使用 **Astra** 数据存储

Astra Data Store

NetApp
June 15, 2022

目录

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
AstraDSExportPolicy
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                                      ADSCLUSTER
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:
      --as string                      Username to impersonate for the
operation
      --as-group stringArray           Group to impersonate for the
operation, this flag can be
                                         repeated to specify multiple
groups.
      --cache-dir string               Default HTTP cache directory
                                         (default "/u/arda/.kube/http-
cache")
      --certificate-authority string   Path to a cert file for the
certificate authority
      --client-certificate string      Path to a client certificate file
for TLS
      --client-key string              Path to a client key file for TLS
      --cluster string                 The name of the kubeconfig
cluster to use
      --context string                 The name of the kubeconfig
context to use
  -h, --help                           help for astrads
        --insecure-skip-tls-verify       If true, the server's
certificate will not be checked
                                         for validity. This will make
your HTTPS connections insecure
        --kubeconfig string              Path to the kubeconfig file to
use for CLI requests.
    -n, --namespace string               If present, the namespace scope
for this CLI request
        --request-timeout 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.
                                         (default "0")
    -s, --server string                  The address and port of the
Kubernetes API server
        --token string                   Bearer token for authentication
to the API server
        --user 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.
                                   This should be a positive integer
      -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
```

```
                                    uploading because it was created with
the 'local' option or if
                                    automatic uploads of AutoSupports is
disabled
                                    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
           --nodes string           Specify nodes to collect for storage
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
   ```

   响应：

   ```
   adsDataNetworks:
       -addresses:  dataIP1,dataIP2,dataIP3,dataIP4,NEW_DATA_IP
   ```

   a. 保存 CR 。

   b. 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
AGE     VERSION   LABELS
sti-astramaster-050.lab.org   Ready    control-plane,master   3h39m
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=
sti-rx2540-556a.lab.org       Ready    worker                 3h38m
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
sti-rx2540-556b.lab.org       Ready    worker                 3h38m
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
sti-rx2540-534d.lab.org       Ready    worker                 3h38m
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,kuber
netes.io/arch=amd64,kubernetes.io/hostname=sti-rx2540-
557a.lab.org,kubernetes.io/os=linux,node-role.kubernetes.io/worker=true
sti-rx2540-557b.lab.org       Ready    worker                 3h38m
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
```

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`nhcitj1525`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   v1.20.0
 nhcitjj1526       Ready    <none>                 3d18h   v1.20.0
 nhcitjj1527       Ready    <none>                 3d18h   v1.20.0
 nhcitjj1528       Ready    <none>                 3d18h   v1.20.0
 scs000039783-1    Ready    control-plane,master   3d18h   v1.20.0
```

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 drives adddrive show-available --cluster=CLUSTER_NAME
```

响应：

```
Node: node1.name
Add drive maximum size: 100.0 GiB
Add drive minimum size: 100.0 GiB
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
Add drive maximum size: 66.7 GiB
Add drive minimum size: 100.0 GiB
No suitable drives to add exist.

Node: node3.name
Add drive maximum size: 100.0 GiB
Add drive minimum size: 100.0 GiB
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
Add drive maximum size: 66.7 GiB
Add drive minimum size: 100.0 GiB
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 drives 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 drives 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 adscl -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
6000c290   sti-rx2540-300b.lab.netapp.com   ard-6e4b4af    ReadyToReplace
13m
```

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

有关详细信息 ...

