/by-id/scsi-3c29312aa362469fb3da9c c29312aa362469fb3da9c 0  
100 GiB false

Node: node4.name  
Current node size: 600.0 GiB  
Maximum licensed node size: 18.0 TiB  
Total size that can be added to this node without stranding: 200.0 GiB  
Add drive minimum/recommended size: 100.0 GiB. Larger disks will be constrained to this size  
No suitable drives to add exist.

#### 4. 执行以下操作之一：

- 如果所有可用驱动器都具有相同的名称、则可以同时将其添加到相应的节点。将以大写字母表示的信息替换为适合您环境的值：

```
kubectl astrads adddrive create --cluster=CLUSTER_NAME --name  
REQUEST_NAME --drivesbyname all=DRIVE_NAME
```

- 如果驱动器的名称不同、您可以一次将其添加到相应的节点中一个(您需要对需要添加的每个驱动器重复此步骤)。将以大写字母表示的信息替换为适合您环境的值：

```
kubectl astrads adddrive create --cluster=CLUSTER_NAME --name  
REQUEST_NAME --drivesbyname NODE_NAME=DRIVE_NAME
```

Astra数据存储创建一个请求以添加一个或多个驱动器、此时将显示一条消息、其中包含此请求的结果。

### 更换驱动器

当集群中的驱动器发生故障时，必须尽快更换驱动器以确保数据完整性。如果某个驱动器发生故障、您可以查看有关集群CR节点状态中的故障驱动器的信息、集群运行状况信息以及指标端点。您可以使用以下示例命令查看故障驱动器信息。

在 **nodeStatuss.driveStatuses** 中显示故障驱动器的集群示例

```
kubectl get adsc1 -A -o yaml
```

响应：

```
...  
apiVersion: astrads.netapp.io/v1alpha1  
kind: AstraDSCluster  
...  
nodeStatuses:  
  - driveStatuses:  
    - driveID: 31205e51-f592-59e3-b6ec-185fd25888fa  
      driveName: scsi-36000c290ace209465271ed6b8589b494  
      drivesStatus: Failed  
    - driveID: 3b515b09-3e95-5d25-a583-bee531ff3f31  
      driveName: scsi-36000c290ef2632627cb167a03b431a5f  
      drivesStatus: Active  
    - driveID: 0807fa06-35ce-5a46-9c25-f1669def8c8e  
      driveName: scsi-36000c292c8fc037c9f7e97a49e3e2708  
      drivesStatus: Active  
...
```

故障驱动器CR会在集群中自动创建、其名称与故障驱动器的UUID对应。

```
kubectl get adsfd -A -o yaml
```

响应：

```
...
apiVersion: astrads.netapp.io/v1alpha1
kind: AstraDSFailedDrive
metadata:
  name: c290a-5000-4652c-9b494
  namespace: astrads-system
spec:
  executeReplace: false
  replaceWith: ""
status:
  cluster: arda-6e4b4af
  failedDriveInfo:
    failureReason: AdminFailed
    inUse: false
    name: scsi-36000c290ace209465271ed6b8589b494
    path: /dev/disk/by-id/scsi-36000c290ace209465271ed6b8589b494
    present: true
    serial: 6000c290ace209465271ed6b8589b494
    node: sti-rx2540-300b.lab.org
  state: ReadyToReplace
```

```
kubectl astrads faileddrive list --cluster arda-6e4b4af
```

响应：

NAME	NODE	CLUSTER	STATE
AGE			
6000c29013m	sti-rx2540-300b.lab.netapp.com	ard-6e4b4af	ReadyToReplace

## 步骤

1. 使用`kubectl astrad faileddrive show-replacements`命令列出可能的替代驱动器、该命令可筛选符合更换限制(集群中未使用、未挂载、无分区且等于或大于故障驱动器)的驱动器。

要在不筛选可能的替代驱动器的情况下列出所有驱动器，请在 `show-replacements` 命令中添加 `-all` 。

```
kubectl astrads faileddrive show-replacements --cluster ard-6e4b4af
--name 6000c290
```

响应:

NAME	IDPATH	SERIAL	PARTITIONCOUNT	MOUNTED	SIZE
sdh	/scsi-36000c29417	45000c	0	false	100GB

2. 使用 `replace` 命令将驱动器替换为已传递的序列号。如果 `-wait` 时间已过, 则命令将完成替换或失败。

```
kubectl astrads faileddrive replace --cluster arda-6e4b4af --name
6000c290 --replaceWith 45000c --wait
```

响应:

```
Drive replacement completed successfully
```



如果使用不适当的 `-replaceWith` 序列号执行 `kubectl astrad faileddrive replace`, 则会显示类似以下内容的错误:

```
kubectl astrads replacedrive replace --cluster astrads-cluster-f51b10a
--name 6000c2927 --replaceWith BAD_SERIAL_NUMBER
Drive 6000c2927 replacement started
Failed drive 6000c2927 has been set to use BAD_SERIAL_NUMBER as a
replacement
...
Drive replacement didn't complete within 25 seconds
Current status: {FailedDriveInfo:{InUse:false Present:true Name:scsi-
36000c2 FiretapUUID:444a5468 Serial:6000c Path:/scsi-36000c
FailureReason:AdminFailed Node:sti-b200-0214a.lab.netapp.com}
Cluster:astrads-cluster-f51b10a State:ReadyToReplace
Conditions:[{Message: "Replacement drive serial specified doesn't
exist", Reason: "DriveSelectionFailed", Status: False, Type:' Done'}]}
```

3. 要重新运行驱动器更换, 请使用 `-force` 和上一个命令:

```
kubectl astrads faileddrive replace --cluster astrads-cluster-f51b10a
--name 6000c2927 --replaceWith VALID_SERIAL_NUMBER --force
```

有关详细信息 ...

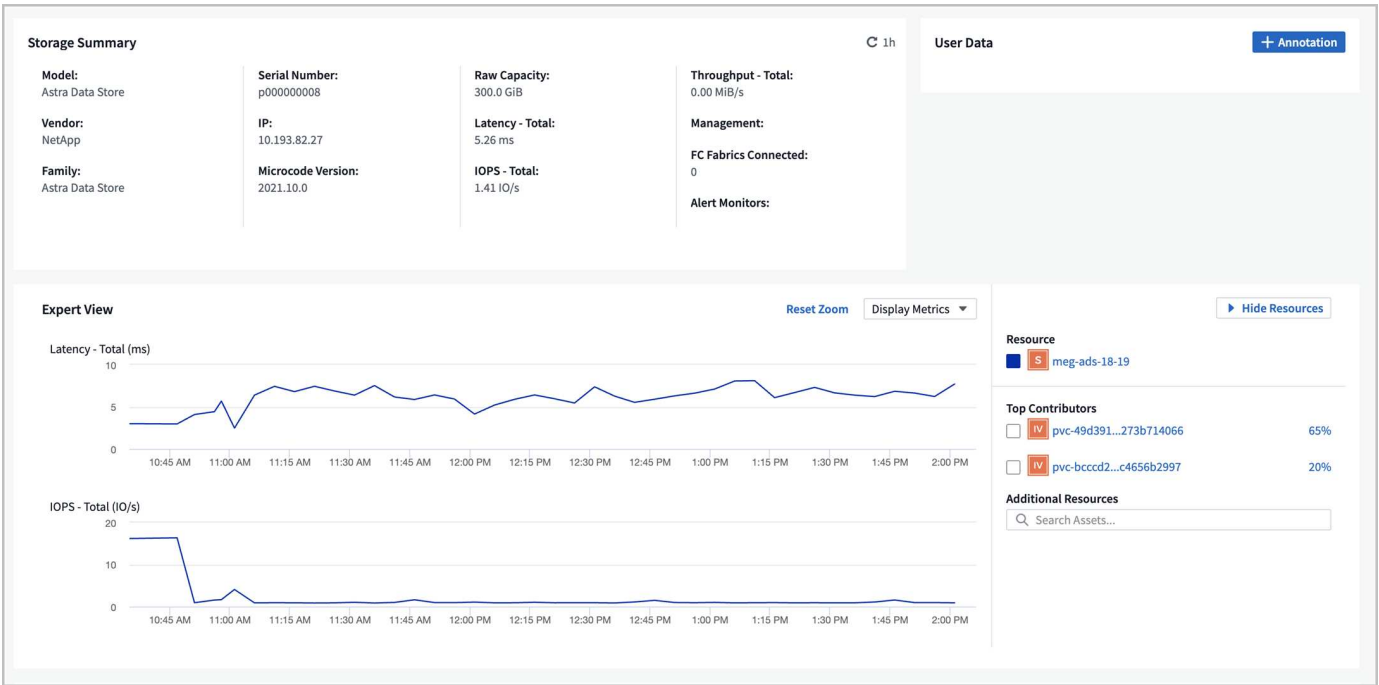
- ["使用kubectl命令管理Astra Data Store资源"](#)
- ["将节点添加到Astra控制中心的存储后端集群"](#)

## 监控Astra数据存储

### 使用 Cloud Insights 监控指标

您可以使用Cloud Insights 监控Astra数据存储指标。

以下是Cloud Insights 中显示的一些示例Astra数据存储指标：



您还可以使用显示在Astra数据存储中生成的指标列表 [\[Open Metrics API help\]](#)。

您可以完成以下任务：

- [\[Complete Cloud Insights connection prerequisite tasks\]](#)
- [\[Acquisition Unit storage\]](#)
- [\[Download and run the installation script\]](#)
- [\[Edit the Cloud Insights connection\]](#)
- [\[Disconnect from Cloud Insights\]](#)

### 完成 Cloud Insights 连接前提条件任务

在将 Astra 数据存储与 Cloud Insights 连接之前，您需要完成以下任务：

- ["安装 Astra 数据存储监控操作员"](#) 这是Astra Data Store安装说明的一部分。

- "安装 [kubectl-astrad 二进制文件](#)" 这是Astra Data Store安装说明的一部分。
- "创建 [Cloud Insights 帐户](#)"。
- 确保以下命令可用： `jk` ， `curl` ， `grep` 和 `jq`

收集以下信息：

- 对以下类别具有读 / 写权限的 \* Cloud Insights API 访问令牌 \*：采集单元，数据收集，数据载入和日志载入。此选项将用于读 / 写操作，设置采集单元以及设置数据载入过程。
- \* Kubernetes API 服务器 IP 地址和端口 \*。用于监控Astra数据存储集群。
- \* Kubernetes API 令牌 \*。此选项用于调用 Kubernetes API。
- \* 永久性卷配置 \*。有关如何配置永久性卷的信息。

### 采集单元存储

采集单元需要三个永久性卷来存储安装文件，配置数据和日志。监控操作员使用默认存储类创建永久性卷请求。在运行安装程序脚本时，您可以使用 `-s` 选项指定其他存储类名称。

如果您的 Kubernetes 集群没有存储配置程序（例如 NetApp Trident），您可以在运行安装程序脚本时使用 `-r` 选项提供本地文件系统路径。设置 `-r` 选项后，安装程序脚本会在提供的目录中创建三个永久性卷。此目录至少需要 150 GB 的可用空间。

### 下载并运行安装脚本

Cloud Insights 提供了Bash脚本、用于通过监控操作员启用Astra数据存储监控。安装脚本将安装一个采集单元、其中包含Astra Data Store收集器和一个Fluent Bit Agent。

下载 Cloud Insights 租户域名和选定的 Cloud Insights API 访问令牌后，此令牌将嵌入安装程序脚本中。

然后，指标将按如下所示发送：

- Cloud Insights 采集单元将指标发送到Cloud Insights 数据湖。
- Fluent Bit 会将日志发送到日志载入服务。

### 显示安装程序脚本帮助

安装程序脚本的完整帮助文本如下所示：

显示安装程序脚本帮助文本：

```
./cloudinsights-ads-monitoring.sh -h
```

响应：

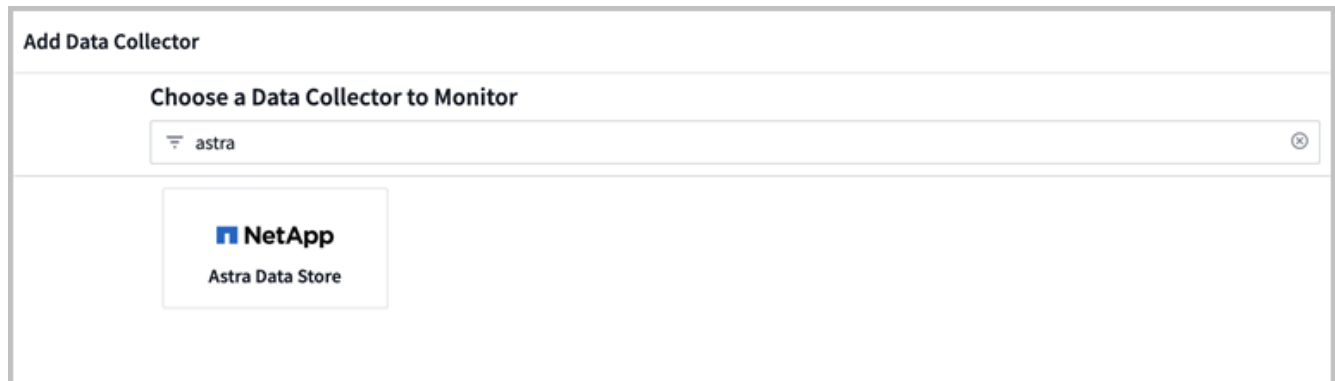
```

USAGE: cloudinsights-ads-monitoring.sh [OPTIONS]
Configure monitoring of Astra Data Store by Cloud Insights.
OPTIONS:
  -h                                Display this help message.
  -d ci_domain_name                 Cloud Insights tenant domain name.
  -i kubernetes_ip                 Kubernetes API server IP address.
  -k ci_api_key                     Cloud Insights API Access Token.
  -n namespace                       Namespace for monitoring components. (default:
netapp-monitoring)
  -p kubernetes_port                Kubernetes API server port. (default: 6443)
  -r root_pv_dir                    Create 3 Persistent Volumes in this directory
for the Acquisition Unit.
                                   Only specify this option if there is no Storage
Provisioner installed and the PVs do not already exist.
  -s storage_class                  Storage Class name for provisioning Acquisition
Unit PVs. If not specified, the default storage class will be used.
  -t kubernetes_token               Kubernetes API server token.

