



# **Astra Data Store**를 사용하십시오

## Astra Data Store

NetApp  
July 19, 2022

# 목차

Astra Data Store를 사용하십시오 .....	1
kubectl 명령을 사용하여 Astra Data Store 미리 보기 자산을 관리합니다 .....	1
테스트 응용 프로그램을 배포합니다 .....	7
클러스터를 관리합니다 .....	11
Cloud Insights를 통해 메트릭을 모니터링합니다 .....	23
Prometheus 및 Grafana를 사용하여 메트릭을 모니터링합니다 .....	35
이벤트 로그를 구성하고 모니터링합니다 .....	37
Astra Data Store 미리 보기와 함께 Astra Control Center를 사용합니다 .....	38
자동 스크립트로 Astra Data Store 미리 보기를 제거합니다 .....	38
스크립트 없이 Astra Data Store 미리 보기를 제거합니다 .....	44

# Astra Data Store를 사용하십시오

## kubectl 명령을 사용하여 Astra Data Store 미리 보기 자산을 관리합니다

kubectl 명령을 사용하거나 Kubernetes API 확장을 사용하여 Astra Data Store 미리보기 자산을 관리할 수 있습니다.

샘플 앱을 배포하는 방법에 대한 자세한 내용은 [여기](#)를 참조하십시오 "테스트 응용 프로그램을 배포합니다".

다음 클러스터 유지 보수 정보는 [여기](#)를 참조하십시오 "클러스터를 관리합니다":

- 노드를 유지보수 모드로 전환합니다
- 드라이브를 교체합니다
- 노드를 추가합니다
- 노드를 교체합니다

무엇을 '필요로 할거야

- 에 설치한 Astra Data Store preview kubectl 플러그인 "[Astra Data Store 미리 보기를 설치합니다](#)"

## Astra Data Store 미리보기에 대한 Kubernetes 사용자 지정 API 리소스를 나열합니다

Kubernetes 내의 kubectl 명령을 사용하여 Astra Data Store 미리보기 클러스터와 상호 작용하고 상태를 관찰할 수 있습니다.

'api-resources' 명령에 나열된 각 항목은 Astra Data Store 미리 보기에서 클러스터 관리를 위해 내부적으로 사용하는 Kubernetes 사용자 정의 리소스 정의(CRD)를 나타냅니다.

이 목록은 나중에 표시된 것처럼 입력을 줄이기 위해 각 Astra Data Store 미리 보기 개체의 단축 이름을 가져오는 데 특히 유용합니다.

1. Astra Data Store 미리 보기를 위한 Kubernetes 사용자 정의 API 리소스 목록을 표시합니다.

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

응답:

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

2. Kubernetes 클러스터의 모든 현재 Astra Data Store 미리보기 객체를 가져오려면 "kubbeck get ads-a" 명령을 사용하십시오.

```
kubectl get ads -A
```

응답:

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/	created	arda-9.11.1	e000000009	10.224.8.146	46h

```

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. 간단한 이름 중 하나를 사용하여 클러스터에 있는 볼륨의 현재 상태를 표시합니다.

```
kubectl get adsvo -A
```

응답:

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

## kubeck 확장명에 대한 도움말 옵션을 사용합니다

kubbeck astrads 명령어에는 편의에 따라 사용 및 플래그 문서를 제공하는 '-h' 스위치가 포함되어 있습니다.

1. Astra Data Store preview kubctl extension의 모든 명령에 대한 도움말을 표시합니다.

```
kubectl astrads -h
```

응답:

```
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

groups.

<code>--cache-dir string</code>	Default HTTP cache directory (default <code>"/u/arda/.kube/http-cache"</code> )
<code>--certificate-authority string</code>	Path to a cert file for the certificate authority
<code>--client-certificate string</code>	Path to a client certificate file for TLS
<code>--client-key string</code>	Path to a client key file for TLS
<code>--cluster string</code>	The name of the kubeconfig cluster to use
<code>--context string</code>	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 string</code>	Path to the kubeconfig file to use for CLI requests.
<code>-n, --namespace string</code>	If present, the namespace scope for this CLI request
<code>--request-timeout string</code>	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).
<code>-s, --server string</code>	A value of zero means don't (default <code>"0"</code> ) The address and port of the Kubernetes API server
<code>--token string</code>	Bearer token for authentication to the API server
<code>--user string</code>	The name of the kubeconfig user to use

2. 명령에 대한 자세한 내용은 'astrads[command]--help'를 사용하십시오.

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

응답:

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:

<pre>-c, --component string</pre>	Specify the component to collect: [storage , controlplane , vasaprovider, all]
<pre>-d, --duration int</pre>	Duration is the duration in hours from the startTime for collection
<pre>-e, --event string</pre>	of AutoSupport. This should be a positive integer
<pre>(default "manual")</pre>	Specify the callhome event to trigger.
<pre>-f, --forceUpload</pre>	Configure an AutoSupport to upload if
<pre>it is in the compressed state</pre>	and not
<pre></pre>	uploading because it was created with
<pre>the 'local' option or if</pre>	automatic uploads of AutoSupports is
<pre>disabled</pre>	at the cluster level.
<pre>-h, --help</pre>	help for collect
<pre>-l, --local</pre>	Only collect and compress the
<pre>autosupport bundle. Do not upload</pre>	to support.
	Use 'download' to copy the collected



```

bundle after it is in
--nodes string          the 'compressed' state
                        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")

```

## 테스트 응용 프로그램을 배포합니다

Astra Data Store 미리 보기와 함께 사용할 수 있는 테스트 응용 프로그램을 배포하는 단계는 다음과 같습니다.

이 예에서는 Hrom 리포지토리를 사용하여 Bitnami의 MongoDB 차트를 배포합니다.

무엇을 '필요로 할거야

- Astra Data Store 미리보기 클러스터가 구축 및 구성되었습니다
- Trident 설치가 완료되었습니다

단계

1. Bitnami에서 Helm repo 추가:

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

2. MongoDB 구축:

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

3. MongoDB Pod의 상태를 확인합니다.

```

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

```

4. MongoDB에서 사용되는 영구 볼륨 클레임(PVC)을 확인합니다.

```
~% 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. kubelet 명령 'Get astradsvolume'을 사용하여 볼륨을 나열합니다.

```
~% 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. kubect 명령 describe astradsvolume을 사용하여 볼륨을 설명합니다.

```
~% 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

```

## 클러스터를 관리합니다

Astra Data Store 미리 보기와 함께 `kubctl` 명령을 사용하여 클러스터를 관리할 수 있습니다.

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

무엇을 '필요로 할거야

- `kubctl` 및 `kubctl-astrads` 플러그인이 설치된 시스템. 을 참조하십시오 ["Astra Data Store 미리 보기를 설치합니다"](#).

## 노드를 추가합니다

추가하려는 노드는 Kubernetes 클러스터의 일부여야 하며 클러스터의 다른 노드와 비슷한 구성을 가져야 합니다.