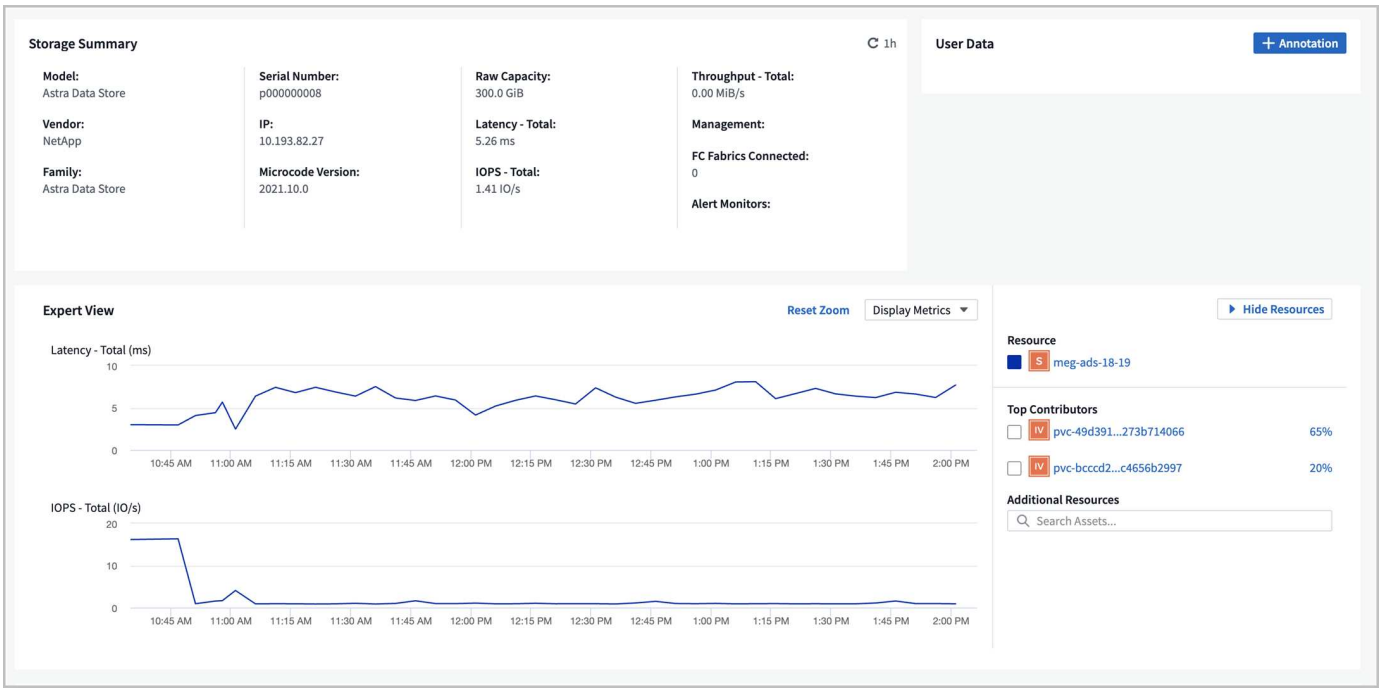
- "使用kubectl命令管理Astra Data Store资源"
- "将节点添加到Astra控制中心的存储后端集群"