```

#### 运行安装脚本


1. 如果您还没有 Cloud Insights 帐户，请创建一个。
2. 登录到 Cloud Insights 。
3. 从 Cloud Insights 菜单中，单击 \* 管理 \* > \* 数据收集器 \* 。
4. 单击 \* + Data Collector\* 以添加新收集器。




5. 单击 \* Astra Data Store\* 图块。
6. 选择正确的 Cloud Insights API 访问令牌或创建新令牌。
7. 按照说明下载安装程序脚本，更新权限并运行此脚本。

此脚本包含您的 Cloud Insights 租户 URL 和选定的 Cloud Insights API 访问令牌。






Select a Data Collector



Configure Collector




**NetApp**  
Astra Data Store

## Configure Collector

Configure Kubernetes Operator to monitor NetApp Astra Data Store (ADS).

**What Operating System or Platform Are You Using?**


Kubernetes

**Select existing API Access Token or create a new one**

default\_ads\_api\_key1 (...d0gHof)

[+ API Access Token](#)

[Production Best Practices ?](#)

**Configure Astra Data Store** [Need Help?](#)

- 1

The commands *awk*, *curl*, *grep*, *jq*, *kubectl* and the *kubectl-astads* plugin must be installed where the installer script is run. You will need the Kubernetes API server IP address and a Kubernetes API token to run this install script. See the documentation if you need help finding this information.
- 2

Copy Installer Script

☐ Reveal Installer Script

```
#!/usr/bin/env bash
SCRIPT=`basename $0`

CI_DOMAIN_NAME="f49uaky.gstabler-ads.cloudinsights-dev.netapp.com"
CI_API_KEY="eyJraWQ1OiI5OTk5IiwidHlwIjoI5ldUIiwiaWxnIjoI5FMzODQifQ.eyJjcVhdG9yTG9naW41OiJhZG1pb2IiImRpc3BsYXl0YW1lIjoIZGVmYXVsdF9hZHNfYXBPX2tleTEgKG9uIGJlaGFsZiBvZiBhZG1pb2I"

```
- 3

Copy the above installer script and save it as *cloudinsights-ads-monitoring.sh*
- 4

Copy Permissions Command

☐ Reveal Permissions Command

```
chmod +x cloudinsights-ads-monitoring.sh

```
- 5

Paste the permissions command in a terminal to enable execute permissions on the installer script.
- 6

Copy Install Command

☐ Reveal Install Command

```
./cloudinsights-ads-monitoring.sh -i <KUBERNETES_IP> -t <KUBERNETES_TOKEN>

```
- 7

Paste the install command in a terminal, replace the placeholders with the correct values for your environment, and run the command. It will take several minutes to complete. The script will use the default namespace 'netapp-monitoring'. Additional options are available for customized environments. The default configuration for kubectl should point to the kubernetes cluster to be monitored.
- 8

Complete Setup

8. 完成脚本后，单击 \* 完成设置 \*。

安装脚本完成后，Astra Data Store 收集器将显示在 Datasources 列表中。



如果脚本因错误而退出，您可以稍后在解决错误后再次运行它。如果您的环境不使用默认设置，则该脚本支持其他参数，例如监控操作员命名空间和 Kubernetes API 服务器端口。使用 `./cloudinsights-ads-monitoring.sh -h` 中的 `-h` 选项查看使用情况和帮助文本。

如果配置成功，安装脚本将生成如下输出：

```
Configuring Cloud Insights monitoring for Astra Data Store . . .
Configuring monitoring namespace
...
Configuring output sink and Fluent Bit plugins
Configuring Acquisition Unit
...
Acquisition Unit has been installed successfully.
Configuring Astra Data Store data collector
Astra Data Store collector data '<CLUSTER_NAME>' created
Configuration done!
```

#### 代理 **CR** 示例

以下是运行安装程序脚本后 `monitoring-NetApp` 代理 **CR** 的外观示例。

```

spec:
  au:
    isEnabled: true
    storageClassName: auto-sc
  cluster-name: meg-ads-21-22-29-30
  docker-repo: docker.repo.eng.netapp.com/global/astra
  fluent-bit:
    - name: ads-tail
      outputs:
        - sink: ADS_STDOUT
      substitutions:
        - key: TAG
          value: firetapems
        - key: LOG_FILE
          values:
            - /var/log/firetap/*/ems/ems
            - /var/log/firetap/ems/*/ems/ems
        - key: ADS_CLUSTER_NAME
          value: meg-ads-21-22-28-29-30
    - name: agent
    - name: ads-tail-ci
      outputs:
        - sink: CI
      substitutions:
        - key: TAG
          value: netapp.ads
        - key: LOG_FILE
          values:
            - /var/log/firetap/*/ems/ems
            - /var/log/firetap/ems/*/ems/ems
        - key: ADS_CLUSTER_NAME
          value: meg-ads-21-22-28-29-30
  output-sink:
    - api-key: abcd
      domain-name: bz19ngz.gst-adsdemo.ci-dev.netapp.com
      name: CI
  serviceAccount: sa-netapp-monitoring
  status:
    au-pod-status: UP
    au-uuid: eddeccc6-3aa3-4dd2-a98c-220085fae6a9

```

## 编辑 Cloud Insights 连接

您可以稍后编辑 Kubernetes API 令牌或 Cloud Insights API 访问令牌：

- 如果要更新 Kubernetes API 令牌，应从 Cloud Insights UI 编辑 Astra 数据存储收集器。
- 如果要更新用于遥测和日志的 Cloud Insights API 访问令牌，应使用 kubectl 命令编辑监控操作员 CR 。

#### 更新 **Kubernetes API** 令牌

1. 登录到 Cloud Insights 。
2. 选择 \* 管理 \* > \* 数据收集器 \* 以访问数据收集器页面。
3. 找到 Astra Data Store 集群的条目。
4. 单击页面右侧的菜单，然后选择 \* 编辑 \* 。
5. 使用新值更新 Kubernetes API Token 字段。
6. 选择 \* 保存收集器 \* 。

#### 更新 **Cloud Insights API** 访问令牌

1. 登录到 Cloud Insights 。
2. 选择 \* 管理 \* > \* API 访问 \* 并单击 \* + API 访问令牌 \* ，创建新的 Cloud Insights API 访问令牌。
3. 编辑代理 CR ：

```
kubectl --namespace netapp-monitoring edit agent agent-monitoring-netapp
```

4. 找到 output-sink 部分，找到名为 CI 的条目。
5. 对于标签 api-key ，请将当前值替换为新的 Cloud Insights API 访问令牌。

此部分如下所示：

```
output-sink:
- api-key: <api key value>
  domain-name: <tenant url>
  name: CI
```

6. 保存并退出编辑器窗口。

监控操作员将更新 Fluent Bit 以使用新的 Cloud Insights API 访问令牌。

#### 断开与 **Cloud Insights** 的连接

要断开与 Cloud Insights 的连接，您需要先从 Cloud Insights UI 中删除 Astra 数据存储收集器。完成此操作后，您可以从监控操作员中删除采集单元、Telegraf (如果已配置) 和 Fluent Bit 配置。

#### 删除 **Astra** 数据存储收集器

1. 登录到 Cloud Insights 。
2. 选择 \* 管理 \* > \* 数据收集器 \* 以访问数据收集器页面。

3. 找到 Astra Data Store 集群的条目。
4. 选择屏幕右侧的菜单，然后选择 \* 删除 \*。
5. 单击确认页面上的 \* 删除 \*。

删除采集单元、Telegraf (如果已配置)和Fluent Bit

1. 编辑代理 CR：

```
kubectl --namespace netapp-monitoring edit agent agent-monitoring-netapp
```

2. 找到 au 部分并将 isenabled 设置为 false
3. 找到 flual-bit 部分，然后删除名为 ads-tail -CI 的插件。如果没有其他插件，您可以删除 flual-bit 部分。
4. 如果配置了Telegraf、请找到`telegraf`部分、然后删除名为`ads-open-metric`的插件。如果没有其他插件，您可以删除 电报 部分。
5. 找到 output-sink 部分，然后卸下名为 CI 的接收器。
6. 保存并退出编辑器窗口。

监控操作员将更新Telegraf (如果已配置)和Fluent Bit配置、并删除采集单元POD。

7. 如果您使用本地目录作为采集单元 PV ，而不是存储配置程序，请删除这些 PV：

```
kubectl delete pv au-lib au-log au-pv
```

然后，删除运行采集单元的节点上的实际目录。

8. 删除采集单元 POD 后，您可以从 Cloud Insights 中删除采集单元。
  - a. 在 Cloud Insights 菜单中，选择 \* 管理 \* > \* 数据收集器 \*。
  - b. 单击 \* 采集单元 \* 选项卡。
  - c. 单击采集单元 POD 旁边的菜单。
  - d. 选择 \* 删除 \*。

监控操作员将更新Telegraf (如果已配置)和Fluent Bit配置、并删除采集单元。

打开指标 **API** 帮助

下面列出了可用于从Astra数据存储收集指标的API。

- "help" 行说明了指标。
- "type" 行表示指标是量表还是计数器。

```
# HELP astrads_cluster_capacity_logical_percent Percentage cluster logical
```

```

capacity that is used (0-100)
# TYPE astrads_cluster_capacity_logical_percent gauge
# HELP astrads_cluster_capacity_max_logical Max Logical capacity of the
cluster in bytes
# TYPE astrads_cluster_capacity_max_logical gauge
# HELP astrads_cluster_capacity_max_physical The sum of the space in the
cluster in bytes for storing data after provisioning efficiencies, data
reduction algorithms and replication schemes are applied
# TYPE astrads_cluster_capacity_max_physical gauge
# HELP astrads_cluster_capacity_ops The IO operations capacity of the
cluster
# TYPE astrads_cluster_capacity_ops gauge
# HELP astrads_cluster_capacity_physical_percent The percentage of cluster
physical capacity that is used (0-100)
# TYPE astrads_cluster_capacity_physical_percent gauge
# HELP astrads_cluster_capacity_used_logical The sum of the bytes of data
in all volumes in the cluster before provisioning efficiencies, data
reduction algorithms and replication schemes are applied
# TYPE astrads_cluster_capacity_used_logical gauge
# HELP astrads_cluster_capacity_used_physical Used Physical capacity of a
cluster in bytes
# TYPE astrads_cluster_capacity_used_physical gauge
# HELP astrads_cluster_other_latency The sum of the accumulated latency in
seconds for other IO operations of all the volumes in a cluster. Divide by
astrads_cluster_other_ops to get the average latency per other operation
# TYPE astrads_cluster_other_latency counter
# HELP astrads_cluster_other_ops The sum of the other IO operations of all
the volumes in a cluster
# TYPE astrads_cluster_other_ops counter
# HELP astrads_cluster_read_latency The sum of the accumulated latency in
seconds of read IO operations of all the volumes in a cluster. Divide by
astrads_cluster_read_ops to get the average latency per read operation
# TYPE astrads_cluster_read_latency counter
# HELP astrads_cluster_read_ops The sum of the read IO operations of all
the volumes in a cluster
# TYPE astrads_cluster_read_ops counter
# HELP astrads_cluster_read_throughput The sum of the read throughput of
all the volumes in a cluster in bytes
# TYPE astrads_cluster_read_throughput counter
# HELP astrads_cluster_storage_efficiency Efficacy of data reduction
technologies. (logical used / physical used)
# TYPE astrads_cluster_storage_efficiency gauge
# HELP astrads_cluster_total_latency The sum of the accumulated latency in
seconds of all IO operations of all the volumes in a cluster. Divide by
astrads_cluster_total_ops to get average latency per operation
# TYPE astrads_cluster_total_latency counter