단계

1. 새 노드의 데이터 IP가 ADSCluster CR에 아직 포함되지 않은 경우 다음을 수행합니다.
  - a. `astradscluster` CR을 편집하고 ADS 데이터 네트워크 주소 필드에 추가 데이터 IP를 추가합니다.

```
~% kubectl edit astradscluster <cluster-name> -n astrads-system
```

응답:

```
adsDataNetworks:
  -addresses: dataIP1,dataIP2,dataIP3,dataIP4,*newdataIP*
```

- a. CR을 저장합니다.
- b. Astra Data Store 미리 보기 클러스터에 노드를 추가합니다.

```
~% kubectl astrads nodes add --cluster <cluster-name>
```

2. 그렇지 않으면 노드를 추가하기만 하면 됩니다.

```
~% kubectl astrads nodes add --cluster <cluster-name>
```

3. 노드가 추가되었는지 확인합니다.

```
~% kubectl astrads nodes list
```

## 노드를 유지보수 모드로 전환합니다

호스트 유지보수 또는 패키지 업그레이드를 수행해야 하는 경우 노드를 유지보수 모드로 전환해야 합니다.



노드는 이미 Astra Data Store 미리 보기 클러스터에 포함되어 있어야 합니다.

노드가 유지보수 모드일 때는 클러스터에 노드를 추가할 수 없습니다. 이 예제에서는 노드 "nhcitj1525"를 유지 관리 모드로 설정합니다.

단계

1. 노드 세부 정보를 표시합니다.

```
~% kubectl get nodes
```

응답:

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. 노드가 유지보수 모드가 아닌지 확인합니다.

```
~% kubectl astrads maintenance list
```

응답(유지보수 모드에 있는 노드가 이미 없음):

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

3. 유지보수 모드를 활성화합니다.

```
~% kubectl astrads maintenance create <cr-name> --node-name=<<node-name>> --variant=Node
```

샘플:

```
~% kubectl astrads maintenance create maint1 --node-name="nhcitjj1525"
--variant=Node
Maintenance mode astrads-system/maint1 created
```

4. 노드를 나열합니다.

```
~% kubectl astrads nodes list
```

응답:

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

5. 유지보수 모드의 상태를 점검합니다.

```
~% kubectl astrads maintenance list
```

응답:

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

유지보수 모드는 거짓으로 시작해 참으로 바뀝니다. 유지 보수 상태가 PreparingForMaintenance에서 ReadyforMaintenance로 바뀝니다.

6. 노드 유지보수가 완료된 후 유지보수 모드를 비활성화합니다.

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

7. 노드가 더 이상 유지보수 모드가 아닌지 확인합니다.

```
~% kubectl astrads maintenance list
```

## 노드를 교체합니다

Astra Data Store 미리 보기와 함께 kubectl 명령을 사용하여 클러스터에서 장애가 발생한 노드를 교체합니다.

단계

1. 모든 노드 나열:

```
~% kubectl astrads nodes list
```

응답:

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

2. 클러스터 설명:

```
~% kubectl astrads clusters list
```



응답:

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

3. 노드 HA가 장애가 발생한 노드에서 FALSE로 표시되는지 확인합니다.

```
~% kubectl describe astradscluster -n astrads-system
```

응답:

```
Name:          cluster-multinodes-21209
Namespace:     astrads-system
Labels:        <none>
Annotations:   kubectl.kubernetes.io/last-applied-configuration:

{"apiVersion":"astrads.netapp.io/v1alpha1","kind":"AstraDSCluster","meta
data":{"annotations":{},"name":"cluster-multinodes-21209","namespa...
API Version:   astrads.netapp.io/v1alpha1
Kind:          AstraDSCluster

State:          Disabled
Variant:        None
Node HA:        false
Node ID:        4
Node Is Reachable: false
Node Management IP: 172.21.192.192
Node Name:      sti-rx2540-532d.ct1.gdl.englab.netapp.com
Node Role:      Storage
Node UUID:      6f6b88f3-8411-56e5-b1f0-a8e8d0c946db
Node Version:   12.75.0.6167444
Status:         Added
```

4. "AdsNode Count"의 값을 3으로 감소함으로써 실패한 노드를 제거하도록 Astra클러스터 CR을 수정합니다.

```
cat manifests/astradscluster.yaml
```

응답:

```
apiVersion: astrads.netapp.io/v1alpha1
kind: AstraDSCluster
metadata:
```

```

name: cluster-multinodes-21209
namespace: astrads-system
spec:
  # ADS Node Configuration per node settings
  adsNodeConfig:
    # Specify CPU limit for ADS components
    # Supported value: 9
    cpu: 9
    # Specify Memory Limit in GiB for ADS Components.
    # Your kubernetes worker nodes need to have at least this much RAM
    free
    # for ADS to function correctly
    # Supported value: 34
    memory: 34
    # [Optional] Specify raw storage consumption limit. The operator
    # will only select drives for a node up to this limit
    capacity: 600
    # [Optional] Set a cache device if you do not want auto detection
    # e.g. /dev/sdb
    # cacheDevice: ""
    # Set this regex filter to select drives for ADS cluster
    # drivesFilter: ".*"

    # [Optional] Specify node selector labels to select the nodes for
    # creating ADS cluster
    # adsNodeSelector:
    #   matchLabels:
    #     customLabelKey: customLabelValue

    # Specify the number of nodes that should be used for creating ADS
    # cluster
    adsNodeCount: 3

    # Specify the IP address of a floating management IP routable from any
    # worker node in the cluster
    mvip: "172..."

    # Comma separated list of floating IP addresses routable from any host
    # where you intend to mount a NetApp Volume
    # at least one per node must be specified
    # addresses: 10.0.0.1,10.0.0.2,10.0.0.3,10.0.0.4,10.0.0.5
    # netmask: 255.255.255.0
    adsDataNetworks:
      - addresses: "172..."
        netmask: 255.255.252.0

```

```

# [Optional] Provide a k8s label key that defines which protection
domain a node belongs to
# adsProtectionDomainKey: ""

# [Optional] Provide a monitoring config to be used to setup/configure
a monitoring agent.
monitoringConfig:
  namespace: "netapp-monitoring"
  repo: "docker.repo.eng.netapp.com/global/astra"

autoSupportConfig:
  # AutoUpload defines the flag to enable or disable AutoSupport
  upload in the cluster (true/false)
  autoUpload: true
  # Enabled defines the flag to enable or disable automatic
  AutoSupport collection.
  # When set to false, periodic and event driven AutoSupport
  collection would be disabled.
  # It is still possible to trigger an AutoSupport manually while
  AutoSupport is disabled
  # enabled: true
  # CoredumpUpload defines the flag to enable or disable the upload of
  coredumps for this ADS Cluster
  # coredumpUpload: false
  # HistoryRetentionCount defines the number of local (not uploaded)
  AutoSupport Custom Resources to retain in the cluster before deletion
  historyRetentionCount: 25
  # DestinationURL defines the endpoint to transfer the AutoSupport
  bundle collection
  destinationURL: "https://testbed.netapp.com/put/AsupPut"
  # ProxyURL defines the URL of the proxy with port to be used for
  AutoSupport bundle transfer
  # proxyURL:
  # Periodic defines the config for periodic/scheduled AutoSupport
  objects
  periodic:
    # Schedule defines the Kubernetes Cronjob schedule
    - schedule: "0 0 * * *"
    # PeriodicConfig defines the fields needed to create the
    Periodic AutoSupports
    periodicconfig:
      - component:
          name: storage
          event: dailyMonitoring
          userMessage: Daily Monitoring Storage AutoSupport bundle
          nodes: all

