



# **Astra** データストアを使用

## Astra Data Store

NetApp  
June 02, 2022

# 目次

Astra データストアを使用 .....	1
kubectlコマンドを使用してAstraデータストアのリソースを管理 .....	1
テストアプリケーションを展開します .....	7
Astraデータストアクラスターを管理 .....	11
Astraデータストアを監視 .....	21
Secure Astraデータストア .....	36
Astraデータストアライセンスを更新 .....	41
Astraデータストアをアップグレード .....	42
Astraデータストアを自動化されたスクリプトでアンインストールします .....	44

# Astra データストアを使用

## kubectlコマンドを使用してAstraデータストアのリソースを管理

kubectlコマンドを使用し、Kubernetes API拡張機能を使用して、Astraデータストアのリソースを管理できます。

サンプルアプリの展開方法については、を参照してください ["テストアプリケーションを展開します"](#)。

クラスタのメンテナンスについては、を参照してください ["クラスタを管理"](#)。

必要なもの

- にインストールしたAstra Data Store kubectlプラグイン ["Astra データストアをインストール"](#)

### Astraデータストア用のKubernetesカスタムAPIリソースを列挙

Kubernetes内でkubectlコマンドを使用して、Astraデータストアクラスタとやり取りし、状態を確認することができます。

「api-resources」コマンドにリストされた各項目は、Astraデータストアがクラスタの管理に内部的に使用するKubernetesカスタムリソース定義（CRD）を表します。

このリストは、後で示すように、各Astra Data Storeオブジェクトの簡単な名前を取得して入力を減らすのに特に役立ちます。

1. Astraデータストア用のKubernetesカスタムAPIリソースのリストを表示します。

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

対応：

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. Kubernetesクラスタ内の現在のAstraデータストアオブジェクトをすべて取得するには、「`kubect1 get ads-a`」 コマンドを使用します。

```
kubect1 get ads -A
```

対応：

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

- 短縮名の 1 つを使用して、クラスタ内のボリュームの現在の状態を表示します。

```
kubectl get adsvo -A
```

対応：

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				

## kubectl 拡張子の help オプションを使用します

'kubectl astras コマンド'には'h' スイッチが含まれており、'便利な使用法とフラグ・ドキュメント'が提供されています

- Astra Data Store kubectl拡張機能のすべてのコマンドのヘルプを表示します。

```
kubectl astrads -h
```

対応：

```
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 for a single server request. Non-zero values should contain a corresponding time unit (e.g. 1s, 2m, 3h).
<code>-s, --server</code> string	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. コマンドの詳細については 'astrads [command]--help' を使用してください

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

対応:

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  
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
                                Specify nodes to collect for storage
                                --nodes string
component. (default "all")      Specify nodes to collect for storage
    -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")

```

## テストアプリケーションを展開します

Astra データストアで利用できるテストアプリケーションを導入する手順は次のとおりです。

この例では、Helm リポジトリを使用して Bitnami から MongoDB チャートを導入します。

必要なもの

- Astra データストア クラスターの導入と構成
- Trident のインストールが完了しました

手順

1. Bitnami から Helm repo を追加します。

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

2. MongoDB の導入

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

### 3. MongoDB ポッドのステータスを確認します。

```
~% kubectl get pods -n ns-mongodb
NAME                                READY   STATUS    RESTARTS   AGE
mongodb-9846ff8b7-rfr4r            1/1     Running   0           67s
```

### 4. MongoDB で使用した Persistent Volume Claim (PVC ; 永続ボリューム要求)を確認します。

```
~% 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. kubectl コマンド 'get astradsvolume' を使用して 'ボリュームを一覧表示します

```
~% 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. kubectl コマンド 'describe astadsvolume' を使用して 'ボリュームを説明します

```
~% 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

```

## Astraデータストアクラスタを管理

クラスタを管理するには、Astraデータストアでkubectlコマンドを使用します。

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

必要なもの

- kubectl および kubectl-astras プラグインがインストールされたシステム。を参照してください ["Astra データストアをインストール"](#)。

## ノードを追加します

追加するノードは Kubernetes クラスタに含まれ、クラスタ内の他のノードと同様の設定である必要があります。



Astra Control Centerを使用してノードを追加するには、を参照してください ["ストレージバックエンドクラスタにノードを追加します"](#)。

### 手順

1. 新しいノードのdataIPがまだAstra Data StoreクラスタCRに含まれていない場合は、次の手順を実行します。
  - a. クラスタCRを編集し、「adsDataNetworks\*Addresses \*」フィールドにデータIPを追加します。大文字の情報を環境に適した値に置き換えます。

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

対応：

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

- a. CR を保存します。
- b. アストラデータストアクラスタにノードを追加します。大文字の情報を環境に適した値に置き換えます。

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

2. それ以外の場合はノードを追加します。大文字の情報を環境に適した値に置き換えます。

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

3. ノードが追加されたことを確認します。

```
kubectl astrads nodes list
```

## ノードを削除します

クラスタ内のノードを削除するには、kubectlコマンドをAstraデータストアとともに使用します。

### 手順

1. すべてのノードを一覧表示します。

```
kubectl astrads nodes list
```

対応：

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

2. ノードを削除対象としてマークします。大文字の情報を環境に適した値に置き換えます。

```
kubectl astrads nodes remove NODE_NAME
```

対応：

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

ノードを削除対象としてマークするとノードのステータスはアクティブから現在の状態に変わります

3. 削除したノードの現在のステータスを確認します

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

対応：

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. ノードからAstraデータストアをアンインストールします。大文字の情報を環境に適した値に置き換えます。

```
kubectl astrads nodes uninstall NODE_NAME
```

