



開始使用

Astra Data Store

NetApp
June 15, 2022

This PDF was generated from <https://docs.netapp.com/zh-tw/astra-data-store/get-started/requirements.html> on June 15, 2022. Always check docs.netapp.com for the latest.

目錄

開始使用	1
Astra資料儲存區需求	1
Astra Data Store快速入門	5
安裝Astra Data Store	6
設定Astra Data Store元件	24
Astra Data Store Early Access Program (AP) 版本限制	35
Astra Data Store的常見問題集	36

開始使用

Astra資料儲存區需求

開始驗證您的環境是否符合Astra Data Store要求。

Astra Data Store支援裸機和VM型部署。Astra Data Store叢集可在具有四個以上工作者節點的Kubernetes叢集上執行。Astra Data Store軟體可與在相同Kubernetes叢集上執行的其他應用程式共存。

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



如果您打算從Astra Control Center管理Astra資料儲存區叢集、請確定您的Astra資料儲存區叢集符合 "[將由Astra Control Center管理的叢集需求](#)" 除了此處概述的要求之外、

Kubernetes工作節點資源

以下是在Kubernetes叢集中的每個工作節點上、指派給Astra Data Store軟體所需的資源：

資源	最低	最大值
Astra Data Store叢集中的節點數	4.	16
資料磁碟機數量	<ul style="list-style-type: none">• 3（存在獨立的快取裝置）• 4（如果沒有快取裝置）	14
資料磁碟機大小	100GiB	4TiB
選用快取裝置的數量	1（8GiB尺寸）	不適用

Astra Data Store為每個Kubernetes工作節點支援下列vCPU和RAM組合：

- 9個vCPU、RAM為38GiB
- 23個vCPU、搭配94 GiB的RAM



若要獲得最佳寫入效能、您應該設定專屬的高耐用度、低延遲、低容量快取裝置。

每個工作節點都有下列額外需求：

- 用於儲存Astra Data Store記錄檔的主機磁碟（開機）可用空間為100GiB或更大。
- 至少一個10GbE或更快的網路介面、適用於叢集、資料和管理流量。此外、也可以使用額外的1GbE或更快介面來分隔管理流量。

硬體與軟體

Astra Data Store軟體已在下列硬體平台、軟體和儲存組態上通過驗證。請造訪 ["NetApp社群支援"](#) 如果Kubernetes叢集組態不同、

硬體平台

- Dell R640
- Dell R740
- HPE 360
- HPE DL380
- Lenovo SR630
- Lenovo SR650



VM型部署也可以使用列示為相容的伺服器 ["VMware相容性指南"](#)。

儲存設備

Astra Data Store已通過SATA、SAS及NVMe TLC SSD的驗證。

對於VM型部署、您可以使用呈現為虛擬磁碟或直通磁碟機的磁碟機。



- 如果您的主機在硬體RAID控制器後方使用SSD、請將硬體RAID控制器設定為使用「PassthThrough」模式。
- 每個磁碟機都應該有唯一的序號。在虛擬機器建立虛擬機器期間、將屬性「disk.enableuid=true」新增至虛擬機器的進階設定。

軟體

- Hypervisor：Astra Data Store已通過VMware VM部署與vSphere 7.0的驗證。Astra Data Store不支援KVM型部署。
- Astra Data Store已在下列主機作業系統上通過驗證：
 - Red Hat Enterprise Linux 8.4
 - Red Hat Enterprise Linux 8.2
 - Red Hat Enterprise Linux 7.9
 - Red Hat Enterprise Linux CoreOS (RMCOS)
 - CentOS 8.
 - Ubuntu 20.04
- Astra Data Store已通過下列Kubernetes發佈版本的驗證：
 - Red Hat OpenShift 4.6到4.9
 - Google Anthos 1.8到1.10
 - Kubernetes 1.20到1.23
 - Rancher RKE 1.3.3



Astra Data Store需要Astra Trident來進行儲存資源配置和協調。支援Astra Trident版本210.1至22.04。請參閱 "[Astra Trident安裝說明](#)"。

網路

Astra Data Store每個叢集需要一個IP位址、才能用於MVIP。它必須是未使用或未設定的IP位址、與MIP位於同一子網路中。Astra Data Store管理介面應與Kubernetes節點的管理介面相同。