```

```
- component:
  name: controlplane
  event: daily
  userMessage: Daily Control Plane AutoSupport bundle
```

5. 클러스터에서 노드가 제거되었는지 확인합니다.

```
~% kubectl get nodes --show-labels
```

응답:

NAME	STATUS	ROLES
sti-astramaster-237	Ready	control-plane,master
sti-rx2540-532d	Ready	<none>
sti-rx2540-533d	Ready	<none>

```
~% kubectl astrads nodes list
```

응답:

NODE NAME	NODE STATUS	CLUSTER NAME
sti-rx2540-534d	Added	cluster-multinodes-21209
sti-rx2540-535d	Added	cluster-multinodes-21209
sti-rx2540-536d	Added	cluster-multinodes-21209

```
~% kubectl get nodes --show-labels
```

응답:

NAME	STATUS	ROLES	AGE	VERSION
sti-astramaster-237	Ready	control-plane,master	24h	v1.20.0
sti-rx2540-532d	Ready	<none>	24h	v1.20.0

```
~% kubectl describe astradscluster -n astrads-system
```

응답:

```
Name:          cluster-multinodes-21209
Namespace:     astrads-system
Labels:        <none>
Kind:          AstraDSCluster
Metadata:
...
```

6. 클러스터 CR을 수정하여 교체할 노드를 클러스터에 추가합니다. 노드 수는 4까지 증가합니다. 추가를 위해 새 노드가 선택되었는지 확인합니다.

```
rvi manifests/astradscluster.yaml
cat manifests/astradscluster.yaml
apiVersion: astrads.netapp.io/v1alpha1
kind: AstraDSCluster
metadata:
  name: cluster-multinodes-21209
  namespace: astrads-system
```

```
~% kubectl apply -f manifests/astradscluster.yaml
```

응답:

```
astradscluster.astrads.netapp.io/cluster-multinodes-21209 configured
```

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

응답:

NAME	READY	STATUS	RESTARTS	AGE
astrads-cluster-controller...	1/1	Running	1	24h
astrads-deployment-support...	3/3	Running	0	24h
astrads-ds-cluster-multinodes-21209	1/1	Running		

```
~% kubectl astrads nodes list
```

응답:

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

```
~% kubectl astrads clusters list
```

응답:

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

```
~% kubectl astrads drives list
```

응답:

DRIVE NAME	DRIVE ID	DRIVE STATUS	NODE NAME	CLUSTER NAME
scsi-36000...	c3e197f2...	Active	sti-rx2540...	cluster-multinodes-21209

## 드라이브를 교체합니다

클러스터에서 드라이브가 고장난 경우 데이터 무결성을 보장하기 위해 가능한 한 빨리 드라이브를 교체해야 합니다. 드라이브에 장애가 발생하면 클러스터 CR 노드 상태, 클러스터 상태 정보 및 메트릭 끝점에서 장애가 발생한 드라이브 정보를 볼 수 있습니다.

노드 **Statuses.driveStatuses**에서 장애가 발생한 드라이브를 표시하는 클러스터의 예

```
$ kubectl get adscl -A -o yaml
```

응답:

```

...
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
  ...

```

장애가 발생한 드라이브 CR은 장애가 발생한 드라이브의 UUID에 해당하는 이름으로 클러스터에 자동으로 생성됩니다.

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

응답:

```

...
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.ctl.gdl.englab.netapp.com
  state: ReadyToReplace

```

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

응답:

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

단계

1. 교체 제한 사항에 맞는 드라이브를 필터링하는 "kubbeck astrads show-replacement" 명령을 사용하여 가능한 교체 드라이브를 나열합니다(클러스터에서 사용되지 않음, 마운트되지 않음, 파티션 없음, 오류가 발생한 드라이브보다 크거나 같음).

가능한 대체 드라이브를 필터링하지 않고 모든 드라이브를 나열하려면 'show-replacement' 명령에 '--all'을 추가합니다.

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

응답:

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

2. "replace" 명령을 사용하여 드라이브를 전달된 일련 번호로 교체합니다. 명령이 대체를 완료하거나, '--wait' 시간이 경과되면 실패합니다.