5. ノードがクラスタから削除されたことを確認します。

```
kubectl astrads nodes list
```

ノードがAstraデータストアから削除される。



## ノードをメンテナンスモードにします

ホストのメンテナンスやパッケージのアップグレードが必要な場合は、ノードをメンテナンスモードにする必要があります。



ノードがAstraデータストアクラスタにすでに含まれている必要があります。

ノードが保守モードのときは、クラスタにノードを追加できません。この例では、ノード「nhcitj1525」をメンテナンスモードにします。

### 手順

1. ノードの詳細を表示します。

```
kubectl get nodes
```

対応：

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. ノードがまだメンテナンスモードになっていないことを確認します。

```
kubectl astrads maintenance list
```

応答（メンテナンスモードのノードがありません）：

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

3. メンテナンスモードを有効にします。大文字の情報を環境に適した値に置き換えます。

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

例：

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

対応：

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

#### 4. ノードを一覧表示します。

```
kubectl astrads nodes list
```

対応：

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

#### 5. メンテナンスモードのステータスを確認します。

```
kubectl astrads maintenance list
```

対応：

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

保守モードの場合は 'false' として起動し 'true' に変更します「保守状態」が「準備期間の保守」から「ReadyforMaintenance」に変更されます。

#### 6. ノードのメンテナンスが完了したら、メンテナンスモードを無効にします。

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

#### 7. ノードが保守モードでなくなったことを確認します。

```
kubectl astrads maintenance list
```

## ノードにドライブを追加

Astraデータストアでkubectlコマンドを使用して、Astraデータストアクラスタ内のノードに物理ドライブまたは仮想ドライブを追加。

必要なもの

- 次の条件を満たす1つ以上のドライブ：
  - ノードに設置済み（物理ドライブ）またはノードVMに追加済み（仮想ドライブ）
  - ドライブにパーティションがありません
  - ドライブがクラスタで使用されていません
  - クラスタ内のドライブの物理容量がライセンスで許可されている物理容量を超えない（たとえば、CPUコアあたり2TBのストレージをライセンスで付与すると、10ノードのクラスタの最大物理ドライブ容量は20TBになります）
  - ドライブには、ノード内の他のアクティブドライブのサイズ以上が含まれています



Astraデータストアのノードあたりのドライブ数は16本以下17台目のドライブを追加しようとすると、ドライブ追加要求は拒否されます。

手順

1. クラスタについて説明します。

```
kubectl astrads clusters list
```

対応：

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

2. クラスタ名をメモします。
3. クラスタ内のすべてのノードに追加できるドライブを表示します。cluster\_nameをクラスタの名前に置き換えます。

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

対応：

```

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. 次のいずれかを実行します。

- 使用可能なすべてのドライブの名前が同じ場合は、それらのドライブをそれぞれのノードに同時に追加できます。大文字の情報を環境に適した値に置き換えます。

```

kubect1 astrads drives adddrive create --cluster=CLUSTER_NAME --name
REQUEST_NAME --drivesbyname all=DRIVE_NAME

```

- ドライブの名前が異なる場合は、各ノードにドライブを1つずつ追加できます（追加する必要があるドライブごとにこの手順を繰り返す必要があります）。大文字の情報を環境に適した値に置き換えます。

```

kubect1 astrads drives adddrive create --cluster=CLUSTER_NAME --name
REQUEST_NAME --drivesbyname NODE_NAME=DRIVE_NAME

```

Astraデータストアがドライブの追加要求を作成し、要求の結果を含むメッセージが表示される。

## ドライブを交換します

クラスタ内のドライブで障害が発生した場合は、データの整合性を確保するために、できるだけ早くドライブを交換する必要があります。ドライブで障害が発生した場合は、クラスタのCRノードステータス、クラスタの健全性状態、および指標エンドポイントにある、障害が発生したドライブに関する情報を確認できます。障害が発生したドライブの情報を表示するには、次のコマンドを使用します。

**nodeStatus.driveStatuses** で障害が発生したドライブを示すクラスタの例

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

対応：

```
...
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
  ...
```

障害が発生したドライブCRは、障害が発生したドライブのUUIDに対応する名前でクラスタ内に自動的に作成されます。

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

対応：

```
...
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
```

対応：

NAME	NODE	CLUSTER	STATE
AGE			
6000c290	sti-rx2540-300b.lab.netapp.com	ard-6e4b4af	ReadyToReplace
13m			

手順

1. 交換可能なドライブを'kubectl strads faileddrive show -replacions'コマンドで一覧表示しますこのコマンドは'交換の制限に適合するドライブをフィルタリングします（クラスタでは未使用'マウントなし'パーティションなし'または障害が発生したドライブ以上）

可能な交換用ドライブをフィルタリングせずにすべてのドライブを一覧表示するには '--all' を 'show-replacements' コマンドに追加します

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

対応：

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

2. パスしたシリアル番号でドライブを交換するには 'replace' コマンドを使用しますコマンドは置換を完了するか '--wait' 時間が経過すると失敗します

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



kubectl の astrads faileddrive replace' が不適切なシリアル番号を使用して実行された場合、次のようなエラーが表示されます

```
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. ドライブ交換を再実行するには '前のコマンドで --force' を使用します