此外、每個節點也可依照下表所述進行設定：



下表使用下列縮寫：MIP：管理IP位址CIP：叢集IP位址MVIP：管理虛擬IP位址

組態	所需的IP位址
每個節點一個網路介面	<ul style="list-style-type: none">• 每個節點兩（2）個：<ul style="list-style-type: none">◦ MI/CIP：每個節點的管理介面上有一（1）個預先設定的IP位址◦ 資料IP：每個節點的一（1）個未使用或未設定的IP位址、與MIP位於同一子網路中
每個節點有兩個網路介面	<ul style="list-style-type: none">• 每個節點三個：<ul style="list-style-type: none">◦ MIP：每個節點的管理介面上有一（1）個預先設定的IP位址◦ CIP：每個節點的資料介面上有一（1）個預先設定的IP位址、與MIP位於不同的子網路中◦ 資料IP：在CIP所在的同一子網路中、每個節點有一（1）個未使用或未設定的IP位址



這些組態不使用VLAN標記。

防火牆考量

在強制執行網路防火牆流量篩選的環境中、防火牆必須設定為允許傳入流量進入下表所列的連接埠和傳輸協定。「IP位址」欄使用下列縮寫：

- MIP：每個節點的管理介面上的主要IP位址
- CIP：每個節點叢集介面上的主要IP位址
- Dip：在節點上設定一或多個資料IP位址
- MVIP：在一個叢集節點上設定的管理虛擬IP位址

連接埠/傳輸協定	IP 位址	目的	附註
111/TCP	突降	NFS	資料IP會在執行時間在叢集節點之間移動。每個節點都應該允許所有資料IP（或整個子網路）的此流量。
442/TCP	MIP	API	
635/TCP	突降	NFS	資料IP會在執行時間在叢集節點之間移動。每個節點都應該允許所有資料IP（或整個子網路）的此流量。
2010/udp	CIP	叢集/節點探索	包括傳入和傳出udp連接埠2010的單點傳播和廣播流量、包括回覆。
2049/ TCP	突降	NFS	資料IP會在執行時間在叢集節點之間移動。每個節點都應該允許所有資料IP（或整個子網路）的此流量。
2181-2183/TCP	CIP	分散式通訊	
2443/TCP	MIP	API	
2443/TCP	MVIP	API	MVIP位址可由任何叢集節點代管、並在需要時於執行時間重新定位。
4000至4006/TCP	CIP	叢集內RPC	
4045-4046/TCP	突降	NFS	資料IP會在執行時間在叢集節點之間移動。每個節點都應該允許所有資料IP（或整個子網路）的此流量。
7700/TCP	CIP	工作階段管理程式	
9919/TCP	MIP	DMS API	
9920/TCP	突降	DMS REST伺服器	
ICMP	CIP + DIP	節點內通訊、健全狀況檢查	資料IP會在執行時間在叢集節點之間移動。每個節點都應該允許所有資料IP（或整個子網路）的此流量。

Astra Trident

Astra Data Store要求應用程式Kubernetes叢集執行Astra Trident以進行儲存資源配置和協調。支援Astra Trident 版本210.1至22.04。Astra資料儲存區可設定為 **"儲存後端"** 使用Astra Trident來配置持續磁碟區。

Container網路介面

Astra Data Store已通過下列Container Network Interfaces（Container Network Interfaces、簡稱CNI）的驗證。

- 適用於RKE叢集的Calico
- Calico和Weave Net CNI適用於香草Kubernetes叢集
- 適用於Red Hat OpenShift Container Platform（OCP）的OpenShift SDN
- Google Anthos的Cilium



- 隨Cilium CNI部署的Astra Data Store需要Portmap外掛程式、才能支援hostPort。您可以將「CNI鏈結模式：portmap」新增至cilium組態地圖、然後重新啟動Cilium Pod、以啟用CNI鏈結模式。
- 只有Calico和OpenShift SDN CNI才支援啟用防火牆的組態。

授權

Astra Data Store需要有效授權才能啟用完整功能。

["請在此註冊"](#) 以取得Astra Data Store授權。下載授權的指示將會在您註冊後寄送給您。

下一步

檢視 ["快速入門"](#) 總覽：

以取得更多資訊

["Astra資料儲存區的限制"](#)