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



부적절한 일련 번호를 사용하여 kubbtl astrads faileddrive replace를 실행하면 다음과 같은 오류가 나타납니다.



```
~% 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. 드라이브 교체를 다시 실행하려면 이전 명령으로 '--force'를 사용하십시오.

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

를 참조하십시오

- ["kubectl 명령을 사용하여 Astra Data Store 미리 보기 자산을 관리합니다"](#)

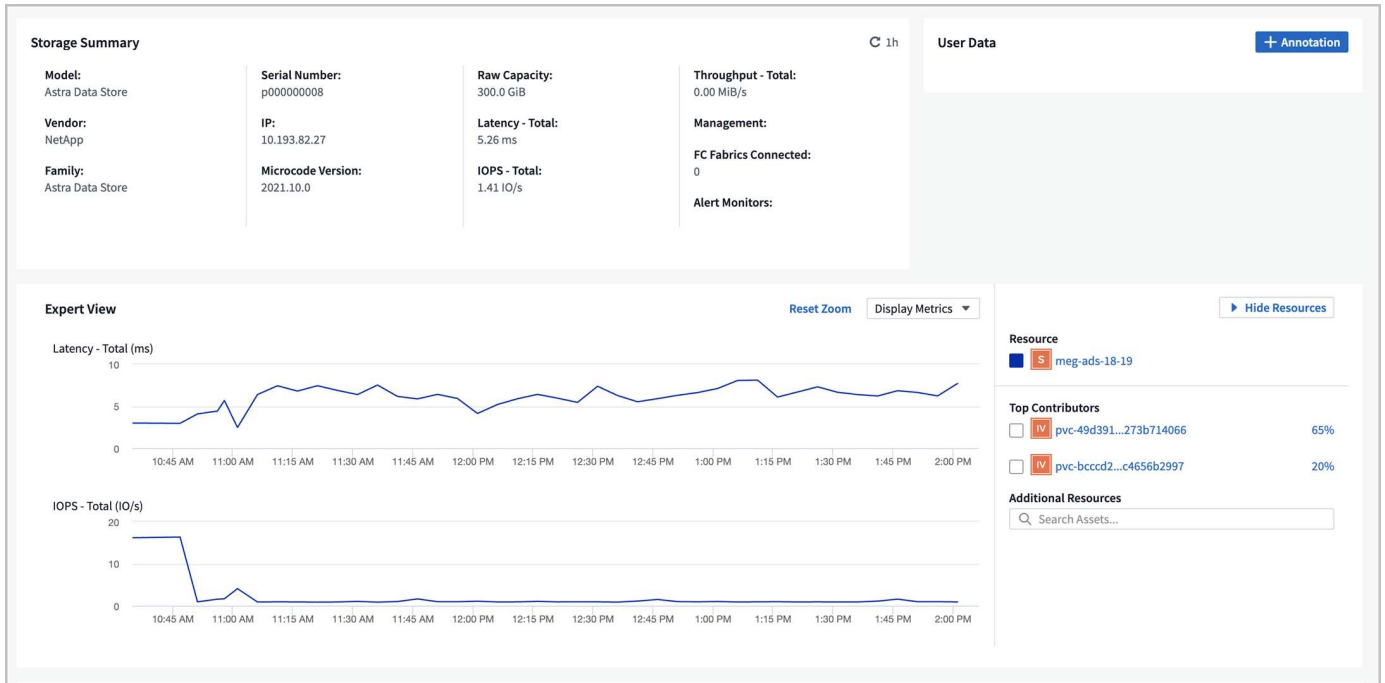
## Cloud Insights를 통해 메트릭을 모니터링합니다

Cloud Insights를 사용하여 Astra 데이터 저장소 미리 보기 메트릭을 모니터링할 수 있습니다.

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

다음은 Cloud Insights에 표시되는 몇 가지 Astra 데이터 저장소 미리 보기 메트릭입니다

.



을 사용하여 Astra Data Store 미리 보기에서 생성된 메트릭 목록을 표시할 수도 있습니다 [\[Open Metrics API help\]](#).

## Cloud Insights 연결 필수 작업을 완료합니다

Astra 데이터 저장소를 Cloud Insights와 연결하기 전에 다음 작업을 완료해야 합니다.

- ["Astra Data Store Monitoring Operator를 설치합니다"](#) 이것은 Astra Data Store 미리 보기 설치 지침의 일부입니다.
- ["kubbeck-astrads 바이너리를 설치합니다"](#) 이것은 Astra Data Store 미리 보기 설치 지침의 일부입니다.
- ["Cloud Insights 계정을 만듭니다"](#).
- 'awk, curl, grep', 'jq' 명령을 사용할 수 있는지 확인합니다

다음 정보를 수집합니다.

- 획득 장치, 데이터 수집, 데이터 수집 및 로그 수집 등의 범주에 대한 읽기/쓰기 권한이 있는 \* Cloud Insights API 액세스 토큰 \*. 이 작업은 읽기/쓰기 작업, 획득 장치 설정 및 데이터 수집 프로세스 설정에 사용됩니다.
- \* Kubernetes API 서버 IP 주소 및 포트 \*. Astra Data Store 미리보기 클러스터를 모니터링하는 데 사용됩니다.
- \* Kubernetes API 토큰 \*. Kubernetes API를 호출하는 데 사용됩니다.
- \* 영구 볼륨 구성 \*. 영구 볼륨의 프로비저닝 방법에 대한 정보입니다.

## 획득 장치 저장

획득 장치는 설치 파일, 구성 데이터 및 로그를 저장하기 위해 세 개의 영구 볼륨이 필요합니다. Monitoring Operator는 기본 스토리지 클래스를 사용하여 영구 볼륨 클레임을 생성합니다. 설치 프로그램 스크립트를 실행할 때 '-s' 옵션을 사용하여 다른 스토리지 클래스 이름을 지정할 수 있습니다.

Kubernetes 클러스터에 스토리지 공급자(예: NetApp Trident)가 없으면 설치 관리자 스크립트를 실행할 때 '-r' 옵션을 사용하여 로컬 파일 시스템 경로를 제공할 수 있습니다. '-r' 옵션이 설정되면 설치 프로그램 스크립트는 제공된 디렉토리 내에 세 개의 영구 볼륨을 생성합니다. 이 디렉토리에는 최소 150GB의 여유 공간이 필요합니다.

## 설치 스크립트를 다운로드하고 실행합니다

Cloud Insights는 모니터링 오퍼레이터를 통해 Astra 데이터 저장소 미리 보기 모니터링을 활성화하는 Bash 스크립트를 제공합니다. 설치 스크립트는 Astra Data Store Collector, Telegraf 에이전트 및 Fluent Bit Agent와 함께 획득 장치를 설치합니다.

Cloud Insights 테넌트 도메인 이름과 선택한 Cloud Insights API 액세스 토큰은 다운로드될 때 설치 프로그램 스크립트에 포함됩니다.

그런 다음 다음과 같이 메트릭이 전송됩니다.

- Telegraf는 Cloud Insights 데이터 레이크에 메트릭을 전송합니다.
- Fluent bit는 로그 수집 서비스에 로그를 보냅니다.

설치 프로그램 스크립트 도움말을 표시합니다

설치 프로그램 스크립트에 대한 전체 도움말 텍스트가 아래에 나와 있습니다.

설치 프로그램 스크립트 도움말 텍스트 표시:

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

응답:

```
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.
```

설치 스크립트를 실행합니다


1. Cloud Insights 계정이 없는 경우 계정을 만듭니다.
2. Cloud Insights에 로그인합니다.

3. Cloud Insights 메뉴에서 \* Admin \* > \* Data Collector \* 를 클릭합니다.
4. 새 수집기를 추가하려면 \* + Data Collector \* 를 클릭합니다.




5. Astra Data Store \* 타일을 클릭합니다.
6. 올바른 Cloud Insights API 액세스 토큰을 선택하거나 새 토큰을 생성하십시오.
7. 지침에 따라 설치 프로그램 스크립트를 다운로드하고, 권한을 업데이트하고, 스크립트를 실행합니다.


이 스크립트에는 Cloud Insights 테넌트 URL 및 선택한 Cloud Insights API 액세스 토큰이 포함되어 있습니다.



Select a Data Collector



Configure Collector



**NetApp**  
Astra Data Store

## Configure Collector

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

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

Kubernetes

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

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

[+ API Access Token](#)

[Production Best Practices ?](#)

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

- 1

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

Copy Installer Script

☐ Reveal Installer Script

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

CI_DOMAIN_NAME="f49uaky.gstabler-ads.cloudinsights-dev.netapp.com"
CI_API_KEY="eyJraWQ1OiI5OTk5IiwidHlwIjoIc3R5bWVudF9hZHNfYXBPX2t1eTEgKG9uIGJlaGFsZiBvZiBhZG1pbik"

```
- 3

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

Copy Permissions Command

☐ Reveal Permissions Command

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

```
- 5

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

Copy Install Command

☐ Reveal Install Command

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

```
- 7

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

Complete Setup

8. 스크립트가 완료되면 \* 설치 완료 \* 를 클릭합니다.

설치 스크립트가 완료되면 데이터 소스 목록에 Astra Data Store Collector가 나타납니다.