```

```

# HELP astrads_cluster_total_ops The sum of the IO operations of all the
volumes in a cluster
# TYPE astrads_cluster_total_ops counter
# HELP astrads_cluster_total_throughput The sum of the read and write
throughput of all the volumes in a cluster in bytes
# TYPE astrads_cluster_total_throughput counter
# HELP astrads_cluster_utilization_factor The ratio of the current cluster
IO operations based on recent IO sizes to the cluster iops capacity. (0.0
- 1.0)
# TYPE astrads_cluster_utilization_factor gauge
# HELP astrads_cluster_volume_used The sum of used capacity of all the
volumes in a cluster in bytes
# TYPE astrads_cluster_volume_used gauge
# HELP astrads_cluster_write_latency The sum of the accumulated latency in
seconds of write IO operations of all the volumes in a cluster. Divide by
astrads_cluster_write_ops to get the average latency per write operation
# TYPE astrads_cluster_write_latency counter
# HELP astrads_cluster_write_ops The sum of the write IO operations of all
the volumes in a cluster
# TYPE astrads_cluster_write_ops counter
# HELP astrads_cluster_write_throughput The sum of the write throughput of
all the volumes in a cluster in bytes
# TYPE astrads_cluster_write_throughput counter
# HELP astrads_disk_base_seconds Base for busy, pending and queued.
Seconds since collection began
# TYPE astrads_disk_base_seconds counter
# HELP astrads_disk_busy Seconds the disk was busy. 100 *
(astrads_disk_busy / astrads_disk_base_seconds) = percent busy (0-100)
# TYPE astrads_disk_busy counter
# HELP astrads_disk_capacity Raw Capacity of a disk in bytes
# TYPE astrads_disk_capacity gauge
# HELP astrads_disk_io_pending Summation of the count of pending io
operations for a disk times time. Divide by astrads_disk_base_seconds to
get the average pending operation count
# TYPE astrads_disk_io_pending counter
# HELP astrads_disk_io_queued Summation of the count of queued io
operations for a disk times time. Divide by astrads_disk_base_seconds to
get the average queued operations count
# TYPE astrads_disk_io_queued counter
# HELP astrads_disk_read_latency Total accumulated latency in seconds for
disk reads. Divide by astrads_disk_read_ops to get the average latency per
read operation
# TYPE astrads_disk_read_latency counter
# HELP astrads_disk_read_ops Total number of read operations for a disk
# TYPE astrads_disk_read_ops counter
# HELP astrads_disk_read_throughput Total bytes read from a disk

```

```

# TYPE astrads_disk_read_throughput counter
# HELP astrads_disk_write_latency Total accumulated latency in seconds for
disk writes. Divide by astrads_disk_write_ops to get the average latency
per write operation
# TYPE astrads_disk_write_latency counter
# HELP astrads_disk_write_ops Total number of write operations for a disk
# TYPE astrads_disk_write_ops counter
# HELP astrads_disk_write_throughput Total bytes written to a disk
# TYPE astrads_disk_write_throughput counter
# HELP astrads_value_scrape_duration Duration to scrape values
# TYPE astrads_value_scrape_duration gauge
# HELP astrads_volume_capacity_available The minimum of the available
capacity of a volume and the available capacity of the cluster in bytes
# TYPE astrads_volume_capacity_available gauge
# HELP astrads_volume_capacity_available_logical Logical available
capacity of a volume in bytes
# TYPE astrads_volume_capacity_available_logical gauge
# HELP astrads_volume_capacity_percent Percentage of volume capacity
available (0-100). (capacity available / provisioned) * 100
# TYPE astrads_volume_capacity_percent gauge
# HELP astrads_volume_capacity_provisioned Provisioned capacity of a
volume in bytes after setting aside the snapshot reserve. (size - snapshot
reserve = provisioned)
# TYPE astrads_volume_capacity_provisioned gauge
# HELP astrads_volume_capacity_size Total capacity of a volume in bytes
# TYPE astrads_volume_capacity_size gauge
# HELP astrads_volume_capacity_snapshot_reserve_percent Snapshot reserve
percentage of a volume (0-100)
# TYPE astrads_volume_capacity_snapshot_reserve_percent gauge
# HELP astrads_volume_capacity_snapshot_used The amount of volume snapshot
data that is not in the active file system in bytes
# TYPE astrads_volume_capacity_snapshot_used gauge
# HELP astrads_volume_capacity_used Used capacity of a volume in bytes.
This is bytes in the active filesystem unless snapshots are consuming more
than the snapshot reserve. (bytes in the active file system + MAX(0,
snapshot_used-(snapshot_reserve_percent/100*size))
# TYPE astrads_volume_capacity_used gauge
# HELP astrads_volume_other_latency Total accumulated latency in seconds
for operations on a volume that are neither read or write. Divide by
astrads_volume_other_ops to get the average latency per other operation
# TYPE astrads_volume_other_latency counter
# HELP astrads_volume_other_ops Total number of operations for a volume
that are neither read or write
# TYPE astrads_volume_other_ops counter
# HELP astrads_volume_read_latency Total accumulated read latency in
seconds for a volume. Divide by astrads_volume_read_ops to get the average

```



```

latency per read operation
# TYPE astrads_volume_read_latency counter
# HELP astrads_volume_read_ops Total number of read operations for a
volume
# TYPE astrads_volume_read_ops counter
# HELP astrads_volume_read_throughput Total read throughput for a volume
in bytes
# TYPE astrads_volume_read_throughput counter
# HELP astrads_volume_total_latency Total accumulated latency in seconds
for all operations on a volume. Divide by astrads_volume_total_ops to get
the average latency per operation
# TYPE astrads_volume_total_latency counter
# HELP astrads_volume_total_ops Total number of operations for a volume
# TYPE astrads_volume_total_ops counter
# HELP astrads_volume_total_throughput Total throughput for a volume in
bytes
# TYPE astrads_volume_total_throughput counter
# HELP astrads_volume_write_latency Total accumulated write latency in
seconds for volume. Divide by astrads_volume_write_ops to get the average
latency per write operation
# TYPE astrads_volume_write_latency counter
# HELP astrads_volume_write_ops Total number of write operations for a
volume
# TYPE astrads_volume_write_ops counter
# HELP astrads_volume_write_throughput Total write throughput for a volume
in bytes
# TYPE astrads_volume_write_throughput counter

```

## 使用 Prometheus 和 Grafana 监控指标

您可以使用Prometheus和Grafana监控Astra数据存储指标。您可以将Prometheus配置为从Astra Data Store Kubernetes集群指标端点收集指标、并且可以使用Grafana将指标数据可视化。

您需要什么？ **#8217** ； 将需要什么

- 确保已在Astra数据存储集群或可与Astra数据存储集群通信的其他集群上下载并安装Prometheus和Grafana软件包。按照官方文档中的说明安装每个工具：
  - ["安装 Prometheus"](#)
  - ["安装 Grafana"](#)
- Prometheus和Grafana需要能够与Astra Data Store Kubernetes集群进行通信。如果Astra Data Store集群上未安装Prometheus和Grafana、则需要确保它们可以与Astra Data Store集群上运行的指标服务进行通信。

### 配置 Prometheus

Astra数据存储在Kubernetes集群中的TCP端口9341上公开指标服务。您需要配置 Prometheus 以从此服务收集指标。

## 步骤

1. 为您的 Prometheus 安装编辑 `Prometheus.yml` 配置文件。
2. 添加指向Astra Data Store服务名称及其端口的服务目标。例如：

```
scrape_configs:
  static_configs:
    - targets: ['astrads-metrics-service.astrads-system:9341']
```

3. 启动 Prometheus 服务。

## 配置 Grafana

您可以将 Grafana 配置为显示 Prometheus 收集的指标。

## 步骤

1. 为您的 Grafana 安装编辑 `datasources.yaml` 配置文件。
2. 将 Prometheus 添加为数据源。例如：

```
apiVersion: 1

datasources:
  - name: astradatastore-prometheus
    type: prometheus
    access: proxy
    url: http://localhost:9090
    jsonData:
      manageAlerts: false
```

3. 启动 Grafana 服务。
4. 按照 Grafana 文档中的说明进行操作 ["开始使用"](#)。

## 导入 Grafana 信息板模板

为安装Astra数据存储而下载的捆绑包文件包含Grafana信息板模板文件、您可以从Grafana中导入这些文件。这些信息板模板可帮助您查看Astra数据存储提供的指标类型以及查看方式。

## 步骤

1. 打开Astra Data Store `.tar.gz` 软件包。
2. 打开 `manifests` 目录。
3. 提取 `grafana_cluster.json` 和 `grafana_volume.json` 文件。
4. 使用 Grafana Web UI ， ["将信息板模板文件导入到 Grafana 中"](#)。

## 配置和监控事件日志

要监控事件管理系统（EMS）日志，您可以执行以下高级任务：

- [\[Configure monitoring in the Astra Data Store cluster custom resource \(CR\)\]](#)
- [\[Set up Cloud Insights\]](#)
- [\[Stream event logs to Elastic\]](#)。

在**Astra Data Store**集群自定义资源(CR)中配置监控

如果尚未在Astra Data Store集群CR上配置监控选项、您可以使用`astrad` extensions进行设置。

输入 ...

```
kubectl astrads monitoring setup -n <NAMESPACE OF AGENT INSTALLED> -r  
<DOCKER REPO TO FIND FLUENT/TELEGRAF ETC IMAGES>
```

其中：

- 已安装代理的命名空间：输入监控代理的命名空间，这是监控操作员的 monitoring-NetApp CR 的默认名称。
- `-r` 是可选的，用于设置 Fluent 或 Telegraf 映像所在的 Docker 注册表。默认情况下，此路径设置为 `docker.repo.eng.netapp.com/global/astra`，您可以进行更改。

### 设置 Cloud Insights

要查看日志，设置 Cloud Insights 是可选的；但是，使用 Cloud Insights 查看数据很有帮助。请参见 ["如何设置 NetApp Cloud Insights"](#) 用于Astra数据存储。

将事件日志流式传输到 **Elastic**

要将 EMS 事件和其他 POD 日志流式传输到 Elastic 等第三方端点，请使用 `astrad` 扩展。

输入 ...

```
kubectl astrads monitoring --host <ELASTIC HOST NAME> --port <ELASTIC HOST  
PORT> es
```



弹性主机名可以是 IP 地址。

## 安全的Astra数据存储

### 管理安全证书

Astra数据存储在集群的软件组件之间使用相互传输层安全(MTLS)加密。每个Astra Data Store集群都有一个自签

名根CA证书(astrads-cert)和一个中间CA证书(astrads-cert -<cluster\_name>)。这些证书由Astra数据存储操作员管理；操作员会在每个证书到期日期前7天自动续订这些证书。您也可以手动撤消这些证书。

## 撤消证书

如果Astra数据存储控制器、节点或CA证书受到影响、您可以通过删除其MTLS密钥来撤消该证书。执行此操作时、Astra数据存储操作员会自动颁发一个新证书。您可以随时撤消Astra数据存储证书。



如果您撤消CA证书、则此操作将撤消由该CA签名的任何证书。

## 步骤

1. 登录到Astra Data Store集群中的控制器节点。
2. 列出系统上的现有证书。例如：

```
kubectl get secrets -n astrads-system | grep astrads-cert
```

输出应类似于以下内容：

```
astrads-cert-astrads-cluster-controller
kubernetes.io/tls      4      6d6h
astrads-cert-astrads-cluster-f23d158
kubernetes.io/tls      4      6d6h
astrads-cert-astrads-ds-dms-astrads-cluster-f23d158
kubernetes.io/tls      4      6d6h
astrads-cert-astrads-ds-support-astrads-cluster-f23d158
kubernetes.io/tls      4      6d6h
astrads-cert-astrads-support-astrads-cluster-f23d158
kubernetes.io/tls      4      6d6h
astrads-cert-root
kubernetes.io/tls      4      6d6h
astrads-cert-sti-net-com
kubernetes.io/tls      5      6d6h
```

3. 在输出中、记下需要撤消的证书的名称。
4. 使用`kubectl`实用程序撤消证书、并将`certificate\_name`替换为证书的名称。例如：

```
kubectl delete secret CERTIFICATE_NAME -n astrads-system
```

现有证书将被撤消、并自动生成一个新证书。

## 管理外部密钥

您可以使用一个或多个外部密钥管理服务器来保护集群用于访问加密数据的密钥。外部密钥管理服务器是存储环

境中的第三方系统，可使用密钥管理互操作性协议（Key Management Interoperability Protocol，KMIP）为节点提供密钥。



默认情况下、在创建Astra Data Store集群时、Astra Data Store会通过内部密钥提供程序启用空闲软件加密(软件加密)。

管理密钥涉及以下自定义资源定义(CRD)：

- **\* AstraDSKeyProvider\***：配置外部KMIP服务器、该服务器可以是服务器集群。
- **\* AstraDSSEARKeyrotate\***：从密钥提供程序获取新的密钥加密密钥并将其提供给Astra数据存储。