Astra Data Store快速入門

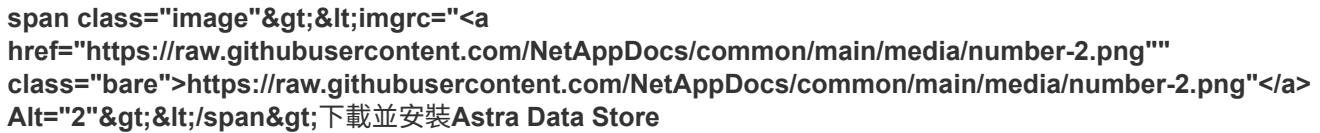
本頁提供Astra Data Store入門所需步驟的高階概觀。每個步驟中的連結都會帶您前往提供更多詳細資料的頁面。

歡迎試用！如果您想要試用Astra Data Store、可以使用90天試用版授權。 ["請在此註冊"](#) 取得Astra Data Store授權。

 <https://raw.githubusercontent.com/NetAppDocs/common/main/media/number-1.png>
<https://raw.githubusercontent.com/NetAppDocs/common/main/media/number-1.png>
alt="1"> 檢閱Kubernetes叢集需求

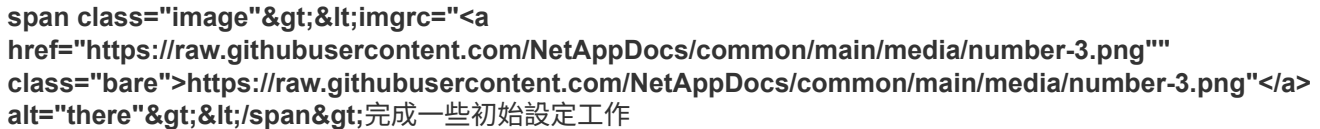
- 叢集必須以正常狀態執行、且至少有四個或更多個工作節點。
- Astra Data Store部署中的每個Kubernetes工作節點、都應該有相同介面類型（SATA、SAS或NVMe）的SSD、以及指派給Astra Data Store叢集的相同磁碟機數量。
- 每個SSD都應有一個唯一的序號。

深入瞭解 ["Astra資料儲存區需求"](#)。

 <https://raw.githubusercontent.com/NetAppDocs/common/main/media/number-2.png>
Alt="2">下載並安裝Astra Data Store

- 從下載Astra Data Store ["NetApp 支援網站"](#)。
- 在您的本機環境中安裝Astra Data Store。
- 套用Astra Data Store授權。
- 安裝Astra Data Store叢集。

深入瞭解 ["安裝Astra Data Store"](#)。

 <https://raw.githubusercontent.com/NetAppDocs/common/main/media/number-3.png>
alt="there">完成一些初始設定工作

- 安裝Astra Trident。
- 安裝Kubernetes快照自訂資源定義（CRD）和控制器。
- 將Astra Data Store設定為儲存後端。
- 建立預設的Astra Data Store儲存類別。
- 設定Astra資料儲存區以使用遙測服務。

深入瞭解 ["初始設定程序"](#)。

完成Astra Data Store設定之後、接下來您可以：

- 使用kubectl命令和kubectl astrads擴充功能來管理叢集、包括將節點置於維護模式、更換磁碟機或更換節點等工作。深入瞭解 ["如何搭配Astra Data Store使用kubectl命令"](#)。
- 設定監控端點。深入瞭解 ["設定監控端點"](#)。
- ["使用Astra Data Store搭配VMware"](#)。

["安裝Astra Data Store"](#)。

安裝Astra Data Store

您可以使用Kubernetes原生命令或使用Astra Control Center中的UI來安裝Astra Data Store。

安裝選項

- 使用**Kubernetes**原生命令：若要使用Kubernetes原生命令安裝Astra Data Store、請完成中所述的安裝步驟[此程序](#)。
- 使用**Astra Control Center**：若要使用Astra Control Center安裝Astra Data Store、請完成中所述的安裝步驟[此程序](#)。