오류로 인해 스크립트가 종료되면 나중에 오류가 해결되면 스크립트를 다시 실행할 수 있습니다. 이 스크립트는 환경에서 기본 설정을 사용하지 않는 경우 모니터링 운영자 네임스페이스 및 Kubernetes API 서버 포트와 같은 추가 매개 변수를 지원합니다. 사용법과 도움말 텍스트를 보려면 의 ``h' 옵션을 사용하십시오./cloudinsights-ads-monitoring.sh -h.

설치 스크립트는 구성이 성공할 때 다음과 같은 출력을 생성합니다.

```
Configuring Cloud Insights monitoring for Astra Data Store . . .
Configuring monitoring namespace
...
Configuring output sink and Fluent Bit plugins
Configuring Telegraf 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!
```

## 상담원 CR의 예

다음은 설치 프로그램 스크립트를 실행한 후 Monitoring-NetApp 에이전트 CR이 어떻게 보일지에 대한 예입니다.

```
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
telegraf:
- name: ads-open-metric
  outputs:
  - sink: CI
  run-mode:
  - ReplicaSet
  substitutions:
  - key: URLS
    values:
    - http://astrads-metrics-service.astrads-
system.svc.cluster.local:9341
  - key: METRIC_TYPE
    value: ads-metric
  - key: ADS_CATEGORY
    value: netapp_ads
  - key: ADS_CLUSTER_NAME
    value: meg-ads-21-22-28-29-30
- name: agent
status:
  au-pod-status: UP
  au-uuid: eddeccc6-3aa3-4dd2-a98c-220085fae6a9

```

## Cloud Insights 연결을 편집합니다

나중에 Kubernetes API 토큰 또는 Cloud Insights API 액세스 토큰을 편집할 수 있습니다.

- Kubernetes API 토큰을 업데이트하려면 Cloud Insights UI에서 Astra Data Store Collector를 편집해야 합니다.
- 원격 측정 및 로그에 사용되는 Cloud Insights API 액세스 토큰을 업데이트하려면 kubctl 명령을 사용하여 모니터링 오퍼레이터 CR을 편집해야 합니다.

## Kubernetes API 토큰을 업데이트합니다

1. Cloud Insights에 로그인합니다.
2. Admin \* > \* Data Collector \* 를 선택하여 Data Collector 페이지에 액세스합니다.
3. Astra Data Store 클러스터의 항목을 찾습니다.
4. 페이지 오른쪽에 있는 메뉴를 클릭하고 \* 편집 \* 을 선택합니다.
5. Kubernetes API 토큰 필드를 새 값으로 업데이트합니다.
6. Collector 저장 \* 을 선택합니다.

## Cloud Insights API 액세스 토큰을 업데이트합니다

1. Cloud Insights에 로그인합니다.
2. 관리자 \* > \* API 액세스 \* 를 선택하고 \* + API 액세스 토큰 \* 을 클릭하여 새 Cloud Insights API 액세스 토큰을 만듭니다.
3. 상담원 CR 편집:

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

4. 출력 싱크 섹션을 찾아 이름이 'CI'인 항목을 찾습니다.
5. 'api-key'라는 레이블의 경우 현재 값을 새 Cloud Insights API 액세스 토큰으로 바꿉니다.

섹션은 다음과 같이 표시됩니다.

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

6. 편집기 창을 저장하고 종료합니다.

모니터링 운영자는 새로운 Cloud Insights API 액세스 토큰을 사용하도록 Telegraf 및 Fluent 비트를 업데이트합니다.

## Cloud Insights와의 연결을 해제합니다

Cloud Insights와의 연결을 끊으려면 먼저 Cloud Insights UI에서 Astra 데이터 저장소 수집기를 삭제해야 합니다. 이 작업이 완료되면 모니터링 작동기에서 획득 장치, 텔레그라프 및 Fluent 비트 구성을 제거할 수 있습니다.

## Astra Data Store 미리 보기 수집기를 제거합니다

1. Cloud Insights에 로그인합니다.
2. Admin \* > \* Data Collector \* 를 선택하여 Data Collector 페이지에 액세스합니다.
3. Astra Data Store 클러스터의 항목을 찾습니다.



4. 화면 오른쪽의 메뉴를 선택하고 \* Delete \* 를 선택합니다.

5. 확인 페이지에서 \* 삭제 \* 를 클릭합니다.

획득 장치, 텔레그라프 및 **Fluent** 비트를 제거합니다

1. 상담원 CR 편집:

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

2. au 섹션을 찾아 IsEnabled를 false로 설정합니다

3. '유창한 비트' 섹션을 찾아 ads-tail-CI라는 플러그인을 제거합니다. 플러그인이 더 이상 없으면 "fluent-bit" 섹션을 제거할 수 있습니다.

4. Telegraf 섹션을 찾아 ads-open-metric이라는 플러그인을 제거합니다. 플러그인이 더 이상 없으면 Telegraf 섹션을 제거할 수 있습니다.

5. 출력 싱크 섹션을 찾아 'CI'라는 싱크를 제거합니다.

6. 편집기 창을 저장하고 종료합니다.

모니터링 오퍼레이터는 Telegraf 및 Fluent 비트 구성을 업데이트하고 획득 장치 포드를 삭제합니다.

7. Storage Provisioner 대신 Acquisition Unit PVS에 로컬 디렉토리를 사용한 경우 PVS를 삭제합니다.

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

그런 다음 획득 장치가 실행 중인 노드에서 실제 디렉토리를 삭제합니다.

8. 획득 장치 포드가 삭제된 후 Cloud Insights에서 획득 장치를 삭제할 수 있습니다.

- Cloud Insights 메뉴에서 \* Admin \* > \* Data Collector \* 를 선택합니다.
- Acquisition Units(획득 단위) \* 탭을 클릭합니다.
- 획득 장치 포드 옆에 있는 메뉴를 클릭합니다.
- 삭제 \* 를 선택합니다.

모니터링 작업자는 Telegraf 및 Fluent 비트 구성을 업데이트하고 획득 장치를 제거합니다.

## 메트릭 **API** 도움말을 엽니다

다음은 Astra Data Store 미리 보기에서 메트릭을 수집하는 데 사용할 수 있는 API 목록입니다.