您可以执行以下与外部密钥管理相关的任务：

- [\[Set up external key management\]](#)
- [\[Check the software encryption at rest status\]](#)
- [\[Change external to internal key management\]](#)
- [\[Rotate keys for security\]](#)

## 设置外部密钥管理

在Astra Data Store中设置外部密钥管理时、可以使用`kubectl astrad`命令。

您需要在集群或KMIP服务器上获得SSL证书、以便可以设置外部密钥、例如使用OpenSSL。

### 步骤

1. 为密钥提供程序客户端准备证书。包括客户端证书、客户端专用密钥和信任CA捆绑包。



您需要在集群或KMIP服务器上准备SSL证书、以便设置外部密钥、例如、使用OpenSSL。

2. 登录到Astra Data Store集群中的一个节点。
3. 输入以下kubectl扩展命令、为Astra Data Store集群配置密钥提供程序：

```
kubectl-astrads key-provider certs --key key.pem
--client-cert client_cert.pem --ca-cert server_ca.pem
--hostnames=<kmip_server_ip> <key_provider_cr_name>
--namespace astrads-system --cluster <ads_cluster_name>
```

以下示例将为ADS集群"astrads-cluster-f23d158"配置一个名为"hashicorp"的外部密钥提供程序。

```
kubectl-astrads key-provider certs --key key.pem
--client-cert client_cert.pem --ca-cert server_ca.pem
--hostnames=10.235.nnn.nnn hashicorp
--namespace astrads-system --cluster astrads-cluster-f23d158
```

1. 将Astra数据存储集群配置为通过AstraDSCluster CR对sear使用外部密钥管理器。显示帮助。

```
kubectl-astrads clusters sears -h
```

响应：

```
Configure SEARS in AstraDS cluster
```

```
Usage:
```

```
astrads clusters sears [flags]
```

```
Flags:
```

```
-d, --duration string    Duration for key rotation (default "2160h")  
-h, --help               help for sears
```

```
Global Flags:
```

```
--ads-cluster-name string      Name of the ADS Cluster  
--ads-cluster-namespace string Namespace of the ADS Cluster  
...
```

以下命令会将Astra Data Store集群配置为使用`AstraDSKeyProvider hashicorp`作为sear的密钥管理器。此命令还会使用密钥轮换时间、其默认值为90天(2160小时)。

```
kubectl-astrads clusters sears -d 500h hashicorp  
--ads-cluster-name=astrads-cluster-f23d158  
--ads-cluster-namespace=astrads-system
```

## 检查软件空闲加密状态

您可以检查空闲软件加密的配置。

### 步骤

1. 检查AstraDSCluster CR。

```

Name:          astrads-cluster-f23d158
Namespace:     astrads-system
Labels:        <none>
Annotations:   <none>
API Version:   astrads.netapp.io/v1beta1
Kind:          AstraDSCluster
...
Spec:
...
  Software Encryption At Rest:
    Ads Key Provider:      hashicorp
    Key Rotation Period:   500h0m0s
...
Status:
...
  Software Encryption At Rest Status:
    Key Active Time:       2022-05-16T15:53:47Z
    Key Provider Name:     hashicorp
    Key Provider UUID:     ccfc2b0b-dd98-5ca4-b778-99debef83550
    Key UUID:              nnnnnnnn-nnnn-nnnn-nnnn-nnnnnnnnnnnn

```

## 将外部密钥管理更改为内部密钥管理

如果您当前使用的是外部密钥管理器，则可以将其更改为内部密钥管理器。

### 步骤

1. 通过删除SoftwareEncryptionAtRest配置来更改AstraDSCluster CR。
2. (可选)删除先前的AstraDSKeyProvider及其关联密钥。



不会自动删除先前的密钥提供程序和密钥。

## 为安全起见、请轮换密钥

密钥轮换可增强安全性。默认情况下，Astra数据存储每90天自动轮换一次密钥。您可以更改默认设置。此外，您还可以根据需要轮换按键。

### 配置自动密钥轮换

1. 更新CRD中的AstraDSSEARKeyrotate参数。

```

kubectl patch astradscluster astrads-cluster-f23d158
-n astrads-system
--type=merge -p '{"spec": {"softwareEncryptionAtRest": {
"keyRotationPeriod": "3000h"}}}'

```

## 配置按需密钥轮换

1. 创建AstraDSSEARKeyrotateRequest CR以轮换密钥。

```
cat << EOF | kubectl apply -f -
apiVersion: astrads.netapp.io/v1beta1
kind: AstraDSSEARKeyRotateRequest
metadata:
  name: manual
  namespace: astrads-system
spec:
  cluster: astrads-cluster-f23d158
EOF
```

## 更新Astra Data Store许可证

您可以更新已安装的Astra数据存储评估许可证、以延长评估期限。您可以使用以下三种方法之一更新许可证：

- 要使用Astra控制中心更新Astra数据存储许可证、请参见 ["更新存储后端许可证"](#)。
- 要使用Astra VMware插件更新Astra Data Store许可证、请参见 ["使用VMware管理Astra数据存储"](#)。
- 要使用命令行更新Astra Data Store许可证、请参见 [\[Update the Astra Data Store license using the command line\]](#)。

### 使用命令行更新Astra Data Store许可证

您可以使用`kubectl`实用程序更新Astra Data Store许可证。

#### 步骤

1. 应用从NetApp获得的替代NetApp许可证文件(NLF)。在运行命令之前、输入集群的名称(<Astra-Data-Store-cluster-name>)和许可证文件的路径(<file\_path/file.txt>)：

```
kubectl astrads license add --license-file-path <file_path/file.txt>
--ads-cluster-name <Astra-Data-Store-cluster-name> -n astrads-system
```

2. 验证是否已添加此许可证：

```
kubectl astrads license list
```

您应看到类似于以下内容的响应：



NAME	ADSCUSTER	VALID	PRODUCT
EVALUATION	ENDDATE	VALIDATED	
p100000006	astrads-example-cluster	true	Astra Data Store
2023-01-23	2022-04-04T14:38:54Z		true

## 升级Astra数据存储

您可以升级Astra数据存储以利用最新功能和修复。您可以使用Astra Data Store `kubecti`扩展来升级Astra Data Store。

### 使用kubecti升级Astra数据存储

您可以使用Astra Data Store `kubecti`扩展来升级Astra Data Store。

#### 下载Astra Data Store软件包并提取映像

##### 步骤

1. 登录到 ["NetApp 支持站点"](#) 并下载Astra数据存储捆绑包(Astra\_Data\_Store\_2022.05.01.tar)。
2. (可选) 使用以下命令验证捆绑包的签名：

```
openssl dgst -sha256 -verify Astra_Data_Store_2022.05.01.pub -signature
Astra_Data_Store_2022.05.01.sig 2022.12.01_ads.tar
```

3. 创建目录：

```
mkdir Astra_Data_Store_2022.05.01
cd Astra_Data_Store_2022.05.01
```

4. 提取映像：

```
tar -vxf <path to tar file>/Astra_Data_Store_2022.05.01.tar
```



这些图像将提取到工作目录中创建的`astrads/images`目录中。

#### 将二进制文件和推送映像复制到本地注册表

##### 步骤

1. 将kubecti-astrad二进制文件从用于提取映像的目录复制到安装了Kubernetes kubecti二进制文件的标准路径(在以下示例中使用`/usr/bin/`作为路径)。kubecti-astrad是一个自定义kubecti扩展、用于安装和管理Astra数据存储集群。



使用`which kubect!`命令查找安装kubect!二进制文件的路径。

```
cp -p .astrads/bin/kubect!-astrads /usr/bin/.
```

2. 将Astra Data Store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 login [your_registry_path]
```

- b. 将环境变量设置为要推送Astra数据存储映像的注册表路径；例如、repo.company.com。

```
export REGISTRY=repo.company.com/astrads
```

- c. 运行以下脚本、将映像加载到Docker、标记映像并将其推送到本地注册表：

```
for astraImageFile in $(ls astrads/images/*.tar) ; do
    astraImage=$(docker load --input ${astraImageFile} | sed 's~Loaded
image(s): ~~')
    astraImageShort=`echo $astraImage | sed 's~.*/~~'`
    docker tag ${astraImage} ${REGISTRY}/${astraImageShort}
    docker push ${REGISTRY}/${astraImageShort}
done
sed -i 's~\[YOUR_REGISTRY\]~'\${REGISTRY}'~'
./astrads/manifests/*.yaml
```

## 执行升级

### 步骤

1. 将`astradsoperator.yaml`文件复制到本地目录：

```
cp /PATH/TO/FILE/astradsoperator.yaml ./
```

2. 升级操作员。将大写的参数替换为适用于您环境的信息：

```
kubect!-astrads upgrade ads-operator --repository-url REPOSITORY_URL
--operator-yaml astradsoperator.yaml
```

3. 开始升级Astra Data Store。将大写的参数替换为适用于您环境的信息：

```
kubect1-ast1ads upgrade ads-version --repository-url REPOSITORY_URL  
--ads-version-yaml ./astrads/manifests/ast1radsversion.yaml
```

此时将显示一条消息、通知您升级已启动、需要几分钟才能完成。

## 使用自动化脚本卸载Astra Data Store

要卸载Astra数据存储和控制平台、您需要删除工作负载、绑定、卷、导出策略、Astra数据存储集群、许可证、部署环境和Astra数据存储命名空间。

卸载可使用不同的方法完成：

- [\[Uninstall Astra Data Store with an automated script\]](#)
- [\[Uninstall Astra Data Store manually without a script\]](#)
- [\[Troubleshoot the Astra Data Store uninstall process\]](#)

## 使用自动化脚本卸载Astra Data Store

此过程使用自动脚本卸载Astra数据存储。

您需要什么？ #8217 ；将需要什么

- root 管理权限

Astra数据存储卸载过程将指导您完成以下高级步骤：

- [\[Remove existing workloads and bindings\]](#)
- [\[Uninstall Astra Data Store cluster\]](#)
- [\[Validate the removal of the astrads-system namespace\]](#)
- [\[Ensure containers are not running on worker nodes\]](#)
- [\[Delete OpenShift Container Platform resources\]](#)

删除现有工作负载和绑定

卸载Astra数据存储之前、必须先删除以下内容

- 使用Astra Data Store作为存储后端的所有应用程序工作负载
- 使用Astra数据存储作为后端的Trident绑定

这样可以确保 Kubernetes 环境保持干净，这在重新安装时非常重要。

卸载 **Astra Data Store** 集群

要卸载Astra Data Store、您可以使用从NetApp支持站点下载的Astra Data Store tar文件中的`uninstall.sh`脚本。

1. 在 manifests 目录中找到 uninstall.sh。

2. 运行以下 sed 命令：

```
sed -i -e 's~netappdsoperator.yaml~astradsoperator.yaml~' uninstall.sh
```

3. 运行以下脚本以指示要卸载的内容：

```
./uninstall.sh

You must run this script with an argument specifying what should be
uninstalled
To uninstall the ADS cluster run ./uninstall.sh cluster
To uninstall everything run ./uninstall all
```

4. 如果您只想卸载集群，请输入 `uninstall.sh <cluster>`

否则，如果要卸载所有内容，请输入 `uninstall.sh`



大多数情况下，您将卸载所有内容。如果您稍后要重新部署集群，则可能只需要卸载集群。

5. 在提示符处，确认要继续并输入 `erasedata`

响应：

```
./uninstall.sh all

Enter 'erasedata' to confirm you want proceed with the uninstall:
erasedata
+-----+
| Wed Feb  2 10:14:01 EST 2022                               |
| ADS cluster uninstall started                               |
+-----+
Deleting astradsvolumes
Deleted astradsvolumes
Deleting astradsexportpolicies
Deleted astradsexportpolicies
Deleting astradsvolumesnapshots
Deleted astradsvolumesnapshots
Deleting astradsclusters
Deleted astradsclusters
Deleting astradslicenses
Deleted astradslicenses
+-----+
```

```

| Wed Feb  2 10:15:18 EST 2022 |
| ADS cluster uninstall done   |
+-----+

+-----+
| Wed Feb  2 10:15:18 EST 2022 |
| ADS system uninstall started  |
+-----+

Removing astradsversion
astradsversion.astrads.netapp.io "astradsversion" deleted
Removed astradsversion
Removing daemonsets
daemonset.apps "astrads-ds-nodeinfo-astradsversion" deleted
Removed daemonsets
Removing deployments
deployment.apps "astrads-cluster-controller" deleted
deployment.apps "astrads-license-controller" deleted
deployment.apps "astrads-operator" deleted
Removed deployments
Removing all other AstraDS resources
namespace "astrads-system" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsautosupports.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradscLOUDsnapshots.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsclusters.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsexportpolicies.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsfaileddrives.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradslICENSES.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsnfsOPTIONS.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsnodeinfoes.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsnodemANAGEMENTS.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsqospolicies.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsversions.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsvOLUMEFILES.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io

```