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

を参照してください。

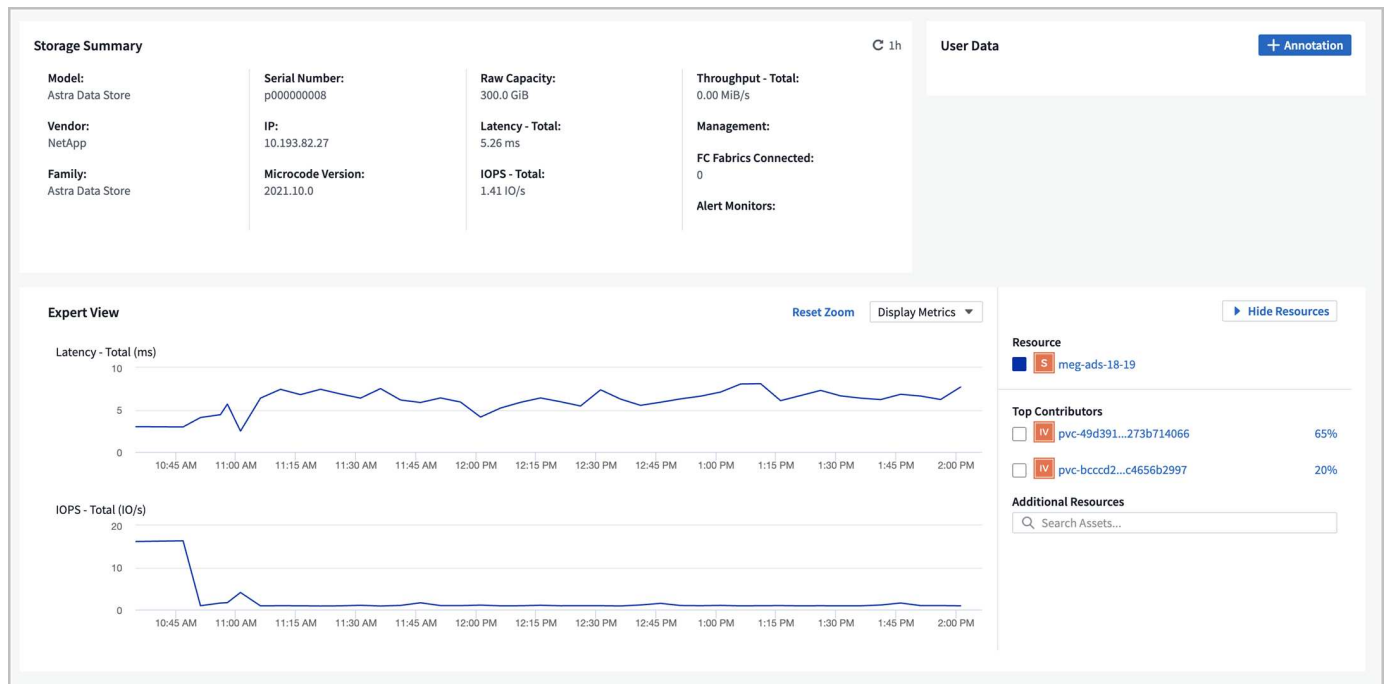
- ["kubectlコマンドを使用してAstraデータストアのリソースを管理"](#)

## Astraデータストアを監視

### Cloud Insights で指標を監視

Cloud Insights を使用して、Astraデータストアの指標を監視できます。

Cloud Insights に表示されるAstraデータストアの指標の例を次に示します。



を使用して、Astraデータストアで生成された指標のリストを表示することもできます [\[Open Metrics API help\]](#)。

次のタスクを実行できます。

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

## Cloud Insights 接続の前提条件となる作業を完了する

Cloud Insights を使用して Astra データストアに接続する前に、次の作業を完了する必要があります。

- ["Astra Data Store Monitoring Operator をインストールします"](#) これは、Astra Data Storeのインストール手順の一部です。
- ["kubecti-astras バイナリをインストールします"](#) これは、Astra Data Storeのインストール手順の一部です。
- ["Cloud Insights アカウントを作成します"](#)。
- 'awk'、'curl'、'grep' および 'jq' の各コマンドが使用可能であることを確認します

次の情報を収集します。

- \* Cloud Insights API アクセストークン \*。各カテゴリに対する読み取り / 書き込み権限：Acquisition Unit、Data Collection、Data Ingestion、Log Ingestionこれは、読み取り / 書き込み処理、Acquisition Unit のセットアップ、およびデータの取り込みプロセスのセットアップに使用されます。



- \* Kubernetes API サーバの IP アドレスとポート \*。これは、Astra データストア クラスターを監視するために使用されます。
- \* Kubernetes API トークン \*。これは Kubernetes API を呼び出すために使用されます。
- \* 永続ボリューム構成 \*。永続ボリュームのプロビジョニング方法に関する情報。

## Acquisition Unit のストレージ

Acquisition Unit には、インストールファイル、設定データ、およびログを格納するための永続ボリュームが 3 つ必要です。Monitoring Operator は、デフォルトのストレージクラスを使用して、永続的ボリューム要求を作成します。インストーラ・スクリプトの実行時に '-s' オプションを使用して '別のストレージ・クラス名を指定できます

Kubernetes クラスターにストレージプロビジョニングツール（NetApp Trident など）がない場合は、インストーラ スクリプトの実行時に '-r' オプションを使用してローカルファイルシステムパスを提供できます。'-r' オプションが設定されている場合 'インストーラ・スクリプトは '指定されたディレクトリ内に 3 つの永続ボリュームを作成しますこのディレクトリには、150GB 以上の空きスペースが必要です。

インストールスクリプトをダウンロードして実行します

Cloud Insights では、モニタリングオペレータによるアストラデータストアのモニタリングを有効にするための Bash スクリプトを提供しています。インストールスクリプトは、Astra Data Store コレクタと Fluent ビットエージェントを備えた Acquisition Unit をインストールします。

Cloud Insights テナントのドメイン名と選択した Cloud Insights API アクセストークンは、ダウンロード時にインストーラ スクリプトに組み込まれます。

その後、次のように指標が送信されます。

- Cloud Insights Acquisition Unit から Cloud Insights データレイクに指標が送信されます。
- FLUENT ビットは、ログ取り込みサービスにログを送信します。

インストーラ スクリプトのヘルプを表示します

インストーラ スクリプトの完全なヘルプテキストを次に示します。

インストーラ スクリプトのヘルプテキストを表示します。

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

対応：

```

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.