- "도움말" 줄에 메트릭이 설명되어 있습니다.
- "유형" 선은 메트릭이 게이지 또는 카운터인지 여부를 나타냅니다.

```
# 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 및 Grafana를 사용하여 메트릭을 모니터링합니다

Prometheus 및 Grafana를 사용하여 Astra Data Store 미리보기 메트릭을 모니터링할 수 있습니다. Ameta Data Store 미리보기 Kubernetes 클러스터 메트릭 엔드포인트에서 메트릭을 수집하도록 Prometheus를 구성할 수 있으며 Grafana를 사용하여 메트릭 데이터를 시각화할 수 있습니다.

무엇을 '필요로 할거야

- Astra Data Store 미리 보기 클러스터 또는 Astra Data Store 미리 보기 클러스터와 통신할 수 있는 다른 클러스터에 Prometheus 및 Grafana 패키지를 다운로드하여 설치했는지 확인합니다. 공식 설명서의 지침에 따라 각 도구를 설치합니다.
  - ["Prometheus를 설치합니다"](#)
  - ["Grafana를 설치합니다"](#)
- Prometheus 및 Grafana는 Astra Data Store 미리보기 Kubernetes 클러스터와 통신할 수 있어야 합니다. Prometheus와 Grafana가 Astra Data Store 미리보기 클러스터에 설치되어 있지 않은 경우, Astra Data Store 미리보기 클러스터에서 실행 중인 메트릭 서비스와 통신할 수 있는지 확인해야 합니다.

## Prometheus를 구성합니다

Astra Data Store Preview는 Kubernetes 클러스터의 TCP 포트 9341에 메트릭 서비스를 노출합니다. 이 서비스에서 메트릭을 수집하려면 Prometheus를 구성해야 합니다.

단계

1. Prometheus 설치를 위해 'Prometheus.yml' 설정 파일을 편집합니다.
2. Astra Data Store 미리 보기 서비스 이름 및 해당 포트를 가리키는 서비스 목표를 추가합니다. 예를 들면 다음과 같습니다.

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

3. Prometheus 서비스를 시작합니다.

## Grafana 구성

Grafana를 구성하여 Prometheus에서 수집한 메트릭을 표시할 수 있습니다.

단계

1. Grafana 설치를 위해 'datasources.yaml' 구성 파일을 편집합니다.
2. Prometheus를 데이터 소스로 추가합니다. 예를 들면 다음과 같습니다.

```
apiVersion: 1

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

3. Grafana 서비스를 시작합니다.
4. 에 대한 Grafana 설명서의 지침을 따릅니다 ["시작하십시오"](#).

## Grafana 대시보드 템플릿을 가져옵니다

Astra Data Store 미리 보기를 설치하기 위해 다운로드한 번들 파일에는 Grafana 내에서 가져올 수 있는 Grafana 대시보드 템플릿 파일이 포함되어 있습니다. 이러한 대시보드 템플릿을 사용하면 Astra Data Store 미리 보기에서 사용할 수 있는 메트릭의 유형과 이를 보는 방법을 확인할 수 있습니다.

단계

1. Astra Data Store preview '.tar.gz' bundle을 엽니다.
2. 'manifests' 디렉터리를 엽니다.
3. graana\_cluster.json과 graana\_volume.json 파일을 추출한다.
4. Grafana 웹 UI 사용, "[에서 대시보드 템플릿 파일을 Grafana로 가져옵니다](#)".

## 이벤트 로그를 구성하고 모니터링합니다

EMS(이벤트 관리 시스템) 로그를 모니터링하려면 다음과 같은 높은 수준의 작업을 수행해야 합니다.

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

### Astra 데이터 저장소 미리 보기 클러스터 사용자 지정 리소스(CR)에서 모니터링 구성

Astra Data Store preview cluster CR에 모니터링 옵션이 구성되어 있지 않은 경우 "Astrads" 확장을 사용하여 설정할 수 있습니다.

입력:

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

여기서,

- 설치된 에이전트의 네임스페이스: 모니터링 에이전트의 기본 이름인 모니터링 에이전트의 네임스페이스를 입력합니다. 모니터링 오퍼레이터의 경우 NetApp CR입니다.
- '-r'은 Fluent 또는 Telegraf 이미지가 있는 Docker 레지스트리를 설정하는 데 선택 사항입니다. 기본적으로 경로는 변경할 수 있는 dddocker.repo.eng.netapp.com/global/astra`로 설정됩니다.

### Cloud Insights를 설정합니다

로그를 보려면 Cloud Insights 설정은 선택 사항이지만 Cloud Insights를 사용하여 데이터를 보는 것이 좋습니다. 을 참조하십시오 "[NetApp Cloud Insights 설정 방법](#)" Astra 데이터 저장소 미리 보기와 함께 사용할 수 있습니다.

### 이벤트 로그를 Elastic로 스트리밍합니다

EMS 이벤트 및 기타 POD 로그를 Elastic 등의 타사 엔드포인트로 스트리밍하려면 "Astrads" 확장을 사용합니다.

입력:

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



탄성 호스트 이름은 IP 주소일 수 있습니다.

## Astra Data Store 미리 보기와 함께 Astra Control Center를 사용합니다

Astra Control Center UI(사용자 인터페이스)를 사용하여 Astra Data Store 미리 보기 작업을 수행할 수 있습니다.

### Astra Data Store 미리 보기를 위한 Astra Control Center를 설정합니다

Astra Data Store 미리 보기를 위한 Astra Control Center UI를 사용하려면 다음 작업을 완료해야 합니다.

- "Astra Data Store를 실행하는 기본 Kubernetes 클러스터를 Astra Control Center에 추가합니다".
- "Astra Data Store 미리 보기를 Astra Control Center에 스토리지 백엔드로 추가합니다".



스토리지 백엔드를 추가하고 Astra Data Store 미리 보기가 있는 Kubernetes 클러스터가 없는 경우, 먼저 클러스터를 추가해야 합니다.

### Astra Control Center에서 수행할 수 있는 작업

Astra Data Store 미리 보기를 위한 Astra Control Center를 설정한 후 Astra Control Center UI를 사용하여 다음 작업을 수행할 수 있습니다.

- "Astra Control Center를 사용하여 Astra Data Store Preview 자산의 상태를 모니터링합니다".
- "Astra Data Store 미리 보기 백엔드 스토리지를 관리합니다".
- "노드, 디스크 및 지속적인 PVC(Volume Claim) 모니터링".

### 를 참조하십시오

- "Astra 제품군 소개"
- "Astra Control Center 문서"
- "Astra Control API를 참조하십시오"

## 자동 스크립트로 Astra Data Store 미리 보기를 제거합니다

Astra Data Store 미리 보기 및 컨트롤 플레인을 제거하려면 워크로드, 바인딩, 볼륨, 내보내기 정책, Astra Data Store 클러스터, 라이선스, 배포 환경 및 Astra Data Store 미리 보기 네임스페이스를 제거합니다.

또는 을(를) 사용할 수 있습니다 "스크립트 없이 Astra Data Store 미리 보기를 제거합니다".

무엇을 '필요로 할거야

- 루트 관리 권한

Astra Data Store 미리 보기 제거 프로세스는 다음과 같은 고급 단계를 안내합니다.

- [Remove existing workloads and bindings]



- [\[Uninstall Astra Data Store cluster\]](#)
- [\[Validate the removal of the astrads-system namespace\]](#)
- [\[Ensure containers are not running on worker nodes\]](#)
- [\[Delete OpenShift Container Platform resources\]](#)
- [\[Troubleshoot the Astra Data Store preview uninstall process\]](#)

## 기존 워크로드 및 바인딩을 제거합니다

Astra Data Store 미리 보기를 제거하기 전에 먼저 다음을 제거해야 합니다

- Astra Data Store 미리 보기를 스토리지 백엔드로 사용하는 모든 애플리케이션 워크로드
- Astra Data Store 미리 보기를 백엔드로 사용하는 Trident 바인딩

이렇게 하면 Kubernetes 환경이 클린 상태로 유지됩니다. 이는 다시 설치할 때 중요합니다.