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：

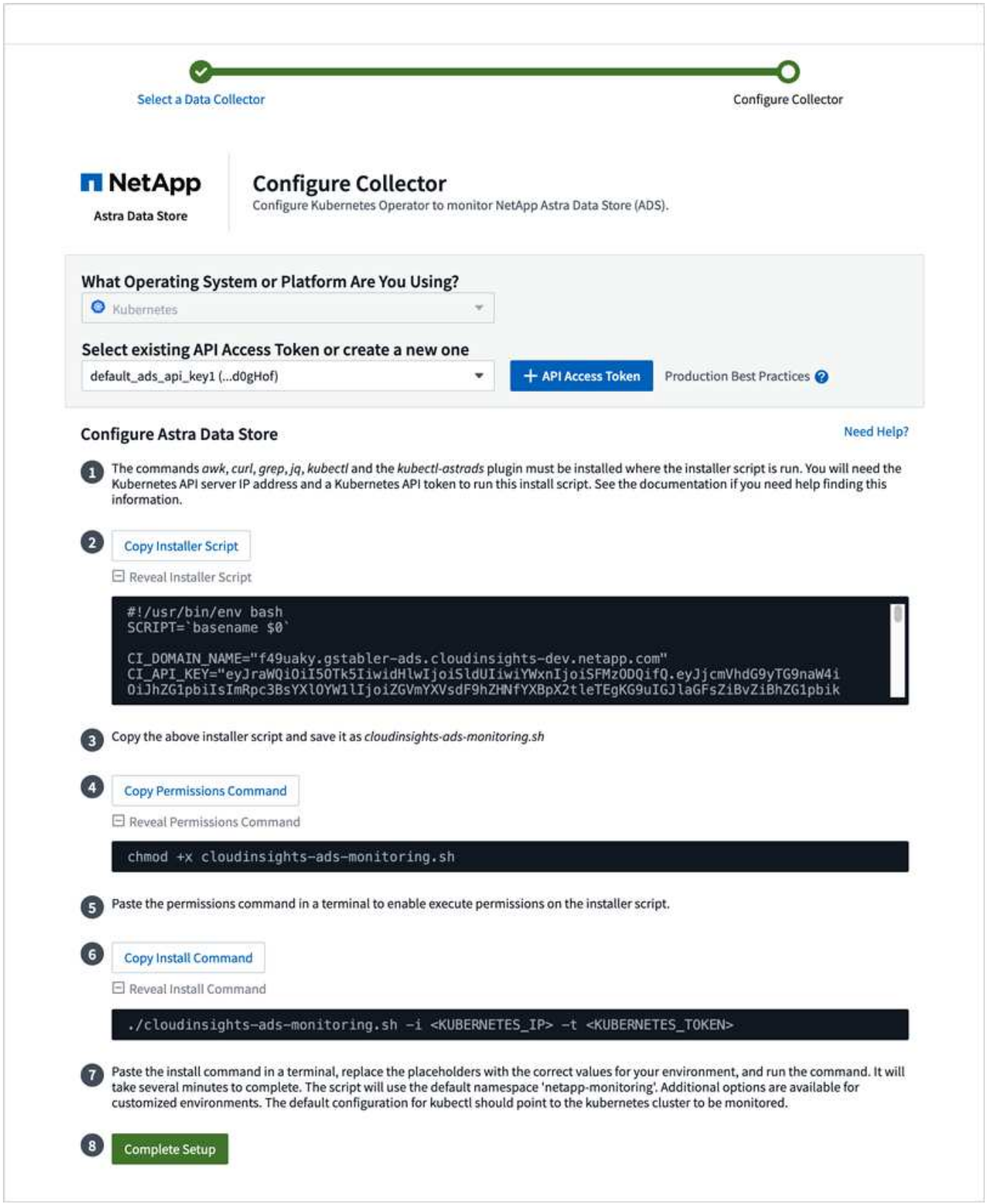
```
USAGE: cloudinsights-ads-monitoring.sh [OPTIONS]
Configure monitoring of Astra Data Store by Cloud Insights.
OPTIONS:
  -h                      Display this help message.
  -d ci_domain_name       Cloud Insights tenant domain name.
  -i kubernetes_ip        Kubernetes API server IP address.
  -k 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。



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: bzl9ngz.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`telecraf`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 thoughput 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 thoughput 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         ADSCLUSTER                  VALID   PRODUCT
EVALUATION   ENDDATE     VALIDATED
p100000006   astrads-example-cluster     true    Astra Data Store    true
2023-01-23   2022-04-04T14:38:54Z
```

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 `kubectl`扩展来升级Astra Data Store。

使用kubectl升级Astra数据存储

您可以使用Astra Data Store `kubectl`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.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.pub -signature
Astra_Data_Store_2022.05.sig 2022.12.01_ads.tar
```

3. 创建目录：

```
mkdir Astra_Data_Store_2022.05
cd Astra_Data_Store_2022.05
```

4. 提取映像：

```
tar -vxzf <path to tar file>/Astra_Data_Store_2022.05.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kubectl-astrad二进制文件从用于提取映像的目录复制到安装了Kubernetes 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：

```
kubectl-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：

```
kubectl-astrads upgrade ads-version --repository-url REPOSITORY_URL
--ads-version-yaml ./astrads/manifests/astradsversion.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~netappsdsoperator.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
Deleting 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-astradsvolumefile-editor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsvolumefile-viewer-
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 **1.** 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-b7f3-97f9abla1bla
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
```