您的需求

- ["開始安裝之前、請先準備好環境以進行Astra Data Store部署"](#)。

- 存取 ["NetApp 支援網站"](#)。"註冊" 如果您尚未擁有完整存取權限的NetApp支援網站帳戶、請下載試用版。
- 答 ["NetApp授權檔案 \(NLF\)"](#) 適用於Astra資料儲存區。下載授權的指示將會在您完成後寄送給您 ["註冊"](#)。
- 具有作用中內容叢集管理權限的作用中Kbeconfig。
- 瞭解 ["角色與權限"](#) 由Astra Data Store使用。

安裝Astra Data Store

標準Kubernetes叢集的Astra Data Store安裝程序會引導您完成下列高層級步驟。此外、也會說明Red Hat OpenShift Container Platform (OCP) 環境的其他安裝步驟。

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



若要搭配Google Anthos使用Astra Data Store、請完成這些安裝步驟、並將Astra Data Store叢集新增至您的Anthos環境。



若要在Rancher RKE環境中安裝Astra Data Store、請完成下列安裝步驟、並以Rancher命令取代KECBEctl命令。

下載Astra Data Store產品組合並擷取映像

1. 登入 ["NetApp 支援網站"](#) 並下載Astra Data Store套裝組合（「Astra_Data_Store_2022.05.tar」）。



如果您要尋找舊版套裝組合的說明、請參閱 ["該版本的文件"](#)。

2. （可選）使用以下命令驗證套件的簽名：

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

3. 建立目錄：

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

4. 擷取影像：

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



影像將擷取至工作目錄中建立的「astrads/images/」目錄。

複製二進位檔並將映像推送至本機登錄

1. 從您用來擷取映像的目錄、將Kkbecl-astra二進位檔複製到安裝k8s kkebecl二進位檔的標準路徑（以下範例中使用「usr/bin/」）。Kustbecl-astrads是自訂的Kvecl擴充功能、可安裝及管理Astra Data Store叢集。



使用「which kubectll」命令尋找安裝kubectll二進位檔的路徑。

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

2. 將Astra Data Store映像目錄中的檔案新增至本機登錄。



請參閱以下自動載入影像的範例指令碼。

- a. 登入您的登錄：

```
docker login [your_registry_path]
```

- b. 將環境變數設為您要推送Astra Data Store映像的登錄路徑、例如「REpo.company.com」。

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

- c. 執行指令碼、將影像載入Docker、標記影像、然後[Subforte_image_local_register_pip]將影像推送到本機登錄：

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

OpenShift程序

下列程序僅適用於Red Hat OpenShift Container Platform（OCP）上的部署。在非OCP Kubernetes叢集上部署時、請跳過此程序。

- 建立命名空間以部署Astra Data Store
- 建立自訂的SCC
- 建立角色與角色繫結

範例 1. 詳細資料

建立名稱空間「astrads系統」、以便安裝所有Astra Data Store元件。

只有在Red Hat OpenShift Container Platform (OCP) 上部署時才需要執行下列步驟。

1. 建立命名空間：

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

範例：ads_names.yaml

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

OpenShift使用安全內容限制 (SCC) 來控制Pod可以執行的動作。根據預設、任何容器的執行都會被授予受限的SCC、而且只會授予該SCC所定義的功能。

受限的SCC不提供Astra Data Store叢集Pod所需的權限。使用此程序可提供Astra Data Store所需的權限（列於範例中）。

將自訂SCC指派給Astra Data Store命名空間的預設服務帳戶。

只有在Red Hat OpenShift Container Platform (OCP) 上部署時、才需要執行下列步驟。

1. 建立自訂SCC：

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

範例：ads_特權_scc.yaml

```

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

```

2. 使用「occ Get SCC」命令顯示新增的SCC：

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

建立Astra Data Store預設服務帳戶所需的角色和角色繫結。

下列yaml定義會指派「astrads.netapp.io` API」群組中Astra Data Store資源所需的各種角色（透過角色繫結）。

只有在Red Hat OpenShift Container Platform（OCP）上部署時、才需要執行下列步驟。

1. 建立定義的角色和角色繫結：

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

範例：oc_role_bindings.yaml

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

```

- kind: ServiceAccount
  name: default
  namespace: astrads-system
---

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

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