## Astra Data Store 클러스터를 제거합니다

Astra Data Store 미리 보기를 제거하려면 NetApp Support 사이트에서 다운로드한 Astra Data Store tar 파일에서 `uninstall.sh` 스크립트를 사용하십시오.

1. 'manifests' 디렉토리에서 `uninstall.sh`를 찾습니다.
2. 다음 'ed' 명령을 실행합니다.

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

3. 제거할 항목을 나타내는 다음 스크립트를 실행합니다.

```
./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. 클러스터를 설치 제거하려면 "`uninstall.sh <cluster>`"를 입력하십시오

그렇지 않으면 모든 항목을 제거하려면 `uninstall.sh`를 입력합니다



대부분의 경우 모든 항목을 제거합니다. 클러스터를 나중에 다시 배포하려는 경우 클러스터만 제거하면 됩니다.

5. 프롬프트에서 계속 진행할지 확인하고 `erasedata`를 입력합니다

응답:

---

```
./uninstall.sh all
```

```
Enter 'erasedata' to confirm you want proceed with the uninstall:  
erasedata
```

```
+-----+  
| Wed Feb  2 10:14:01 EST 2022 |  
| ADS cluster uninstall started |  
+-----+
```

```
Deleting astradsvolumes  
Deleted astradsvolumes  
Deleting astradsexportpolicies  
Deleted astradsexportpolicies  
Deleting astradsvolumesnapshots  
Deleted astradsvolumesnapshots  
Deleting astradsclusters  
Deleted astradsclusters  
Deleting astradslicenses  
Deleted astradslicenses
```

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

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

```
Removing astradsversion  
astradsversion.astrads.netapp.io "astradsversion" deleted  
Removed astradsversion  
Removing daemonsets  
daemonset.apps "astrads-ds-nodeinfo-astradsversion" deleted  
Removed daemonsets  
Removing deployments  
deployment.apps "astrads-cluster-controller" deleted  
deployment.apps "astrads-license-controller" deleted  
deployment.apps "astrads-operator" deleted  
Removed deployments  
Removing all other AstraDS resources  
namespace "astrads-system" deleted  
customresourcedefinition.apiextensions.k8s.io  
"astradsautosupports.astrads.netapp.io" deleted  
customresourcedefinition.apiextensions.k8s.io  
"astradscloudsnapshots.astrads.netapp.io" deleted  
customresourcedefinition.apiextensions.k8s.io
```

```
"astradsclusters.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsexportpolicies.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsfaileddrives.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradslicenses.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsnfsoptions.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsnodeinfoes.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsnodemanagements.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsqospolicies.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsversions.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsvolumefiles.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsvolumes.astrads.netapp.io" deleted
customresourcedefinition.apiextensions.k8s.io
"astradsvolumesnapshots.astrads.netapp.io" deleted
role.rbac.authorization.k8s.io "astrads-astrads-system-admin-role"
deleted
role.rbac.authorization.k8s.io "astrads-astrads-system-reader-role"
deleted
role.rbac.authorization.k8s.io "astrads-astrads-system-writer-role"
deleted
role.rbac.authorization.k8s.io "astrads-leader-election-role" deleted
role.rbac.authorization.k8s.io "astrads-manager-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astrads-admin-
clusterrole" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astrads-reader-
clusterrole" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astrads-writer-
clusterrole" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsautosupport-
editor-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsautosupport-
viewer-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradscloudsnapshot-
editor-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradscloudsnapshot-
viewer-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradscluster-editor-
```

```
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradscluster-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsexportpolicy-
editor-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsexportpolicy-
viewer-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsfaileddrive-
editor-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsfaileddrive-
viewer-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradslicense-editor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradslicense-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsnfsoption-editor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsnfsoption-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsnodeinfo-editor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsnodeinfo-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsnodemanagement-
editor-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsnodemanagement-
viewer-role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsqospolicy-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsversion-editor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsversion-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsvolume-editor-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-astradsvolume-viewer-
role" deleted
clusterrole.rbac.authorization.k8s.io "astrads-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                    |
+-----+

```

## Astrads-system namespace의 제거를 검증합니다

다음 명령을 실행하면 결과가 반환되지 않는지 확인합니다.

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

## 컨테이너를 작업자 노드에서 실행하지 않도록 합니다

소방관이나 netwd 같은 컨테이너가 작업자 노드에서 실행되고 있지 않는지 확인합니다. 각 노드에서 다음을 실행합니다.

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

## OpenShift Container Platform 리소스를 삭제합니다

Red Hat OpenShift Container Platform(OCP)에 Astra Data Store 미리보기를 설치한 경우 OCP SCC(Security Context Constraints) 및 rolebindings 리소스를 제거할 수 있습니다.

OpenShift는 POD가 수행할 수 있는 작업을 제어하는 SCC(Security Context Constraints)를 사용합니다.

표준 제거 프로세스를 완료한 후 다음 단계를 완료합니다.

### 1. SCC 리소스 제거:

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

### 2. rolebindings 리소스 제거:

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



이 단계에서는 "리소스를 찾을 수 없음" 오류를 무시합니다.

### 3. 모든 Kubernetes 노드에서 `/var/lib/kubelet/config.yaml`을 제거합니다.

## Astra Data Store 미리보기 제거 프로세스 문제를 해결합니다

Kubernetes v1.20의 Astra Data Store 미리 보기 제거 프로세스를 사용하면 Pod가 종료 상태로 유지되는 경우가 있습니다.

이 문제가 발생하면 다음 명령을 실행하여 "astrads-system" 네임스페이스의 모든 Pod를 강제로 삭제합니다.

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

## 스크립트 없이 Astra Data Store 미리 보기를 제거합니다

자동 스크립트 없이 Astra Data Store 미리 보기를 수동으로 제거하려면 워크로드, 바인딩, 볼륨, 익스포트 정책, 클러스터, 라이선스, 배포 환경 및 Astra Data Store 미리 보기 네임스페이스.

또는 을(를) 사용할 수 있습니다 ["스크립트를 사용하여 Astra Data Store 미리 보기를 제거합니다"](#).

무엇을 '필요로 할거야

- 루트 관리 권한

Astra Data Store 미리 보기 제거 프로세스는 다음과 같은 고급 단계를 안내합니다.

- [\[Remove existing workloads and bindings\]](#)
- [\[Uninstall the Astra Data Store preview cluster and control plane\]](#)
- [\[Delete the license\]](#)

- [\[Delete the Astra Data Store preview installation\]](#)
- [\[Validate the removal of the astrads-system namespace\]](#)
- [\[Ensure containers are not running on worker nodes\]](#)
- [\[Delete OpenShift Container Platform resources\]](#)
- [\[Troubleshoot the Astra Data Store preview uninstall process\]](#)

## 기존 워크로드 및 바인딩을 제거합니다

Astra Data Store 미리 보기를 제거하기 전에 먼저 다음을 제거해야 합니다

- Astra Data Store 미리 보기를 스토리지 백엔드로 사용하는 모든 애플리케이션 워크로드
- Astra Data Store 미리 보기를 백엔드로 사용하는 Trident 바인딩

이렇게 하면 Kubernetes 환경이 클린 상태로 유지됩니다. 이는 다시 설치할 때 중요합니다.

## Astra Data Store 미리 보기 클러스터와 컨트롤 플레인을 제거합니다

Astra Data Store 미리 보기를 수동으로 제거하려면 아래 단계를 따르십시오.

볼륨 및 익스포트 정책을 삭제합니다

클러스터를 삭제하기 전에 Astra Data Store 미리 보기 볼륨 및 익스포트 정책을 삭제해야 합니다.



먼저 볼륨을 삭제하고 정책을 내보내지 않으면 Astra Data Store 미리 보기 볼륨 개체가 삭제될 때까지 클러스터 삭제 프로세스가 일시 중지됩니다. 클러스터 삭제를 시작하기 전에 이러한 항목을 제거하는 것이 더 효율적입니다.

단계

### 1. 볼륨 삭제:

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

### 2. 익스포트 정책 삭제:

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

## Astra Data Store 미리보기 클러스터를 삭제합니다

클러스터를 삭제하면 클러스터 범위 리소스와 함께 Astra Data Store 미리보기 클러스터 객체 사용자 지정 리소스 (CR)만 삭제됩니다.



클러스터가 삭제된 후에도 운영자, 노드 정보 포드 및 클러스터 컨트롤러(Kubernetes 범위 리소스)는 그대로 유지됩니다.

클러스터를 삭제하면 기본 운영 체제가 노드에서 제거되므로 'firtap' 및 'netwd' 서비스가 중지됩니다.

제거 프로그램을 완료하는 데 약 1분 정도 걸립니다. 그런 다음 Astra Data Store 미리 보기 클러스터 범위 리소스 제거가 시작됩니다.

#### 1. 클러스터를 삭제합니다.

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

### 라이센스를 삭제합니다

1. 클러스터의 각 작업자 노드에 SSH를 통해 'firewTap' 또는 'netwd'가 작업자 노드에서 실행되고 있지 않음을 확인합니다.
2. Astra Data Store Preview 라이선스를 삭제합니다.

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

### Astra Data Store 미리 보기 설치를 삭제합니다

클러스터에서 컨트롤러, 운영자, 네임스페이스 및 지원 포드를 삭제하십시오.

#### 1. Astra Data Store 미리 보기 설치 객체를 삭제합니다.

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

#### 2. 데이터 저장소 DemonSets 및 모든 Astra Data Store 미리 보기 컨트롤러 리소스를 삭제합니다.