```
"astradsvolumes.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsvolumesnapshots.astrads.netapp.io" deleted
role.rbac.authorization.k8s.io "astrads-astrads-system-admin-role"
deleted
role.rbac.authorization.k8s.io "astrads-astrads-system-reader-role"
deleted
role.rbac.authorization.k8s.io "astrads-astrads-system-writer-role"
deleted
role.rbac.authorization.k8s.io "astrads-leader-election-role" deleted
role.rbac.authorization.k8s.io "astrads-manager-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astrads-admin-
clusterrole" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astrads-reader-
clusterrole" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astrads-writer-
clusterrole" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsautosupport-
editor-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsautosupport-
viewer-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradscloudsnapshot-
editor-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradscloudsnapshot-
viewer-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradscluster-editor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradscluster-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsexportpolicy-
editor-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsexportpolicy-
viewer-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsfaileddrive-
editor-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsfaileddrive-
viewer-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradslicense-editor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradslicense-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsnfsoption-editor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsnfsoption-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsnodeinfo-editor-
```

```
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsnodeinfo-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsnodemanagement-
editor-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsnodemanagement-
viewer-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsqospolicy-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsversion-editor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsversion-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsvolume-editor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsvolume-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsvolumeeditor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsvolumeviewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsvolumesnapshot-
editor-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsvolumesnapshot-
viewer-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-manager-role" deleted
rolebinding.rbac.authorization.k8s.io "astrads-astrads-admin-
rolebinding" deleted
rolebinding.rbac.authorization.k8s.io "astrads-astrads-reader-
rolebinding" deleted
rolebinding.rbac.authorization.k8s.io "astrads-astrads-writer-
rolebinding" deleted
rolebinding.rbac.authorization.k8s.io "astrads-leader-election-
rolebinding" deleted
rolebinding.rbac.authorization.k8s.io "astrads-manager-rolebinding"
deleted
clusterrolebinding.rbac.authorization.k8s.io "astrads-astrads-admin-
rolebinding" deleted
clusterrolebinding.rbac.authorization.k8s.io "astrads-astrads-reader-
rolebinding" deleted
clusterrolebinding.rbac.authorization.k8s.io "astrads-astrads-writer-
rolebinding" deleted
clusterrolebinding.rbac.authorization.k8s.io "astrads-manager-
rolebinding" deleted
configmap "astrads-autosupport-cm" deleted
configmap "astrads-firetap-cm" deleted
```

```
configmap "astrads-kevents-asup" deleted
configmap "astrads-metrics-cm" deleted
secret "astrads-autosupport-certs" deleted
+-----+
| Wed Feb  2 10:16:36 EST 2022          |
| ADS system uninstall done            |
+-----+
```

验证是否删除了 **astrads-system** 命名空间

确保以下命令不返回任何结果：

```
kubectl get ns | grep astrads-system
```

确保工作节点上未运行容器

验证 **fireap** 或 **netwd** 等容器是否未在工作节点上运行。在每个节点上运行以下命令。

```
ssh <mynode1>
# runc list
```

删除 **OpenShift** 容器平台资源

如果您在Red Hat OpenShift容器平台(OCP)上安装了Astra数据存储、则可以卸载OCP安全上下文约束(SCC)和绑定资源。

OpenShift 使用安全上下文约束（ SCC ）来控制 Pod 可以执行的操作。

完成标准卸载过程后，请完成以下步骤。

1. 删除 SCC 资源：

```
oc delete -f ads_privileged_scc.yaml
```

2. 删除 rolebindings 资源：

```
oc delete -f oc_role_bindings.yaml
```



忽略这些步骤中的 " 未找到资源 " 错误。



## 手动卸载Astra数据存储、而不使用脚本

此过程无需脚本即可手动卸载Astra数据存储。

要在不使用自动脚本的情况下手动卸载Astra数据存储、您需要删除工作负载、绑定、卷、导出策略、集群、许可证、部署环境和Astra Data Store命名空间。

您需要什么？ **#8217** ；将需要什么

- root 管理权限

Astra数据存储卸载过程将指导您完成以下高级步骤：

- [\[Remove existing workloads and bindings\]](#)
- [\[Uninstall the Astra Data Store cluster and control plane\]](#)
- [\[Delete the license\]](#)
- [\[Delete the Astra Data Store installation\]](#)
- [\[Validate the removal of the astrads-system namespace\]](#)
- [\[Ensure containers are not running on worker nodes\]](#)
- [\[Delete OpenShift Container Platform resources\]](#)

删除现有工作负载和绑定

卸载Astra数据存储之前、必须先删除以下内容

- 使用Astra Data Store作为存储后端的所有应用程序工作负载
- 使用Astra数据存储作为后端的Trident绑定

这样可以确保 Kubernetes 环境保持干净，这在重新安装时非常重要。

卸载Astra数据存储集群和控制平面

按照以下步骤手动卸载Astra数据存储。

删除卷和导出策略

在删除集群之前、您应删除Astra Data Store卷和导出策略。



如果不先删除卷和导出策略、则集群删除过程将暂停、直到删除Astra Data Store volumes对象为止。在开始删除集群之前删除这些项会更高效。

步骤

1. 删除卷：

```
~% kubectl delete astradsvolumes --all -A
~% kubectl get astradsvolumes -A
```

## 2. 删除导出策略：

```
~% kubectl delete astradsexportpolicies --all -A
~% kubectl get astradsexportpolicies -A
```

### 删除Astra数据存储集群

删除集群时、只会删除Astra Data Store集群对象自定义资源(CR)以及集群范围的资源。



即使在删除集群后，操作符， nodeinfo Pod 和集群控制器（即 Kubernetes 范围的资源）仍会保持不变。

删除集群还会从节点卸载底层操作系统，从而停止 fireap 和 netwd 服务。

卸载程序大约需要一分钟才能完成。然后、开始删除Astra Data Store集群范围的资源。

## 1. 删除集群：

```
~% kubectl delete astradsclusters --all -A
~% kubectl get astradsclusters -A
```

### 删除许可证

1. 通过 SSH 连接到集群中的每个工作节点，并验证`fireap`或`netwd`未在工作节点中运行。
2. 删除Astra Data Store许可证：

```
~% kubectl delete astradslicenses --all -A
~% kubectl get astradslicenses -A
```

### 删除Astra Data Store安装

删除集群中的控制器，操作员，命名空间和支持 Pod。

## 1. 删除Astra Data Store安装对象：

```
~% kubectl delete astradsversion astradsversion -n astrads-system
~% kubectl get astradsversion -n astrads-system
```

## 2. 删除数据存储DemonSets和所有Astra Data Store控制器资源：

```
~% kubectl delete ds --all -n astrads-system
~% kubectl get ds -n astrads-system

~% kubectl delete deployments --all -n astrads-system
~% kubectl get deployments -n astrads-system
```

### 3. 删除其余项目和操作符 YAML 文件：

```
~% kubectl delete -f ./manifests/astradsoperator.yaml
~% kubectl get pods -n astrads-system
```

验证是否删除了 **astrads-system** 命名空间

确保以下命令不返回任何结果：

```
~% kubectl get ns | grep astrads-system
```

确保工作节点上未运行容器

验证 `fireap` 或 `netwd` 等容器是否未在工作节点上运行。在每个节点上运行以下命令。

```
ssh <mynode1>
# runc list
```

删除 **OpenShift** 容器平台资源

如果您在 Red Hat OpenShift 容器平台 (OCP) 上安装了 Astra 数据存储，则可以卸载 OCP 安全上下文约束 (SCC) 和绑定资源。

OpenShift 使用安全上下文约束（SCC）来控制 Pod 可以执行的操作。

完成标准卸载过程后，请完成以下步骤。

#### 1. 删除 SCC 资源：

```
oc delete -f ads_privileged_scc.yaml
```

#### 2. 删除 rolebindings 资源：

```
oc delete -f oc_role_bindings.yaml
```



忽略这些步骤中的 " 未找到资源错误 "。

## 手动删除示例

下面显示了执行手动卸载脚本的示例。

```
$ kubectl delete astradsvolumes --all -A
No resources found
$ kubectl delete astradsexportpolicies --all -A
No resources found
$ kubectl delete astradsclusters --all -A
astradscluster.astrads.netapp.io "astrads-sti-c6220-09-10-11-12" deleted

$ kubectl delete astradslicenses --all -A
astradslicense.astrads.netapp.io "e900000005" deleted

$ kubectl delete astradsdeployment astradsdeployment -n astrads-system
astradsdeployment.astrads.netapp.io "astradsdeployment" deleted

$ kubectl delete ds --all -n astrads-system
daemonset.apps "astrads-ds-astrads-sti-c6220-09-10-11-12" deleted
daemonset.apps "astrads-ds-nodeinfo-astradsdeployment" deleted
daemonset.apps "astrads-ds-support" deleted

$ kubectl delete deployments --all -n astrads-system
deployment.apps "astrads-cluster-controller" deleted
deployment.apps "astrads-deployment-support" deleted
deployment.apps "astrads-license-controller" deleted
deployment.apps "astrads-operator" deleted

$ kubectl delete -f /.../firetap/sds/manifests/netappsdsoperator.yaml
namespace "astrads-system" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsautosupports.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradscloudsnapshots.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsclusters.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsdeployments.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsexportpolicies.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsfaileddrives.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
```

```
"astradslicenses.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsnfsoptions.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsnodeinfoes.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsqospolicies.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsvolumefiles.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsvolumes.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsvolumesnapshots.astrads.netapp.io" deleted
role.rbac.authorization.k8s.io "astrads-leader-election-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradscloudsnapshot-
editor-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradscloudsnapshot-
viewer-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradscluster-editor-role"
deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradscluster-viewer-role"
deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradslicense-editor-role"
deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradslicense-viewer-role"
deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsvolume-editor-role"
deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsvolume-viewer-role"
deleted
clusterrole.rbac.authorization.k8s.io "astrads-autosupport-editor-role"
deleted
clusterrole.rbac.authorization.k8s.io "astrads-autosupport-viewer-role"
deleted
clusterrole.rbac.authorization.k8s.io "astrads-manager-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-metrics-reader" deleted
clusterrole.rbac.authorization.k8s.io "astrads-netappexportpolicy-editor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-netappexportpolicy-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-netappsdsdeployment-editor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-netappsdsdeployment-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-netappsdsnfsoption-editor-
role" deleted
```

```

clusterrole.rbac.authorization.k8s.io "astrads-netappsdsnfsoption-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-netappsdsnodeinfo-editor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-netappsdsnodeinfo-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-proxy-role" deleted
rolebinding.rbac.authorization.k8s.io "astrads-leader-election-
rolebinding" deleted
clusterrolebinding.rbac.authorization.k8s.io "astrads-manager-rolebinding"
deleted
clusterrolebinding.rbac.authorization.k8s.io "astrads-proxy-rolebinding"
deleted
configmap "astrads-autosupport-cm" deleted
configmap "astrads-firetap-cm" deleted
configmap "astrads-fluent-bit-cm" deleted
configmap "astrads-kevents-asup" deleted
configmap "astrads-metrics-cm" deleted
service "astrads-operator-metrics-service" deleted
Error from server (NotFound): error when deleting
"/.../export/firetap/sds/manifests/netappsdsoperator.yaml":
deployments.apps "astrads-operator" not found

$ kubectl get ns | grep astrads-system

[root@sti-rx2540-535c ~]# runc list
ID          PID          STATUS      BUNDLE      CREATED      OWNER

```

## 对Astra数据存储卸载过程进行故障排除

如果需要对卸载过程进行故障排除、请查看以下建议。

### Pod处于终止状态

Astra数据存储卸载过程有时可能会使发生原因 Pod在Kubernetes中保持终止状态。

如果发生此问题描述，请运行以下命令强制删除 astrads-system 命名空间中的所有 Pod：

```
kubectl delete pods --all -n astrads-system --force --grace-period 0
```

### 服务质量策略指向旧集群

如果您仅删除Astra数据存储集群并对其进行重新部署、则可能无法创建永久性卷声明(PVC)或卷、因为服务质量(QoS)策略指向旧集群且找不到。

1. 要避免这种情况、请在删除Astra Data Store集群后手动删除QoS策略：

```
kubectl delete AstraDSQosPolicy --all -A
```

## 2. 删除整个Astra Data Store部署(而不仅仅是集群):

```
uninstall.sh all
```

### 删除或卸载**Astra**数据存储后未删除密钥提供程序**CRS**

如果为要删除或卸载的Astra Data Store集群配置了外部密钥提供程序、则可能需要手动清理未删除的任何密钥提供程序CR。

## 示例 2. 详细信息

按照以下临时解决策 说明进行操作：

### 步骤

#### 1. 确认未删除密钥提供程序CRS：

```
kubectl get astradskeyprovider --selector  
astrads.netapp.io/cluster=astrads-cluster-example -n astrads-system
```

响应：

NAME	AGE
externalkeyprovider1	94s

#### 2. 删除密钥提供程序CRS：

##### a. 删除此查找器：

```
kubectl edit astradskeyprovider -n astrads-system
```

##### b. 删除下面突出显示的查找器行：