```

設定私有映像登錄

對於某些環境而言、您可以將組態變更為從使用機密的私有登錄中擷取映像、做為選用步驟。

1. 除非您已在上一步中建立「astrads系統」命名空間：

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

2. 建立秘密：

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

3. 將機密組態資訊新增至服務帳戶：

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



這些變更將會在您執行時套用 [安裝Astra Data Store營運者](#)。

安裝Astra Data Store營運者

1. 列出Astra資料儲存區清單：

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

回應：

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

2. 使用「kubectl apply」部署營運者：

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

回應：



命名空間回應可能會因執行標準安裝或而有所不同 ["OpenShift Container Platform安裝"](#)。

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



```
customresourcedefinition.apiextensions.k8s.io/astradscLOUDsnapshots.astr  
ads.netapp.io created  
customresourcedefinition.apiextensions.k8s.io/astradsclusters.astrads.ne  
tapp.io created  
customresourcedefinition.apiextensions.k8s.io/astradsexportpolicies.astr  
ads.netapp.io created  
customresourcedefinition.apiextensions.k8s.io/astradsfaileddrives.astrad  
s.netapp.io created  
customresourcedefinition.apiextensions.k8s.io/astradskeyproviders.astrad  
s.netapp.io created  
customresourcedefinition.apiextensions.k8s.io/astradsllicenses.astrads.ne  
tapp.io created  
customresourcedefinition.apiextensions.k8s.io/astradsnfsoptions.astrads.  
netapp.io created  
customresourcedefinition.apiextensions.k8s.io/astradsnodeinfoes.astrads.  
netapp.io created  
customresourcedefinition.apiextensions.k8s.io/astradsnodemagements.ast  
rads.netapp.io created  
customresourcedefinition.apiextensions.k8s.io/astradsqospolicies.astrads  
.netapp.io created  
customresourcedefinition.apiextensions.k8s.io/astradssearkeyrotatereques  
ts.astrads.netapp.io created  
customresourcedefinition.apiextensions.k8s.io/astradsversions.astrads.ne  
tapp.io created  
customresourcedefinition.apiextensions.k8s.io/astradsvolumeFiles.astrads  
.netapp.io created  
customresourcedefinition.apiextensions.k8s.io/astradsvolumes.astrads.net  
app.io created  
customresourcedefinition.apiextensions.k8s.io/astradsvolumesnapshots.ast  
rads.netapp.io created  
role.rbac.authorization.k8s.io/astrads-astrads-system-admin-role created  
role.rbac.authorization.k8s.io/astrads-astrads-system-reader-role  
created  
role.rbac.authorization.k8s.io/astrads-astrads-system-writer-role  
created  
role.rbac.authorization.k8s.io/astrads-leader-election-role created  
role.rbac.authorization.k8s.io/astrads-manager-role created  
clusterrole.rbac.authorization.k8s.io/astrads-astrads-admin-clusterrole  
created  
clusterrole.rbac.authorization.k8s.io/astrads-astrads-reader-clusterrole  
created  
clusterrole.rbac.authorization.k8s.io/astrads-astrads-writer-clusterrole  
created  
clusterrole.rbac.authorization.k8s.io/astrads-astradsautosupport-editor-  
role created  
clusterrole.rbac.authorization.k8s.io/astrads-astradsautosupport-viewer-
```

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

```

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

```

3. 確認Astra Data Store營運者Pod已啟動且正在執行：

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

回應：

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

部署Astra Data Store版本Yaml

1. 使用「kubectl apply」部署：

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

2. 確認Pod正在執行：

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

回應：

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

套用Astra Data Store授權

1. 套用您從NetApp取得的NetApp授權檔案（NLF）。執行命令之前、請輸入您所在的叢集名稱（「<Astra Data-Store-cluster名稱>」） [即將部署](#) 或已部署及授權檔案路徑（「<file_path/file.txt>」）：