```
~% 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. 나머지 아티팩트 및 작업자 YAML 파일 삭제:



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

## Astads-system namespace의 제거를 검증합니다

다음 명령을 실행하면 결과가 반환되지 않는지 확인합니다.

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

## 컨테이너를 작업자 노드에서 실행하지 않도록 합니다

소방관이나 netwd 같은 컨테이너가 작업자 노드에서 실행되고 있지 않는지 확인합니다. 각 노드에서 다음을 실행합니다.

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

## OpenShift Container Platform 리소스를 삭제합니다

Red Hat OpenShift Container Platform(OCP)에 Astra Data Store 미리보기를 설치한 경우 OCP SCC(Security Context Constraints) 및 rolebindings 리소스를 제거할 수 있습니다.

OpenShift는 POD가 수행할 수 있는 작업을 제어하는 SCC(Security Context Constraints)를 사용합니다.

표준 제거 프로세스를 완료한 후 다음 단계를 완료합니다.

### 1. SCC 리소스 제거:

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

### 2. rolebindings 리소스 제거:

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



이 단계에서는 "리소스를 찾을 수 없는 오류"를 무시합니다.

### 3. 모든 Kubernetes 노드에서 `/var/lib/kubelet/config.yaml`을 제거합니다.

## 수동 삭제 샘플

다음은 실행 수동 제거 스크립트의 예입니다.

```

$ 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

```

## Astra Data Store 미리보기 제거 프로세스 문제를 해결합니다

Kubernetes v1.20의 Astra Data Store 미리 보기 제거 프로세스를 사용하면 Pod가 종료 상태로 유지되는 경우가 있습니다.

이 문제가 발생하면 다음 명령을 실행하여 "astrads-system" 네임스페이스의 모든 Pod를 강제로 삭제합니다.

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

## 저작권 정보

Copyright © 2022 NetApp, Inc. All rights reserved. 미국에서 인쇄된 본 문서의 어떤 부분도 저작권 소유자의 사전 서면 승인 없이는 어떠한 형식이나 그래픽, 전자적 또는 기계적 수단(사진 복사, 레코딩 등)으로도 저작권 소유자의 사전 서면 승인 없이 전자 검색 시스템에 저장 또는 저장.

NetApp이 저작권을 가진 자료에 있는 소프트웨어에는 아래의 라이선스와 고지 사항이 적용됩니다.

본 소프트웨어는 NetApp에 의해 "있는 그대로" 제공되며 상품성 및 특정 목적에 대한 적합성에 대한 명시적 또는 묵시적 보증을 포함하여 이에 제한되지 않고, 어떠한 보증도 하지 않습니다. NetApp은 대체품 또는 대체 서비스의 조달, 사용 불능, 데이터 손실, 이익 손실, 또는 파생적 손해(소계 물품 또는 서비스의 조달, 사용 손실, 데이터 또는 수익 손실, 계약, 엄격한 책임 또는 불법 행위(과실 또는 그렇지 않은 경우)에 관계없이 어떠한 책임도 지지 않으며, 이는 이러한 손해의 가능성을 사전에 알고 있던 경우에도 마찬가지입니다.

NetApp은 본 문서에 설명된 제품을 언제든지 예고 없이 변경할 권리를 보유합니다. NetApp은 NetApp의 명시적인 서면 동의를 받은 경우를 제외하고 본 문서에 설명된 제품을 사용하여 발생하는 어떠한 문제에도 책임을 지지 않습니다. 본 제품의 사용 또는 구입의 경우 NetApp에서는 어떠한 특허권, 상표권 또는 기타 지적 재산권이 적용되는 라이선스도 제공하지 않습니다.

본 설명서에 설명된 제품은 하나 이상의 미국 특허 또는 해외 특허, 해외 특허, 해외 특허, 해외 특허, 해외 특허, 해외 특허, 해외 특허, 해외 특허, 미국 출원 중인 특허로 보호됩니다.

권리 제한 표시: 정부에 의한 사용, 복제 또는 공개에는 DFARS 252.277-7103(1988년 10월) 및 FAR 52-227-19(1987년 6월)의 기술 데이터 및 컴퓨터 소프트웨어의 권리(Rights in Technical Data and Computer Software) 조항의 하위 조항 (c)(1)(ii)에 설명된 제한사항이 적용됩니다.

## 상표 정보

NETAPP, NETAPP 로고 및 에 나열된 마크는 NetApp에 있습니다 <http://www.netapp.com/TM> 는 NetApp, Inc.의 상표입니다. 기타 회사 및 제품 이름은 해당 소유자의 상표일 수 있습니다.