```
kubectl edit astradskeyprovider externalkeyprovider1 -n astrads-  
system
```



```

apiVersion: astrads.netapp.io/v1beta1
kind: AstraDSKeyProvider
metadata:
  creationTimestamp: "2022-05-24T16:38:27Z"
  finalizers:
  - astrads.netapp.io/astradskeyprovider-finalizer
  generation: 1
  labels:
    astrads.netapp.io/cluster: astrads-cluster-example
    astrads.netapp.io/rsid: "1"
  name: externalkeyprovider1
  namespace: astrads-system
  resourceVersion: "1134699"
  uid: a11111b2-31c0-4575-b7f3-97f9ab1a1bla
spec:
  cluster: astrads-cluster-example
  kmipServer:
    hostnames:
    - 10.xxx.xxx.xxx
    port: 5696
    secretRef: externalkeyprovider1
status:
  keyProviderUUID: a1b2cd34-4fc6-5bae-9184-2288c673181d
  kmipServerStatus:
    capabilities: '{ KMIP_library_version()=17367809,
KMIP_library_version_str()="KMIP
1.9.3a 8-Apr-2019", KMIP_library_version_tag()="KMIP part
of KMIP 1.9.3a 8-Apr-2019",
KMIP_library_is_eval()=false,
KMIP_library_fips_capable()=true(FIPS140),
KMIP_SSL_provider_build_version()=268444095,
KMIP_SSL_provider_version()=268444095,
KMIP_SSL_provider_version_str()="OpenSSL
1.0.2zb-fips 23 Sep 2021" }'
  keyServerUUID: 8422bdd0-74ad-579d-81bd-6d544ac4224a

```

c. 删除此查找器后、删除密钥提供程序CR:

```

kubectl delete astradskeyprovider <key-provider-cr-name> -n
astrads-system

```

## 无法从Astra控制中心Web UI卸载Astra数据存储

如果您从Astra控制中心Web UI启动了Astra数据存储卸载过程、则此过程有时可能会失败。

如果发生此问题描述、请执行以下步骤。

### 步骤

1. 登录到 ["NetApp 支持站点"](#) 并将Astra数据存储捆绑包(Astra\_Data\_Store\_2022.05.01.tar)下载到可以访问Astra数据存储所在的Kubernetes集群的计算机上。
2. 登录到已下载Astra数据存储捆绑包的计算机。
3. 提取捆绑包的内容：

```
tar -xvf <path to tar file>/Astra_Data_Store_2022.05.01.tar
```

4. 更改为存储卸载脚本的清单目录：

```
cd astrads/manifests/
```

5. 手动删除Astra数据存储：

```
./uninstall all
```

# 将Astra数据存储与VMware结合使用

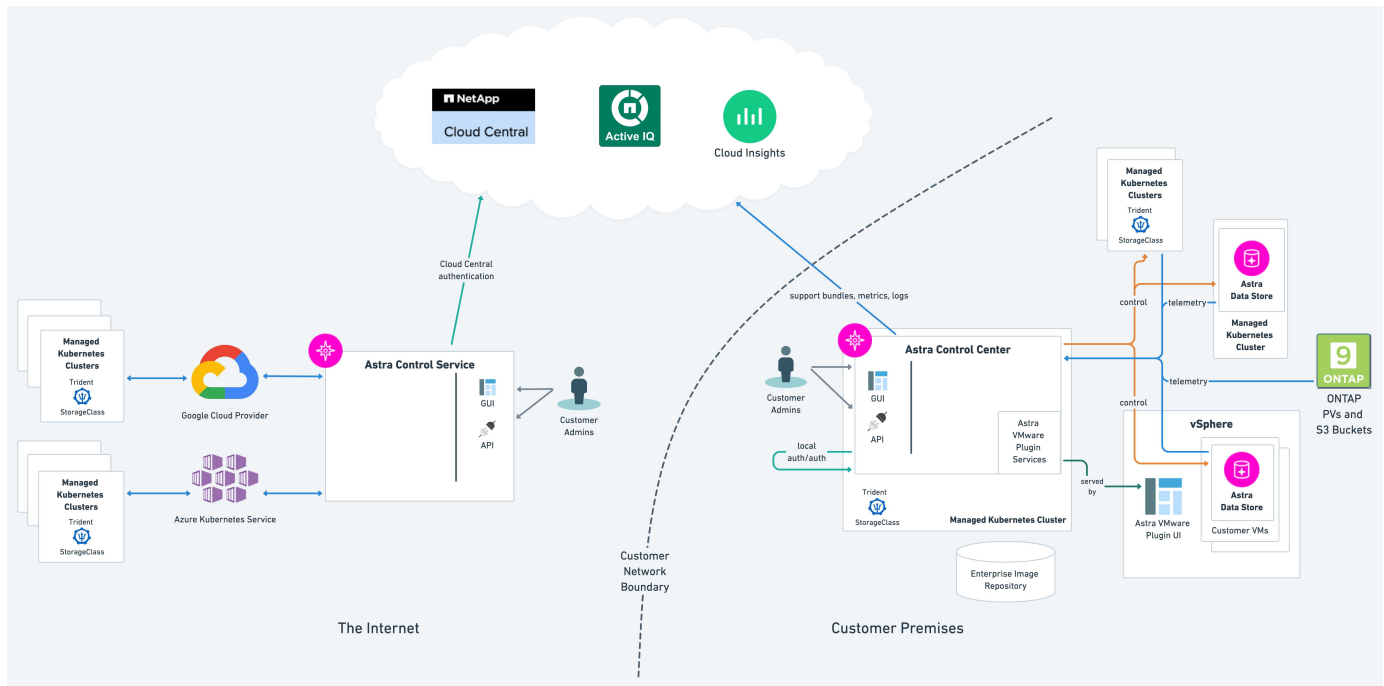
## 了解采用VMware的Astra Data Store

Astra数据存储既支持容器化工作负载、也支持虚拟化工作负载。通过与VVOL和基于存储策略的管理相集成、vSphere管理员可以应用存储服务质量。适用于VMware vSphere的NetApp Astra插件可提供熟悉的管理和监控体验、从而消除了繁琐的存储任务。

适用于VMware vSphere的Astra插件具有以下优势：

- VM粒度存储配置、具有完整的VVOL和VASA集成功能
- 基于存储策略的管理集成
- 适用于vSphere原生 管理的vCenter插件

下图显示了采用VMware的Astra系列生态系统。



有关详细信息 ...

- ["Astra 控制中心文档"](#)
- ["Astra 系列简介"](#)

## 采用VMware的Astra Data Store要求

确认您的环境符合常规要求后 ["Astra 数据存储要求"](#)下、您应验证您的环境是否也满足VMware组件的最低要求、例如VMware VASA Provider和适用于VMware vSphere的NetApp Astra插件。

## VMware vSphere要求

Astra数据存储使用VMware VASA Provider作为API接口与存储进行通信。确保您的环境满足基本要求 "[VASA Provider要求](#)" 以及以下附加要求：

- VMware vSphere 7.0 (支持更新1到更新3)
- 一个用于传入流量的未分配IP地址



- 适用于VMware vSphere的NetApp Astra插件不支持vSphere 7.0 Update 3c；请改用vSphere 7.0 Update 3D。
- 适用于VMware vSphere的NetApp Astra插件不支持链接模式vCenter服务器。

## 适用于VMware vSphere的NetApp Astra插件要求

适用于VMware vSphere的NetApp Astra插件具有以下要求：

- 在Kubernetes集群上运行的Astra Control Center实例
- 在Kubernetes集群上运行的许可的Astra Data Store实例

支持的 **Web** 浏览器

适用于VMware vSphere的NetApp Astra插件支持最新版本的以下Web浏览器：

- Mozilla Firefox
- Microsoft Edge (基于铬)
- Google Chrome

有关详细信息 ...

- ["Astra 控制中心文档"](#)
- ["Astra 系列简介"](#)
- ["Astra 数据存储要求"](#)

## 使用VMware设置Astra数据存储

您可以将Astra数据存储设置为存储后端、并使用适用于VMware vSphere的NetApp Astra插件对其进行管理。

在VMware中设置Astra数据存储涉及以下任务：

- [\[Activate VMware vSphere services using Astra Control Center\]](#)。
- [\[Add a vCenter using Astra Control Center\]](#)。
- [\[Create a custom SCC \(if using OpenShift\)\]](#)
- [\[Use an existing storage backend in the Astra Plugin for VMware vSphere\]](#)。
- [\[Create a datastore using the Astra Plugin for VMware vSphere\]](#)。

- [\[Generate VM storage policies\]](#)。

在使用VMware设置Astra数据存储之前、您需要确保满足以下要求：

- Astra控制中心是 "已安装" 并进行设置。



对于Astra Data Store Early Access Program (EAP)版本、如果要使用Astra Control Center管理Astra Data Store并启用VMware工作流、请仅在`pcloud`命名空间上部署Astra Control Center。

- Astra Data Store已获得许可并已部署。请参见 ["安装 Astra 数据存储"](#)。
- 用于部署Astra控制中心和Astra数据存储的Kubernetes集群必须已由Astra控制中心进行管理。
- 在添加vCenter之前、您已上传Astra控制中心和VASA提供程序包。 ["请参见管理软件包"](#)。

## 使用Astra控制中心激活VMware vSphere服务

通过激活Astra控制中心上的vSphere服务、开始在VMware中设置Astra数据存储。



默认情况下、Astra控制中心中的VMware vSphere服务未启用。

1. 登录到 Astra 控制中心。
2. 从左侧导航栏中选择\*集群\*。

横幅将显示一条消息、指出VMware vSphere服务尚未启用。

3. 选择\*启用VMware vSphere服务\*。

此操作可能需要一段时间。启用服务后、\*添加vCenter\*按钮将启用。

## 使用Astra控制中心添加vCenter

添加第一个vCenter、用于注册适用于VMware vSphere的Astra插件。

要将vCenter添加到Astra控制中心、您必须具有管理权限。



在将插件注册到VMware vSphere后、适用于VMware vSphere的Astra插件图标将显示在VMware快捷方式页面中。有时、在注册适用于VMware vSphere的Astra插件后、此插件不会立即显示。在这种情况下、请等待几秒钟、然后刷新浏览器。

1. 登录到 Astra 控制中心。
2. 从左侧导航栏中选择\*集群\*。
3. 选择\*添加vCenter\*。
4. 输入vCenter Server详细信息、vCenter端口以及管理用户名和密码、以便将其提供给Astra控制中心。



这样可以在VMware vSphere客户端中为此vCenter部署Astra插件。

## 5. 选择 \* 添加 \*。

vCenter将显示在集群页面上、受管vCenter的总数将在Astra控制信息板上进行更新。此操作还会启动适用于VMware vSphere的Astra插件部署。

### 验证是否添加了vCenter

新添加的vCenter将显示在"Clusters"页面和"Dashboard"上。



vCenter和Kubernetes集群都会显示在Astra控制中心信息板上。

1. 访问Astra控制中心。
2. 从左侧导航栏中选择\*集群\*。
3. 验证新管理的vCenter是否显示在"Clusters"页面上。
4. 从左侧导航栏中选择 \* 信息板 \*。
5. 在Astra控制中心信息板中、记下新的受管vCenter集群作为\*受管\*计数的一部分。



受管集群计数包括vCenter和Kubernetes集群。

6. 要查看集群详细信息、请单击\*受管\*计数。

此时将显示集群页面。

## 创建自定义SCC (如果使用OpenShift)

如果您使用的是OpenShift、则可以选择分配安全上下文约束(SCC)、以控制Pod可以执行的操作并控制Pod可以访问的内容。

默认情况下，任何容器的执行都将获得受限的 SCC，并且仅获得该 SCC 定义的功能。受限SCC不提供VASA Provider Pod所需的权限。使用此操作步骤 为VASA Provider部署所使用的服务帐户提供所需的更高权限(在示例中列出)。

为Astra Data Store "NTV-system"命名空间的各种默认服务帐户分配一个自定义SCC、该命名空间由特权和节点-导出程序的SCC混合而组成。

只有在 Red Hat OpenShift 容器平台（ OCP ）上部署时，才需要执行以下步骤。

1. 创建名为`vp-backend\_privileged\_scc.yaml`的自定义SCC：

```
kubectl create -f vp_backend_privileged_scc.yaml
```

示例：VP\_backend\_privileged\_SCC.YAML