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

2. 確認已新增授權：

```
kubectl astrads license list
```

回應：

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

安裝Astra Data Store叢集

1. 開啟Yaml檔案：

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

2. 在Yaml檔案中編輯下列值。



以下步驟為Yaml檔案的簡化範例。

- (必填) 中繼資料：在「metadata」中、將「name」字串變更為叢集名稱。這必須與您在使用時使用的叢集名稱相同 [套用授權](#)。
- (必填) 規格：在「show」中變更下列必要值：
 - 視授權和Astra Data Store安裝大小而定、將「adsNodeConfig」值變更為安裝所需的值：
 - 小型：9個CPU和38個記憶體
 - 中：23個CPU和94個記憶體
 - (選用) 移除「adsNodeSelector」區段的註解。如果您想要限制Astra Data Store只安裝在選取的工作節點集區、請設定此選項。
 - (選用) 指定Astra Data Store叢集應使用的特定節點數、範圍介於4-16之間。
 - 將「mVIP」字串變更為可從叢集中任何工作節點路由傳送之浮動管理IP的IP位址。
 - 在「adsDataNetworks」中、新增一個以逗號分隔的浮動IP位址清單（「Addresses」）、這些位址可從您要掛載NetApp Volume的任何主機路由傳送。每個節點使用一個浮動IP位址。資料網路IP位址應至少與Astra Data Store節點一樣多。對於Astra Data Store、這表示如果您打算稍後擴充叢集、至少需要4個位址或最多16個位址。
 - 在「adsDataNetworks」中、指定資料網路使用的網路遮罩。
 - 在「adsNetworks介面」中、將「<mgmt_interface_name>」和「<cluster與_storage介面名稱>」值取代為您要用於管理、叢集與儲存的網路介面名稱。如果未指定名稱、則節點的主要介面將用於管理、叢集和儲存網路。請務必移除「adsNetworks介面」一節的註解。



叢集和儲存網路必須位於相同的介面上。Astra Data Store管理介面應與Kubernetes節點的管理介面相同。

- (選用) 監控組態：如果您要設定 [監控營運者](#)（若您未使用Astra Control Center進行監控、則為選用）、從區段移除註解、新增套用代理程式CR（監控操作員資源）的命名空間（預設為「NetApp-Monitoring」（NetApp監控）、並新增您在先前步驟中使用的登錄（「您的登錄路徑」）的repo路徑。
- (可選) * autosupSupportConfig*：保留 "AutoSupport" 除非您需要設定Proxy、否則預設值為：

- 對於「proxyURL」、請使用連接埠來設定Proxy的URL、以便AutoSupport 進行套裝組合傳輸。



為了簡單起見、以下Yaml範例中已移除部分意見。

```
apiVersion: astrads.netapp.io/v1beta1
kind: AstraDSCluster
metadata:
  name: astrads-cluster-name
  namespace: astrads-system
spec:
  adsNodeConfig:
    cpu: 9
    memory: 38
    # [Optional] Specify node selector labels to select the nodes for
    # creating ADS cluster
    # adsNodeSelector:
    #   matchLabels:
    #     customLabelKey: customLabelValue
  adsNodeCount: 4
  mvip: ""
  adsDataNetworks:
    - addresses: ""
      netmask:
        # Specify the network interface names to use for management, cluster
        # and storage networks.
        # If none are specified, the node's primary interface will be used for
        # management, cluster and storage networking.
        # To move the cluster and storage networks to a different interface
        # than management, specify all three interfaces to use here.
        # NOTE: The cluster and storage networks need to be on the same
        # interface.
  adsNetworkInterfaces:
    managementInterface: "<mgmt_interface_name>"
    clusterInterface: "<cluster_and_storage_interface_name>"
    storageInterface: "<cluster_and_storage_interface_name>"
    # [Optional] Provide a monitoring config to be used to setup/configure
    # a monitoring agent.
  # monitoringConfig:
  #   # namespace: "netapp-monitoring"
  #   # repo: "[YOUR REGISTRY]"
  autoSupportConfig:
    autoUpload: true
    enabled: true
    coredumpUpload: false
    historyRetentionCount: 25
    destinationURL: "https://support.netapp.com/put/AsupPut"
```

```
# ProxyURL defines the URL of the proxy with port to be used for
AutoSupport bundle transfer
# proxyURL:
periodic:
  - schedule: "0 0 * * *"
    periodicconfig:
      - component:
          name: storage
          event: dailyMonitoring
          userMessage: Daily Monitoring Storage AutoSupport bundle
          nodes: all
      - component:
          name: controlplane
          event: daily
          userMessage: Daily Control Plane AutoSupport bundle
```