```

インストールスクリプトを実行します

1. Cloud Insights アカウントがない場合は作成します。
2. Cloud Insights にログインします。
3. Cloud Insights メニューから、**Admin>\*Data Collector\*** をクリックします。
4. 「\* + Data Collector \*」 をクリックして、新しいコレクタを追加します。



5. 「\* アストラデータストア \*」 タイルをクリックします。
6. 正しい Cloud Insights API アクセストークンを選択するか、新しいトークンを作成します。
7. 指示に従って、インストーラスクリプトをダウンロードし、権限を更新し、スクリプトを実行します。

このスクリプトには、Cloud Insights テナントの URL と選択した Cloud Insights API アクセストークンが含まれています。



Kubernetes

default\_ads\_api\_key1 (...d0gHof) ▼

Production Best Practices ?

[Need Help?](#)

- ## 2 Copy Installer Script

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

CI_DOMAIN_NAME="f49uaky.gstabler-ads.cloudinsights-dev.netapp.com"
CI_API_KEY="eyJyJWQwOiIiOTk5IiwidHlwIjoiaSdUIiwiaWxnIjoisiFMzODQifQ.eyJjcVhdG9yTG9naW4iOiJhZGipbikImRmcc3B5YXl0YW1lIjoizGVmYXVsdF9hZHNFYXBxMjtleTEgK69uIGJlaGF5ZiBvZiBhZGipbik
```

- #### 4 Copy Permissions Command

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

- 6 Copy Install Command

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

- ## 8 Complete Setup

インストールスクリプトが完了すると、「データソース」リストに「Astra Data Store」コネクタが表示されます。



エラーが原因でスクリプトが終了した場合は、エラーが解決してから再度実行できます。デフォルトの設定を使用しない環境では、Monitoring Operator 名前空間や Kubernetes API サーバポートなどの追加のパラメータがサポートされます。使い方とヘルプテキストを表示するには、cloudinsights-ads-monitoring.sh -h オプションを使用します。

設定が正常に完了すると、次のような出力が生成されます。

```
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!
```

エージェント **CR** の例

以下に、インストーラスクリプトの実行後の「Monitoring - NetApp」エージェントの CR の例を示します。

```

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

```

### Cloud Insights 接続を編集します

Kubernetes API トークンまたは Cloud Insights API アクセストークンはあとから編集できます。

- Kubernetes API トークンを更新する場合は、Cloud Insights UI から Astra データストアコレクタを編集する必要があります。
- テレメトリとログに使用される Cloud Insights API アクセストークンを更新する場合は、kubectl コマンドを使用して Monitoring Operator CR を編集する必要があります。

#### Kubernetes API トークンを更新します

1. Cloud Insights にログインします。
2. **[Admin>]** > **[\* Data Collectors]** を選択して、**[Data Collectors]** ページにアクセスします。
3. Astra データストアクラスターのエントリを探します。
4. ページの右側にあるメニューをクリックし、「\* 編集 \*」を選択します。
5. Kubernetes API トークンフィールドを新しい値で更新します。
6. **[コレクタの保存 \*]** を選択します

#### Cloud Insights API アクセストークンを更新します

1. Cloud Insights にログインします。
2. **[Admin>\*API Access\*]** を選択し、**[\*+API アクセストークン \*]** をクリックして、新しい Cloud Insights API アクセストークンを作成します。
3. エージェント CR を編集します。

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

4. 「output-sink」セクションを探し、「ci」という名前のエントリを見つけます。
5. ラベル「api-key」の場合は、現在の値を新しい Cloud Insights API アクセストークンに置き換えます。

セクションは次のようになります。

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

6. エディタウィンドウを保存して終了します。

モニタリングオペレータは、新しいCloud Insights APIアクセストークンを使用するようにFluentビットを更新します。

#### Cloud Insights から切断します

Cloud Insights から切断するには、最初に Cloud Insights UI から Astra データストアコレクタを削除する必要があります。これが完了したら、モニタリングオペレータからAcquisition Unit、Telegraf（設定されている場合）、およびFluentビットの設定を削除できます。

## Astra Data Store コレクタを削除

1. Cloud Insights にログインします。
2. [Admin>] > [\* Data Collectors] を選択して、[Data Collectors] ページにアクセスします。
3. Astra データストアクラスターのエントリを探します。
4. 画面の右側のメニューを選択し、「\* 削除 \*」を選択します。
5. 確認ページで \* Delete \* をクリックします。

## Acquisition Unit、Telegraf（設定されている場合）、およびFluentビットを取り外します

1. エージェント CR を編集します。

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

2. 「au」セクションを探し、「IsEnabled」を「false」に設定します
3. 「FLUENT ビット」セクションを探し、「ADS テール CI」という名前のプラグインを削除します。プラグインがない場合は、「FLUENT - BIT」セクションを削除できます。
4. Telegrafが設定されている場合は、「テレグラム」セクションを探し、「ADSオープンメトリック」という名前のプラグインを削除します。プラグインがない場合は、「テレグラム」セクションを削除できます。
5. 「output-sink」セクションを探し、「ci」という名前のシンクを取り外します。
6. エディタウィンドウを保存して終了します。

モニタリングオペレータがTelegraf（設定されている場合）およびFluentビット設定を更新し、Acquisition Unitポッドを削除します。

7. ストレージプロビジョニング担当者ではなく Acquisition Unit PVS にローカルディレクトリを使用した場合は、PVS を削除します。

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

次に、Acquisition Unit を実行していたノードの実際のディレクトリを削除します。

8. Acquisition Unit ポッドが削除されたら、Cloud Insights から Acquisition Unit を削除できます。
  - a. Cloud Insights メニューで、**Admin>\*Data Collector\*** を選択します。
  - b. [\* Acquisition Units \*（Acquisition Unit \*）] タブをクリックします。
  - c. Acquisition Unit ポッドの横にあるメニューをクリックします。
  - d. 「\* 削除」を選択します。

モニタリングオペレータは、Telegraf（設定されている場合）およびFluentビットの設定を更新し、Acquisition Unitを削除します。

## Open Metrics API のヘルプを参照してください

Astraデータストアから指標を収集するために使用できるAPIのリストを次に示します。

- 「help」行は指標を表します。
- 「type」行は、メトリックがゲージかカウンタかを示します。