```
allowHostDirVolumePlugin: true
allowHostIPC: false
```

```

allowHostNetwork: true
allowHostPID: false
allowHostPorts: true
allowPrivilegeEscalation: true
allowPrivilegedContainer: true
allowedCapabilities:
  - '*'
allowedUnsafeSysctls:
  - '*'
apiVersion: security.openshift.io/v1
defaultAddCapabilities: null
fsGroup:
  type: RunAsAny
groups: []
kind: SecurityContextConstraints
metadata:
  name: vpbackend-privileged
priority: null
readOnlyRootFilesystem: false
requiredDropCapabilities: null
runAsUser:
  type: RunAsAny
seLinuxContext:
  type: RunAsAny
seccompProfiles:
  - '*'
supplementalGroups:
  type: RunAsAny
users:
  - system:serviceaccount:ntv-system:default
  - system:serviceaccount:ntv-system:ntv-auth-svc
  - system:serviceaccount:ntv-system:ntv-autosupport
  - system:serviceaccount:ntv-system:ntv-compliance-svc
  - system:serviceaccount:ntv-system:ntv-datastore-svc
  - system:serviceaccount:ntv-system:ntv-metallb-controller
  - system:serviceaccount:ntv-system:ntv-metallb-speaker
  - system:serviceaccount:ntv-system:ntv-mongodb
  - system:serviceaccount:ntv-system:ntv-nfs-svc
  - system:serviceaccount:ntv-system:ntv-rabbitmq-svc
  - system:serviceaccount:ntv-system:ntv-storage-svc
  - system:serviceaccount:ntv-system:ntv-vault
  - system:serviceaccount:ntv-system:ntv-vault-admin
  - system:serviceaccount:ntv-system:ntv-vault-agent-injector
  - system:serviceaccount:ntv-system:ntv-vault-controller
  - system:serviceaccount:ntv-system:ntv-vault-initializer
  - system:serviceaccount:ntv-system:ntv-vcenter-svc

```

```

- system:serviceaccount:ntv-system:ntv-vm-management-svc
- system:serviceaccount:ntv-system:ntv-watcher-svc
- system:serviceaccount:ntv-system:ntv-vault-sa-vault-tls
- system:serviceaccount:ntv-system:ntv-gateway-svc
- system:serviceaccount:ntv-system:ntv-jobmanager-svc
- system:serviceaccount:ntv-system:ntv-vasa-svc
volumes:
- '*'

```

## 2. 使用 `oc get scc` 命令显示新添加的 SCC：

```
oc get scc vpbakend-privileged
```

响应：

NAME	PRIV	CAPS	SELINUX	RUNASUSER	FSGROUP	SUPGROUP
PRIORITY	READONLYROOTFS	VOLUMES				
vpbakend-privileged	true	["*"]	RunAsAny	RunAsAny	RunAsAny	RunAsAny
<no value>	false	["*"]				

## 在适用于VMware vSphere的Astra插件中使用现有存储后端

使用Astra控制中心UI添加vCenter后、使用适用于VMware vSphere的Astra插件添加Astra数据存储存储后端。

此过程将完成以下操作：

- 将现有存储后端添加到选定vCenter。
- 向选定vCenter注册VASA Provider。VASA提供程序可在VMware和Astra Data Store之间提供通信。
- 将VASA Provider自签名证书添加到存储后端。



有时、添加的存储后端可能需要几分钟时间才能显示在存储后端向导中。



Astra数据存储不应与多个vCenter共享。

### 步骤

1. 访问适用于VMware vSphere的NetApp Astra插件。
2. 从左侧导航栏中选择\*适用于VMware vSphere的Astra插件\*、或者从快捷方式页面中选择\*适用于VMware vSphere的Astra插件\*图标。
3. 从适用于VMware vSphere的Astra插件概述页面中、选择\*使用现有存储后端\*。或者、从左侧导航栏中选择\*存储后端\*>\*添加\*、然后选择\*使用现有存储后端\*。
4. 选择现有的Astra数据存储作为存储后端、然后选择\*下一步\*。



5. 在VASA Provider页面上、输入VASA Provider名称、IP地址(如果使用负载均衡器)、用户名和密码。



对于用户名、可以使用字母数字字符和下划线。请勿输入任何特殊字符。用户名的第一个字母必须以字母字符开头。

6. 指示是否要部署负载均衡器并输入IP地址、此地址将用于访问VASA提供程序。此IP必须是一个与节点IP不同的可路由可用IP。启用负载均衡器后、Metalb将部署在Astra Data Store Kubernetes集群中、并配置为分配可用IP。



如果要使用Google Anthos集群进行部署、请选择不部署负载均衡器、因为Anthos已经将metalb作为负载均衡器运行。在VASA Provider CR (v1beta1\_vasaprovider.yaml)中、metalb Deploy标志应设置为false。

如果选择不部署负载均衡器、则假定已部署并配置负载均衡器、以便为类型为\*负载均衡器\*的Kubernetes服务分配IP。



此时、尚未部署VASA Provider。

7. 选择 \* 下一步 \*。
8. 在证书页面上、查看自签名证书的证书信息。
9. 选择 \* 下一步 \*。
10. 查看摘要信息。
11. 选择 \* 添加 \*。

此操作将部署VASA Provider。

## 在适用于VMware vSphere的Astra插件中验证存储后端

注册Astra Data Store存储后端后、它将显示在适用于VMware vSphere的Astra插件存储后端列表中。

您可以确定存储后端状态和VASA Provider状态。您还可以查看每个存储后端的已用容量。

选择存储后端后、您还可以查看已用容量和可用容量、数据缩减比率以及内部网络管理IP地址。

### 步骤

1. 在适用于VMware vSphere的NetApp Astra插件中、从左侧导航栏中选择\*存储后端\*。
2. 选择Astra Data Store存储后端以查看摘要选项卡。
3. 查看VASA提供程序的已用和可用容量、数据精简率和状态。
4. 选择其他选项卡可查看有关VM、数据存储库、主机和存储节点的信息。

## 使用适用于VMware vSphere的Astra插件创建数据存储库

添加存储后端并注册适用于VMware vSphere的Astra插件后、您可以在VMware中创建数据存储库。

您可以将数据存储库添加到数据中心、计算或主机集群。



您不能使用同一存储后端在同一数据中心下创建多个数据存储库。

您可以使用NFS协议添加VVol数据存储库类型。

#### 步骤

1. 访问适用于VMware vSphere的Astra插件。
2. 从插件菜单中、选择\*创建数据存储库\*。
3. 输入新的数据存储库名称、类型(VVol)和协议(NFS)。
4. 选择 \* 下一步 \*。
5. 从存储页面中、选择您刚刚创建的Astra Data Store存储后端。



您不能使用具有现有数据存储库的存储后端。

6. 选择 \* 下一步 \*。
7. 在摘要页面中、查看相关信息。
8. 选择 \* 创建 \*。



如果遇到与扫描失败或常规系统错误相关的错误、["在vCenter上重新扫描/同步存储提供程序"](#)然后尝试重新创建数据存储库。

## 生成VM存储策略

创建数据存储库后、在创建VM之前、您应使用REST API UI中的`/virtual/apl/v1/vCenters/VM-storage-policies`生成预先设计的VM存储策略。

#### 步骤

1. 要访问REST API UI页面、请转到`https://<ads\_gateway\_ip>:8443`。
2. 转至API `POST /virtual/apl/auth/login`并提供用户名、密码和vCenter主机名。

响应：

```
{
  "vmware-api-session-id": "212f4d6447b05586ab1509a76c6e7da56d29cc5b",
  "vcenter-guid": "8e475060-b3c8-4267-bf0f-9d472d592d39"
}
```

3. 转至API `GET /virtual/apl/auth/validate-session`并完成以下步骤：
  - a. 使用上面生成的`vmware-api-session-id`和`vCenter-guid`作为标头。
  - b. 选择\*立即试用\*。

响应：(身份验证截断如下)：