```
# 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
```

## Prometheus と Grafana で指標を監視します

PrometheusとGrafanaを使用して、Astraデータストアの指標を監視できます。PrometheusでAstra Data Store Kubernetesクラスター指標エンドポイントから指標を収集するように設定でき、Grafanaを使用して指標データを表示できます。

### 必要なもの

- PrometheusパッケージとGrafanaパッケージをAstra Data Storeクラスター、またはAstra Data Storeクラスターと通信可能な別のクラスターでダウンロードしてインストールしておく必要があります。公式ドキュメントの指示に従って、各ツールをインストールします。

- ["Prometheus をインストールする"](#)
- ["Grafana をインストールします"](#)
- PrometheusとGrafanaがAstra Data Store Kubernetesクラスタと通信する必要があります。PrometheusとGrafanaがAstra Data Storeクラスタにインストールされていない場合は、Astra Data Storeクラスタで実行されている指標サービスと通信できることを確認する必要があります。

## Prometheus を設定する

Astraデータストアは、KubernetesクラスタのTCPポート9341で指標サービスを公開します。このサービスから指標を収集するには、Prometheus を設定する必要があります。

### 手順

1. Prometheus インストール用の「prometheus.yml」構成ファイルを編集します。
2. Astra Data Storeサービス名とそのポートを指すサービスターゲットを追加します。例：

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

3. Prometheus サービスを開始します。

## Grafana を設定します

Prometheus で収集された指標を表示するように Grafana を設定できます。

### 手順

1. Grafana インストールの「datasources.yaml」設定ファイルを編集します。
2. Prometheus をデータソースとして追加します。例：

```
apiVersion: 1

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

3. Grafana サービスを開始します。
4. Grafana のマニュアルに記載されている手順に従ってください ["始めましょう"](#)。

## Grafana ダッシュボードテンプレートをインポートします

Astra Data Storeをインストールするためにダウンロードしたバンドルファイルには、Grafana内からインポートできるGrafanaダッシュボードテンプレートファイルが含まれています。このダッシュボードテンプレートを使用すると、Astraデータストアから取得できる指標の種類とその表示方法を確認できます。

### 手順

1. Astra Data Storeの.tar.gzバンドルを開きます。
2. 'マニフェスト'ディレクトリを開きます
3. 「`graafana_cluster.json`」ファイルと「`graafana_volume.json`」ファイルを展開します。
4. Grafana Web UI の使用、["でのダッシュボードテンプレートファイルを Grafana にインポートします"](#)。

## イベントログを設定して監視する

Event Management System（EMS；イベント管理システム）ログを監視するには、次の高度なタスクを実行します。

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

## Astra Data Storeクラスタカスタムリソース（CR）での監視の設定

Astra Data StoreクラスタCRでモニタリングオプションが設定されていない場合は、「astras」拡張機能を使用して設定できます。

### 入力するコマンド

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

ここで、

- インストールされているエージェントのネームスペース：Monitoring Agent のネームスペースを入力します。この名前は、Monitoring Operator のデフォルトの名前である NetApp CR になります。
- -r は 'Fluent イメージまたは Telegraf イメージが置かれている Docker レジストリをセットアップするためのオプションですデフォルトでは 'パスは `docker.repo.eng.netapp.com/global/astra` ' に設定されていますこのパスは変更できます

## Cloud Insights をセットアップする

ログを表示するには Cloud Insights の設定は任意ですが、Cloud Insights を使用してデータを表示すると便利です。を参照してください ["NetApp Cloud Insights のセットアップ方法"](#) Astraデータストアで使用。

## イベントログを Elastic にストリーミングする

EMS イベントやその他のポッドログを Elastic などのサードパーティのエンドポイントにストリーミングする

には、「astrads」拡張機能を使用します。

入力するコマンド

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



Elastic ホスト名は IP アドレスでもかまいません。

## Secure Astra データストア

### セキュリティ証明書を管理する

Astra データストアでは、クラスタのソフトウェアコンポーネント間でMTLS (Mutual Transport Layer Security) 暗号化を使用しています。各Astra Data Storeクラスタには、自己署名ルートCA証明書 (「astrs-cert-root」) と中間CA証明書 (「astrs-cert-<cluster\_name>」) があります。この証明書はAstra Data Store オペレータによって管理され、有効期限の7日前にオペレータが証明書を自動的に更新します。証明書を手動で取り消すこともできます。

#### 証明書を取り消します

Astra データストアコントローラ、ノード、またはCA証明書が侵害された場合、MTLSシークレットを削除することでCA証明書を取り消すことができます。これを行うと、Astra Data Storeオペレータは自動的に新しい証明書を発行します。Astra データストア証明書はいつでも取り消すことができます。



CA証明書を取り消すと、そのCAによって署名された証明書がすべて取り消されます。

#### 手順

1. Astra Data Storeクラスタのコントローラノードにログインします。
2. システム上の既存の証明書の一覧を表示します。例：

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

次のような出力が表示されます。

```

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

```

- 出力には、取り消す必要がある証明書の名前が表示されます。
- 'kubectl'ユーティリティを使用して'証明書を取消しますこれは'certificate\_name'を証明書の名前に置き換えます例：

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

既存の証明書が失効し、代わりに新しい証明書が自動的に生成されます。

## 外部キーを管理します

1 つ以上の外部キー管理サーバを使用して、暗号化されたデータにアクセスする際にクラスタで使用するキーを安全に保管できます。外部キー管理サーバはストレージ環境に配置されたサードパーティのシステムで、Key Management Interoperability Protocol (KMIP) を使用してノードにキーを提供します。



Astraデータストアでは、Astraデータストアクラスタを作成すると、デフォルトで内部キープロバイダを使用して保存データの暗号化 (sear) が有効になります。

キーの管理には、次のカスタムリソース定義 (CRD) が含まれます。

- \* AstraDSKeyProvider\*：外部KMIPサーバを設定します。このサーバはサーバのクラスタの場合があります。
- \* AstraDSSEARKeyRotate\*：キープロバイダから新しいキー暗号化キーを取得し、Astraデータストアに提供します。

外部キー管理に関連して次のタスクを実行できます。

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

- [\[Rotate keys for security\]](#)

## 外部キー管理をセットアップする

Astra Data Storeで外部キー管理を設定するには'kubectl astrs'コマンドを使用します

クラスタまたはKMIPサーバにSSL証明書が必要です。これにより、OpenSSLなどを使用した外部キーの設定などが可能になります。

### 手順

1. キープロバイダクライアントの証明書を準備します。クライアント証明書、クライアント秘密鍵、および信頼CAバンドルが含まれます。



クラスタまたはKMIPサーバで、OpenSSLなどを使用した外部キーの設定を可能にするSSL証明書を準備します。

2. Astraデータストアクラスタのいずれかのノードにログインします。
3. 次のkubectl拡張コマンドを入力して、Astraデータストアクラスタのキープロバイダを設定します。

```
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>
```

次の例では、ADSクラスタ「astrs-cluster-f23d158」に対して「hashicorp」という名前の外部キープロバイダを設定します。

```
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. Astra Data StoreクラスタをAstraDSCluster CR経由で、外部キーマネージャを使用するように設定します。ヘルプを表示します。

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

対応：



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  
...
```

次のコマンドは'Astra Data Store'クラスターを'AstraDSKeyProvider hashicorp'をsearのキー管理ツールとして使用するよう設定します。また、キーのローテーション時間も使用されます。この時間のデフォルト値は90日(2160時間)です。

```
kubect1-astrads clusters sears -d 500h hashicorp  
--ads-cluster-name=astrads-cluster-f23d158  
--ads-cluster-namespace=astrads-system
```

ソフトウェアの保存データの暗号化ステータスを確認します

保存データのソフトウェア暗号化の設定を確認できます。

ステップ

1. AstraDSCluster CRを確認します。

```

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

```

## 外部キー管理を内部キー管理に変更します

現在外部キー管理ツールを使用している場合は、内部キー管理ツールに変更できます。

### 手順

1. SoftwareEncryptionAtRest設定を削除してAstraDSCluster CRを変更します。
2. (オプション) 前のAstraeDSKeyProviderと関連付けられている秘密を削除します。



以前のキープロバイダとシークレットは自動的に削除されません。

## キーをローテーションしてセキュリティを確保します

キーのローテーションにより、セキュリティが向上します。デフォルトでは、Astraデータストアはキーを90日ごとに自動的にローテーションします。デフォルト設定を変更できます。また、必要に応じてキーをオンデマンドでローテーションすることもできます。

### 自動キーローテーションを設定する

1. CRDのAstraeSSEARKeyRotateパラメータを更新します。

```

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

```

オンデマンドのキーローテーションを設定する

1. AstraatDSSEARKeyRotateRequest CRを作成してキーを回転します。

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

## Astraデータストアライセンスを更新

Astraデータストアにインストールされた評価用ライセンスを更新して、評価期間を延長できます。ライセンスは、次の3つの方法のいずれかを使用して更新できます。

- Astra Control Centerを使用してAstraデータストアライセンスを更新するには、を参照してください ["ストレージバックエンドライセンスを更新する"](#)。
- Astra VMwareプラグインを使用してAstraデータストアライセンスを更新するには、を参照してください ["VMwareでAstraデータストアを管理"](#)。
- コマンドラインを使用してAstraデータストアライセンスを更新するには、を参照してください [\[Update the Astra Data Store license using the command line\]](#)。

### コマンドラインを使用してAstraデータストアライセンスを更新

'kubectl'ユーティリティを使用して'Astra Data Storeライセンスを更新できます

手順

1. ネットアップから入手した交換用ネットアップライセンスファイル（NLF）を適用します。コマンドを実行する前に、クラスタの名前（「<Astra-Data-Store-cluster-name>」）とライセンスファイルのパス（「<file\_path/file.txt」）を入力します。

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

2. ライセンスが追加されたことを確認します。

```
kubectl astrads license list
```

次のような応答が表示されます。

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

## Astraデータストアをアップグレード

Astraデータストアをアップグレードすると、最新の機能や修正を利用できます。Astra Data Storeの拡張機能を使用して'Astra Data Storeをアップグレードできます

### kubectlを使用してAstraデータストアをアップグレード

Astra Data Storeの「kubectl」拡張機能を使用して、Astraデータストアをアップグレードできます。

#### Astra Data Storeバンドルをダウンロードしてイメージを展開

手順

1. にログインします ["ネットアップサポートサイト"](#) Astra Data Storeバンドルをダウンロードします('Astra\_Data-Store\_22.05.tar')
2. (任意) 次のコマンドを使用して、バンドルのシグニチャを確認します。

```
openssl dgst -sha256 -verify Astra_Data_Store_2022.05.pub -signature
Astra_Data_Store_2022.05.sig 2022.12.01_ads.tar
```

3. ディレクトリを作成します。

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

4. 画像を抽出します。

```
tar -vxzf <path to tar file>/Astra_Data_Store_2022.05.tar
```



画像は、作業ディレクトリ内に作成された「astrs/images」ディレクトリに抽出されます。

バイナリをコピーし、ローカルレジストリにイメージをプッシュします

手順

1. イメージを抽出するために使用したディレクトリからkubectl-astrasバイナリをKubernetes kubectlバイナリがインストールされている標準パスにコピーします(/usr/bin/は下の例のパスとして使用されます)。kubectl-astrasは、Astraデータストアクラスターをインストールおよび管理するカスタムのkubectl拡張機