```
authorization: eyJhbGciOiJSUzI1NiIsInR...9h15DYYvClT3oA  connection:
keep-alive  content-type: application/json  date: Wed, 18 May 2022
13:31:18 GMT  server: nginx  transfer-encoding: chunked
```

4. 转至API 虚拟化/apI/v1/vcenters /vm-storage-policies、并将在上一响应中生成的承载令牌添加为"authorization"。

此时将显示"200"响应、并生成三个VM存储策略。

5. 在vCenter存储策略页面上验证新的虚拟机存储策略(名为铜牌、银牌和金牌)。
6. 请继续创建VM。

## 下一步行动

接下来、您可能需要执行以下任务：

- 创建VM。
- 挂载数据存储库。请参见 ["挂载数据存储库"](#)。

## 有关详细信息 ...

- ["Astra 控制中心文档"](#)
- ["Astra 系列简介"](#)

## 监控VMware安装的组件

您可以使用适用于VMware vSphere的NetApp Astra插件监控Astra Data Store安装的组件。您可以监控系统的运行状况、包括存储后端、VASA Provider、VM和VVOL。您还可以查看容量和vCenter信息。

### 使用适用于VMware vSphere的Astra插件信息板监控系统的运行状况

使用VMware环境管理Astra数据存储涉及监控存储后端和VASA提供程序的整体运行状况。

使用适用于VMware vSphere的NetApp Astra插件信息板、您可以查看以下信息：

- 此vCenter中所有存储后端的已用物理容量和可用容量。您可以将鼠标悬停在信息上并查看更多详细信息。
- 运行状况良好且运行状况不正常的存储后端和VASA提供程序
- 排名前10位的VM的延迟、IOPS、吞吐量和容量利用率。

在信息板中、您可以执行多项附加任务：

- 监控容量
- 使用现有存储后端。请参见 ["设置存储后端"](#)。
- 访问产品文档

## 查看信息板的步骤

1. 访问适用于VMware vSphere的Astra插件。
2. 在概述页面中、查看以下部分：
  - a. \*存储后端\*部分：您可以单击存储后端和VASA提供程序的状态以查看其状态的详细信息。您也可以单击以查看所有存储后端。
  - b. \*存储后端容量\*部分：查看选定vCenter中存储后端的总物理已用容量和可用容量。要更改vCenter Server、请单击右上角的vCenter Server选项。
  - c. \*虚拟机\*部分：查看容量利用率排名前10位的虚拟机。



您可以改为单击表标题来显示所需内容、例如、延迟较高的前10个VM。

## 在其他视图中监控Astra数据存储的步骤

1. 访问以下视图以监控Astra数据存储组件：
  - \*虚拟机\*选项卡：列出由Astra数据存储管理的所有虚拟机、与信息板相比、信息板仅列出排名前10位的虚拟机。
  - \*存储\*深入分析：显示与存储系统关联的主机、虚拟机和数据存储库。
  - \*虚拟机存储\*视图：提供VASA提供程序创建的VVOL的详细信息。

## 查看存储后端阈值设置

存储后端容量阈值设置用于控制何时在存储后端的所有数据存储库上显示警报通知。

使用适用于VMware vSphere的Astra插件部署或添加存储后端时、系统会设置以下默认阈值：

- 90%全满会生成红色警报
- 80%全满会生成黄色警报

您可以在VMware中查看系统生成警报的级别。



对于Astra数据存储早期访问计划、如果在多个数据中心中使用同一个存储容器、则可能会看到数据存储库的警报不正确。

## 步骤

1. 访问适用于VMware vSphere的NetApp Astra插件。
2. 从左侧导航栏中、选择\*设置\*。
3. 查看设置阈值。

## 有关详细信息 ...

- ["Astra 控制中心文档"](#)
- ["Astra 系列简介"](#)

# 管理VMware安装的Astra Data Store组件

您可以在vSphere环境中以及从Astra控制中心管理以下Astra数据存储组件：

- [\[Work with managed vCenters\]](#)
- [\[Manage VMs from vSphere\]](#)
- [\[Manage the storage backend\]](#)
- [\[Manage datastores\]](#)

## 使用受管vCenter

您可以通过以下方式使用受管vCenter：

- [\[View vCenter details in Astra Control Center\]](#)
- [\[View vCenter details in Astra Control Center\]](#)
- [\[Unmanage a vCenter in Astra Control Center\]](#)

在**Astra**控制中心中查看**vCenter**详细信息

您可以确定与集群关联的所有vCenter。

步骤

1. 从Astra控制中心左侧导航栏中、选择\*集群\*。
2. 查看vCenter列表。
3. 选择 \* 保存 \*。

在**Astra**控制中心中查看**vCenter**详细信息

您可能希望查看系统和集群的运行状况。您可以通过查看Astra控制中心信息板来确定管理的集群数量。

步骤

1. 从Astra控制中心左侧导航栏中、选择\*集群\*。
2. 选择vCenter。
3. 查看相关信息。

在**Astra**控制中心取消管理**vCenter**

如果您不再希望Astra控制中心管理vCenter、则可以取消对其进行管理。此操作将从Astra控制中心删除和取消注册vCenter。



必须先从适用于VMware vSphere的Astra插件中删除与此vCenter关联的集群、存储后端和VM。

步骤

1. 从Astra控制中心左侧导航栏中、选择\*集群\*。
2. 从集群页面中、选择vCenter。



或者、选择多个vCenter并选择\*取消全部管理\*。

3. 右键单击\*操作\*菜单并选择\*取消管理\*。
4. 在Unmanage vCenter页面中、键入"unmanage"。
5. 选择\*是、取消管理vCenter\*。

## 从vSphere管理VM

您可以使用原生 vSphere操作管理与Astra数据存储关联的VM。

- "删除虚拟机"
- "重命名VM"
- "调整VM大小"



在此版本中、一次只能调整一个VM磁盘的大小。尝试调整多个磁盘大小将失败。

- "打开或关闭VM"
- "暂停虚拟机"
- "重置虚拟机"

使用原生 vCenter操作的Astra数据存储可使用以下快照工作流：

- "创建Astra数据存储的快照"
- "还原快照"
- "删除快照"



Snapshot操作有时可能会失败、并显示VVol运行时错误。如果发生这种情况、请重试此操作。

## 管理存储后端

您可以删除存储后端。删除存储后端不会将其销毁、也不会删除Astra Data Store产品本身；它只是从VMware取消注册VASA Provider并取消该vCenter的存储后端链接。



如果VASA Provider已启用并部署在vCenter外部、则只能删除Astra数据存储。如果在配置数据存储库过程中使用存储后端、您将无法删除存储后端。

如果Astra数据存储未链接到多个vCenter、则删除该vCenter后、VASA Provider将被取消注册并卸载。

### 步骤

1. 访问适用于VMware vSphere的Astra插件。
2. 从左侧导航栏中、选择\*存储后端\*。
3. 在存储后端页面上、单击存储后端操作菜单并选择\*删除\*。
4. 输入VASA Provider用户名和密码。

5. 选择 \* 删除 \*。

## 管理数据存储库

您可以使用原生 vCenter操作来管理VM、并使用Astra插件扩展来管理数据存储库、从而在vSphere环境中管理Astra数据存储库：

- ["创建数据存储库"](#)
- [\[Mount a datastore\]](#)
- [\[Delete a datastore\]](#)

### 挂载数据存储库

使用适用于VMware vSphere的Astra插件、您可以将数据存储库挂载到一个或多个其他主机上。

#### 步骤

1. 从vCenter中的数据中心清单中选择用于Astra数据存储的数据存储库。
2. 右键单击数据存储库、然后选择\*适用于VMware vSphere的Astra插件\*>\*挂载数据存储库\*。
3. 从在主机上挂载数据存储库页面中、选择要挂载数据存储库的主机。



如果要在所有主机上挂载数据存储库、请选中\*在所有主机上挂载\*。

4. 选择\*挂载\*。

启动此操作后、您可以在vSphere Client的近期任务面板中按照进度进行操作。



如果遇到与扫描失败或常规系统错误相关的错误、["在vCenter上重新扫描/同步存储提供程序"](#) 然后尝试重新创建数据存储库。

### 删除数据存储库

使用适用于VMware vSphere的Astra插件、您可以删除数据存储库。



要删除此数据存储库、必须先删除此数据存储库上的所有VM。

#### 步骤

1. 从vCenter中的数据中心清单中选择数据存储库。
2. 右键单击数据存储库、然后选择\* Astra插件\*>\*删除数据存储库\*。
3. 在删除数据存储库页面中、确认此信息或采取其他建议的操作、以便可以删除此数据存储库。
4. 选择\*删除\*。

## 有关详细信息 ...

- ["Astra 控制中心文档"](#)
- ["Astra 系列简介"](#)

# 从VMware集成环境中卸载Astra Data Store

您可以从vSphere环境中卸载Astra数据存储及其相关组件。

请参见 ["这些说明"](#) 卸载Astra数据存储时。

有关详细信息 ...

- ["Astra 控制中心文档"](#)
- ["Astra 系列简介"](#)



# 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](https://kb.netapp.com/Advice_and_Troubleshooting/Cloud_Services/Astra)

## 获取帮助

NetApp 以多种方式为 Astra 数据存储提供支持。全天候提供免费自助服务支持选项，例如知识库（KB）文章和可宽延的渠道。



您可以获得Astra Data Store的社区技术支持。使用创建案例 "[NetApp 支持站点（NSS）](#)" 不适用于此版本。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。

### 自助服务支持选项

这些选项全天候免费提供：

- "[知识库（需要登录）](#)"

搜索与Astra数据存储相关的文章、常见问题解答或中断修复信息。

- 文档。

这是您当前正在查看的文档站点。

- "[netapp](#)" 容器 " [可宽延通道](#)"

转到 " 容器 " 渠道，与同行和专家建立联系。

- 反馈电子邮件

发送电子邮件至 [astra.feedback@netapp.com](mailto:astra.feedback@netapp.com) ，告知我们您的想法，想法或顾虑。

### 了解更多信息

- "[如何将文件上传到 NetApp（需要登录）](#)"
- "[NetApp 知识库文章](#)"

## 自动支持监控

AutoSupport 监控 Astra 数据存储系统运行时间和信息，并向 NetApp 支持部门发送消息。根据您的配置，可以监控以下系统组件：

- 控制面板

- 存储

默认情况下，期间会启用 AutoSupport "[Astra Data Store 集群安装](#)" 或在将 AutoSupport 自定义资源（CR）应用于集群后。启用后，AutoSupport（ASUP）捆绑包将自动上传到 "[NetApp 支持站点（NSS）](#)" 或可供手动下载。

#### 选项

- [\[AutoSupport triggers and scenarios\]](#)
- [\[Configure custom control plane AutoSupport collection\]](#)
- [\[Configure custom storage AutoSupport collection\]](#)
- [\[List ASUPs in the system\]](#)
- [\[Download an ASUP Bundle\]](#)
- [\[Upload a core file\]](#)

## AutoSupport 触发器和场景

AutoSupport 捆绑包可通过以下方式触发：

- \* 定期 \*：ASUP 捆绑包按 CR 中定义的间隔创建。
- \* 用户触发 \*：您可以手动创建自己的 ASUP 以查看日志。
- \* 核心转储 \*：如果节点上存在核心转储，则会生成 ASUP，并将核心发送给 NetApp 进行进一步调查。
- \* 基于 CallHome 事件 \*：从操作系统的特定 CallHome 事件生成 ASUP。
- \* 基于 Kubernetes 事件 \*：从控制平面中的特定 Kubernetes 事件生成 ASUP。

这些触发情形会生成以下 AutoSupport 类型之一：

- 控制平面**CRS**：一组AutoSupport 数据存储控制平面日志和。
- \* 存储 AutoSupport \*：存储报告和性能数据的集合。
- \* 核心转储 AutoSupport \*：系统核心转储的集合。

## 配置自定义控制平面 AutoSupport 收集

您可以创建一个自定义 AutoSupport 收集配置，用于报告控制平面事件。默认情况下，大多数安装都已在期间启用定期事件报告 "[Astra Data Store 集群安装](#)"。此操作步骤介绍如何配置 AutoSupport CR，以便根据您选择的参数进行报告：

#### 步骤

1. 自定义以下命令以创建控制面板集合 CR：

```
kubectl astasds asup collect -c controlplane --namespace=astrads-system
```

- a. 定义自定义参数：

- `<myASUPname>`：要生成的 AutoSupport CR 的名称。

- `-e <事件名称>`：触发收集的事件名称。事件名称应在 `component-yaml` 中预定义（该名称已挂载到支持控制器）。

#### 示例

```
kubectl astrasds asup collect -c controlplane custom-asup-name -e debug --namespace=astrads-system
```

#### a. 根据需要为系统添加其他参数：

- `-cluster`：在多集群环境中需要此标志。
- `-localCollection`：启用本地收集。默认值为 `false`。
- `-forceUpload`：启用强制上传。默认值为 `false`。
- `--retry`：启用重试。默认值为 `false`。

## 配置自定义 **Storage AutoSupport** 收集

您可以创建自定义 AutoSupport 收集配置，以报告存储组件事件。默认情况下，大多数安装都已在期间启用定期事件报告 "[Astra Data Store 集群安装](#)"。此操作步骤介绍如何配置 AutoSupport CR，以便根据您选择的参数进行报告：

#### 步骤

#### 1. 自定义以下命令以创建存储收集 CR：

```
kubectl astrasds asup collect -c storage --namespace=astrads-system
```

#### a. 定义自定义参数：

- `<myASUPname>`：要生成的 AutoSupport CR 的名称。
- `-e <事件名称>`：触发收集的事件名称。事件名称应在 `component-yaml` 中预定义（该名称已挂载到支持控制器）。

#### 性能事件示例：

```
kubectl astrads asup collect -c storage -e performance example-perf-storage-asup
```

- `-t <ISO_format> -d <hours>`：在指定的持续时间内为所有节点收集存储 ASUP。请使用标准 ISO 日期时间格式（`-t`），持续时间（`D`）以小时为单位。例如：

```
kubectl astrads asup collect -c storage -t 2021-01-01T15:00:00Z -d 24
```

- `-nodes <节点名称>`：为指定节点收集存储 ASUP。例如：

```
kubectl astrads asup collect -c storage --nodes example1
```

- `-nodes nodename1, nodeame2, nodeame3`：为指定节点收集存储 ASUP：

```
kubectl astrads asup collect -c storage --nodes  
example1,example2,example3
```

a. 根据需要为系统添加其他参数：

- `-cluster`：在多集群环境中需要此标志。
- `-localCollection`：启用本地收集。默认值为 `false`。
- `-forceUpload`：启用强制上传。默认值为 `false`。
- `--retry`：启用重试。默认值为 `false`。

## 列出系统中的 ASUP

使用以下命令按名称列出系统中的 ASUP：

```
kubectl astrads asup list --namespace=astrads-system
```

响应示例：

NAMESPACE	NAME	SEQUENCE	NUMBER	EVENT
SIZE	STATE	LOCAL	COLLECTION	
astrads-system	storage-callhome.reboot.unknown-...	1		
callhome.reboot.unknown	0	uploaded	astrads-ds-support-tdl2h:	
astrads-system	storage-callhome.reboot.unknown-...	2		
callhome.reboot.unknown	0	uploaded	astrads-ds-support-xx6n8:	
astrads-system	storage-callhome.reboot.unknown-...	3		
callhome.reboot.unknown	0	uploaded	astrads-ds-support-qghnx:	

## 下载 ASUP 捆绑包

您可以使用此命令下载本地收集的 ASUP 捆绑包。使用 `-o <位置>` 指定当前工作目录以外的位置：

```
./kubectl-astrasds asup download <ASUP_bundle_name> -o <location>
```

## 上传核心文件

如果服务崩溃，则会创建 AutoSupport（ASUP）消息以及崩溃时包含相关内存内容的文件（称为核心文件）。Astra Data Store 会自动将 ASUP 消息上传到 NetApp 支持部门，但您需要手动上传核心文件，以便它

与ASUP消息相关联。

## 步骤

1. 使用以下 `kubectl` 命令查看 ASUP 消息：

```
kubectl astrads asup list --namespace=astrads-system
```

您应看到类似于以下内容的输出：

NAMESPACE	NAME	SEQUENCE NUMBER	EVENT
SIZE	STATE	LOCAL COLLECTION	
astrads-system	storage-coredump-2021...	1	coredump
197848373	compressed	astrads-ds-support-sxxn7:/var/...	

2. 使用以下 `kubectl` 命令从 ASUP 消息中下载核心文件。使用 `-o`` 选项为下载的文件指定目标目录。

```
kubectl astrads asup download storage-coredump-20211216t140851311961680  
-o <absolute_path_to_destination_directory>
```



在极少数情况下，您可能无法下载核心文件，因为其他核心文件已取代它。发生这种情况时，命令将返回错误 `cannot stat : no such file or directory`。如果您看到此错误，可以 ["获取帮助"](#)。

3. 打开 Web 浏览器并浏览到 ["NetApp 身份验证文件上传工具"](#)，如果您尚未登录，请输入您的 NetApp 支持凭据。
4. 选中 `* 我没有案例编号 *` 复选框。
5. 在 `* 最近的区域 *` 菜单中，选择最接近您的区域。
6. 选择 `* 上传 *` 按钮。
7. 浏览并选择先前下载的核心文件。

此时将开始上传。上传完成后，将显示一条成功消息。

## 了解更多信息

- ["如何将文件上传到 NetApp（需要登录）"](#)

# 早期版本的Astra数据存储

可提供先前版本的文档。

- ["Astra Data Store 21.12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

<http://www.netapp.com/us/legal/copyright.aspx>

## 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。

<http://www.netapp.com/us/legal/netapptmlist.aspx>

## 专利

有关 NetApp 拥有的专利的最新列表，请访问：

<https://www.netapp.com/us/media/patents-page.pdf>

## 隐私政策

<https://www.netapp.com/us/legal/privacypolicy/index.aspx>

## 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。

["有关Astra Data Store "EAP"版本的通知"](#)

## 版权信息

版权所有©2022 NetApp、Inc.。保留所有权利。Printed in the U.S.版权所涵盖的本文档的任何部分不得以任何形式或任何手段复制、包括影印、录制、磁带或存储在电子检索系统中—未经版权所有者事先书面许可。

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

本软件由NetApp按"原样"提供、不含任何明示或默示担保、包括但不限于适销性和特定用途适用性的默示担保、特此声明不承担任何任何责任。IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

RESTRICTED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.277-7103 (October 1988) and FAR 52-227-19 (June 1987).

## 商标信息

NetApp、NetApp标识和中列出的标记 <http://www.netapp.com/TM> 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。