能です。



kubectlバイナリがインストールされているパスを検索するには'which kubectl'コマンドを使用します

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

## 2. Astra Data Storeイメージディレクトリ内のファイルをローカルレジストリに追加します。



以下の画像の自動ロードについては、サンプルスクリプトを参照してください。

### a. レジストリにログインします。

```
docker login [your_registry_path]
```

### b. 環境変数を、Astraデータストアイメージをプッシュするレジストリパス（例：repo.company.com）に設定します。

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

### c. 次のスクリプトを実行して、Dockerにイメージをロードし、イメージにタグを付け、ローカルレジストリにイメージをプッシュします。

```
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
```

## アップグレードを実行する

### 手順

#### 1. 'astraadsoperator.yaml'ファイルをローカルディレクトリにコピーします

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

#### 2. オペレータをアップグレードします。大文字の引数を環境に適した情報に置き換えます。

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

3. Astraデータストアのアップグレードを開始大文字の引数を環境に適した情報に置き換えます。

```
kubectl-astrads upgrade ads-version --repository-url REPOSITORY_URL
--ads-version-yaml ./astrads/manifests/astradsversion.yaml
```

アップグレードが開始されたことを示すメッセージが表示され、完了までに数分かかります。

## Astraデータストアを自動化されたスクリプトでアンインストールします

Astraデータストアとコントロールプレーンをアンインストールするには、ワークロード、バインド、ボリューム、エクスポートポリシー、Astraデータストアクラスタ、ライセンス、導入環境、およびAstraデータストアネームスペースを削除します。

アンインストールにはさまざまな方法があります。

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

## Astraデータストアを自動化されたスクリプトでアンインストールします

このプロセスでは、自動スクリプトを使用してAstraデータストアをアンインストールします。

必要なもの

- root 管理権限

Astra Data Storeのアンインストールプロセスでは、次の手順を実行できます。

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

既存のワークロードとバインドを削除します

Astraデータストアをアンインストールする前に、次のものを削除する必要があります

- Astraデータストアをストレージバックエンドとして使用するすべてのアプリケーションワークロード

- バックエンドとしてAstraデータストアを使用するTridentバインディング

これにより、Kubernetes 環境をクリーンな状態のまま維持できます。これは、を再インストールする場合に重要です。

### Astra データストアクラスタをアンインストールします

Astra Data Storeをアンインストールするには、ネットアップサポートサイトからダウンロードしたAstra Data Store tarファイルで「uninstall.sh」スクリプトを使用します。

1. 'マニフェスト'ディレクトリの'uninstall.sh'を探します
2. 次の「sed」コマンドを実行します。

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

3. 次のスクリプトを実行して、アンインストールする項目を指定します。

```
./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. クラスタのみをアンインストールする場合は、「uninstall.sh <cluster>」と入力します

それ以外の場合は、すべてをアンインストールするには、「uninstall.sh」と入力します



ほとんどの場合、すべてをアンインストールします。その後クラスタを再導入する場合は、クラスタだけをアンインストールすることもできます。

5. プロンプトで続行することを確認し、「erasedata」と入力します

対応：

```
./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   |
+-----+

```

### **astrs-system** 名前空間の削除を検証します

次のコマンドで結果が返されないことを確認します。

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

ワーカーノードでコンテナが実行されていないことを確認します

'FIRETAAP' や 'netwd' などのコンテナがワーカー・ノードで実行されていないことを確認します各ノードで次のコマンドを実行します。

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

### **OpenShift Container Platform** リソースを削除します

Red Hat OpenShift Container Platform (OCP) にAstraデータストアをインストールした場合は、OCPセキュリティコンテキスト制約 (SCC) と役割バインディングリソースをアンインストールできます。

OpenShift では、セキュリティコンテキスト制約 ( SCC ) を使用して、ポッドで実行できるアクションを制御します。

標準のアンインストールプロセスが完了したら、次の手順を実行します。

1. SCC リソースを削除します。

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

## 2. ロールバインドリソースを削除します

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



これらの手順で「resources not found」エラーを無視します。

## スクリプトを使用せずにAstraデータストアを手動でアンインストールする

このプロセスでは、スクリプトを使用せずにAstraデータストアを手動でアンインストールします。

自動スクリプトを使用せずにAstraデータストアを手動でアンインストールするには、ワークロード、バインド、ボリューム、エクスポートポリシー、クラスタ、ライセンス、導入環境、Astraデータストアネームスペース

必要なもの

- root 管理権限

Astra Data Storeのアンインストールプロセスでは、次の手順を実行できます。

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

既存のワークロードとバインドを削除します

Astraデータストアをアンインストールする前に、次のものを削除する必要があります

- Astraデータストアをストレージバックエンドとして使用するすべてのアプリケーションワークロード
- バックエンドとしてAstraデータストアを使用するTridentバインディング

これにより、Kubernetes 環境をクリーンな状態のまま維持できます。これは、を再インストールする場合に重要です。

**Astra**データストアクラスタとコントロールプレーンをアンインストールします

Astraデータストアを手動でアンインストールするには、次の手順に従います。

ボリュームとエクスポートポリシーを削除します

クラスタを削除する前に、Astraデータストアボリュームとエクスポートポリシーを削除する必要があります。



ボリュームとエクスポートポリシーを最初に削除しないと、Astra Data Storeボリュームオブジェクトが削除されるまで、クラスタの削除プロセスは一時停止します。クラスタの削除を開始する前に、それらの項目を削除の方が効率的です。

#### 手順

1. ボリュームを削除します。

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

2. エクスポートポリシーを削除します。

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

#### Astraデータストアクラスタを削除

クラスタを削除すると、Astra Data Storeクラスタオブジェクトのカスタムリソース（CR）とクラスタを対象としたリソースのみが削除される。



オペレータ、nodeinfo ポッド、およびクラスタコントローラ（Kubernetes を対象としたリソース）は、クラスタを削除しても削除されません。

クラスタを削除すると、基盤となるオペレーティング・システムもノードからアンインストールされ、'FIRETAAP' および 'netwd' サービスが停止します。

アンインストーラが完了するまでに約 1 分かかります。次に、Astraデータストアクラスタを対象としたリソースの削除が開始されます。

1. クラスタを削除します。

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

#### ライセンスを削除します

1. クラスタ内の各ワーカー・ノードに SSH 接続し、ワーカー・ノードで 'FIRETAAP' または 'netwd' が実行されていないことを確認します。
2. Astraデータストアライセンスを削除します。

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

## Astraデータストアのインストールを削除

クラスタ内のコントローラ、演算子、ネームスペース、およびサポートポッドを削除します。

1. Astra Data Storeインストールオブジェクトを削除します。

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

2. データストアDemonSetsとすべてのAstraデータストアコントローラリソースを削除します。

```
~% 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. 残りのアーティファクトと演算子 YAML ファイルを削除します。

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

## astrs-system 名前空間の削除を検証します

次のコマンドで結果が返されないことを確認します。

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

ワーカーノードでコンテナが実行されていないことを確認します

'FIRETAAP' や 'netwd' などのコンテナがワーカー・ノードで実行されていないことを確認します各ノードで次のコマンドを実行します。

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

## OpenShift Container Platform リソースを削除します

Red Hat OpenShift Container Platform (OCP) にAstraデータストアをインストールした場合は、OCPセキュリティコンテキスト制約 (SCC) と役割バインディングリソースをアンインストールできます。

OpenShift では、セキュリティコンテキスト制約 (SCC) を使用して、ポッドで実行できるアクションを制御します。

標準のアンインストールプロセスが完了したら、次の手順を実行します。

1. SCC リソースを削除します。

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

2. ロールバインドリソースを削除します

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



これらの手順で「resources not found errors」を無視します。

## 手動削除のサンプル

次に、手動アンインストールスクリプトの実行例を示します。

```
$ 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
"astradsgospolicies.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

```

## Astraデータストアのアンインストールプロセスのトラブルシューティング

アンインストールプロセスのトラブルシューティングが必要な場合は、次の点を確認してください。

ポッドが終了状態です

原因 ポッドがKubernetesで終了状態のままになる場合、Astraデータストアのアンインストールプロセスが必要になることがあります。

この問題が発生した場合は、次のコマンドを実行して、'astrs-system' 名前空間内のすべてのポッドを強制的に削除します。

```
kubect1 delete pods --all -n astrads-system --force --grace-period 0
```

**QoS**ポリシーは古いクラスタをポイントしています

Astraデータストアクラスタのみを削除して再導入した場合、サービス品質（QoS）ポリシーが古いクラスタをポイントしていて検出できないために、永続的ボリューム要求（PVC）またはボリュームを作成できないことがあります。

1. この問題を回避するには、Astraデータストアクラスタを削除したあとで、QoSポリシーを手動で削除します。

```
kubect1 delete AstraDSQosPolicy --all -A
```

2. （クラスタだけでなく）Astraデータストア環境全体を削除する。

```
uninstall.sh all
```

**Astra**データストアを削除またはアンインストールしたあとに、キープロバイダ**CRS**が削除されない

削除またはアンインストールするAstraデータストアクラスタ用に外部キープロバイダが設定されている場合は、削除されていないキープロバイダCRを手動でクリーンアップしなければならないことがあります。

## 例 1. 詳細

次の回避策 手順を使用します。

手順

1. キープロバイダCRSが削除されていないことを確認します。

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

対応：

NAME	AGE
externalkeyprovider1	94s

2. キープロバイダCRSを削除します。

- a. ファイナライザを取り外します。

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

- b. 以下のハイライトされたファイナライザラインを取り外します。

```
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. ファイナライザを削除したら、キープロバイダCRを削除します。

```

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

```

## 著作権情報

Copyright © 2022 NetApp, Inc. All rights reserved. 米国で印刷されていますこのドキュメントは著作権によって保護されています。画像媒体、電子媒体、および写真複写、記録媒体などの機械媒体など、いかなる形式および方法による複製も禁止します。テープ媒体、または電子検索システムへの保管-著作権所有者の書面による事前承諾なし。

ネットアップの著作物から派生したソフトウェアは、次に示す使用許諾条項および免責条項の対象となります。

このソフトウェアは、ネットアップによって「現状のまま」提供されています。ネットアップは明示的、または商品性および特定目的に対する適合性の暗示的保証を含み、かつこれに限定されないいかなる暗示的な保証も行いません。ネットアップは、いかなる場合でも、間接的、偶発的、特別、懲罰的、またはまたは結果的損害（代替品または代替サービスの調達、使用の損失、データ、利益、またはこれらに限定されないものを含みますが、これらに限定されません。）ただし、契約、厳格責任、または本ソフトウェアの使用に起因する不法行為（過失やその他を含む）のいずれであっても、かかる損害の可能性について知らされていた場合でも、責任の理論に基づいて発生します。

ネットアップは、ここに記載されているすべての製品に対する変更を随時、予告なく行う権利を保有します。ネットアップによる明示的な書面による合意がある場合を除き、ここに記載されている製品の使用により生じる責任および義務に対して、ネットアップは責任を負いません。この製品の使用または購入は、ネットアップの特許権、商標権、またはその他の知的所有権に基づくライセンスの供与とはみなされません。

このマニュアルに記載されている製品は、1 つ以上の米国特許、その他の国の特許、および出願中の特許により特許、その他の国の特許、および出願中の特許。

権利の制限について：政府による使用、複製、開示は、DFARS 252.227-7103（1988 年 10 月）および FAR 52-227-19（1987 年 6 月）の Rights in Technical Data and Computer Software（技術データおよびコンピュータソフトウェアに関する諸権利）条項の（c）（1）（ii）項、に規定された制限が適用されます。

## 商標情報

NetApp、NetAppのロゴ、に記載されているマーク <http://www.netapp.com/TM> は、NetApp, Inc.の商標です。その他の会社名と製品名は、それを所有する各社の商標である場合があります。