3. 使用「kubectl apply」部署叢集：

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

4. 等待幾分鐘、讓叢集建立作業完成、然後確認Pod正在執行：

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

回應範例：

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

5. 驗證叢集部署進度：

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

回應範例：

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

瞭解與部署相關的事件

在叢集部署期間、作業狀態應從「空白」變更為「進行中」、改為「已建立」。叢集部署將持續約8至10分鐘。若要在部署期間監控叢集事件、您可以執行下列任一命令：

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

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

以下是部署期間的重要事件：

活動	訊息與重要性
控制面板無選項	成功選取[number]個控制面板節點以加入ADS叢集。Astra Data Store營運者利用CPU、記憶體、儲存設備和網路來識別足夠的節點、以建立Astra Data Store叢集。
ADSClusterCreateInProgress	Astra Data Store叢集控制器已啟動叢集建立作業。
ADSClusterCreeSuccess	已成功建立叢集。

如果叢集的狀態並未變更為「進行中」、請查看操作員記錄、以取得節點選擇的詳細資料：

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

如果叢集的狀態卡在「In Progress（進行中）」、請檢查叢集控制器的記錄：

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

使用Astra Control Center安裝Astra資料儲存區

若要搭配Astra Control Center部署及使用Astra Data Store、請執行下列步驟。

您需要的是 **#8217** ；需要的是什麼

- 您已檢閱過 [一般Astra資料儲存區先決條件](#)。

- 您已安裝Astra Control Center。

步驟

1. ["使用Astra Control Center部署Astra資料儲存區"](#)。

下一步

- * Kubernetes原生部署與協力廠商發佈*：執行其他作業來完成Astra Data Store部署 ["設定工作"](#)。
- * Astra控制中心*：如果您已使用Astra控制中心來部署Astra資料儲存區、則不需要遵循這些步驟 ["設定工作"](#) 除非您想設定任何其他監控選項。部署Astra Data Store之後、您可以使用Astra Control Center UI來完成下列工作：
 - ["監控Astra Data Store資產的健全狀況"](#)。
 - ["管理Astra Data Store後端儲存設備"](#)。
 - ["監控節點、磁碟和持續磁碟區宣告 \(PVCS\) "](#)。

設定Astra Data Store元件

您就可以了 ["安裝獨立式Astra Data Store"](#) 並解決一些問題 ["環境先決條件"](#) 您將安裝Astra Trident、設定Kubernetes快照功能、設定儲存後端、以及建立預設儲存類別：



如果您使用Astra Control Center來部署Astra Data Store、除非您打算設定、否則不需要執行這些步驟 [其他監控選項](#)。

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

安裝Astra Trident

對於Astra Data Store、您需要安裝Astra Trident 21.10.1或更新版本。您可以使用下列其中一個選項來安裝Astra Trident：

- ["使用tridentctl安裝Astra Trident"](#)。
- ["使用Trident操作員安裝Astra Trident"](#)。



您可以手動或使用Helm來部署Trident運算子。

安裝Kubernetes Snapshot客戶需求日和控制器

需要Kubernetes快照CRD和控制器才能建立持續磁碟區宣告 (PVR) 快照。如果您的環境尚未安裝CRD和控制器、請執行下列命令來安裝。



以下命令範例假設為目錄「/trident」、但您使用的目錄可以是任何您用來下載Yaml檔案的目錄。

您需要的是 **#8217** ；需要的是什麼

- "開始安裝之前、請先準備好環境以進行Astra Data Store部署"。
- 下載 "Kubernetes快照控制器Yaml檔案"：
 - 設定快照控制器.yaml
 - RBAC快照控制器.yaml
- 下載 "Y反洗錢客戶需求日"：
 - Snapshot.storage ◦ k8s.io_volumesnapshotclasses.yaml
 - Snapshot.storage ◦ k8s.io_volumesnapshotcontents ◦ yaml
 - Snapshot.storage : k8s.io_volumesnapshots.yaml

步驟

1. 套用snapshot.storage ◦ k8s.io_volumesnapshotclasses.yaml：

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

回應：

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

2. 套用snapshot.storage ◦ k8s.io_volumesnapshotcontents ◦ yaml：

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

回應：

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

3. 套用snapshot.storage ◦ k8s.io_volumesnapshots.yaml：

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

回應：

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

4. 套用setup-snapshot控制器.yaml：

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

回應：

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

5. 套用RBAC快照控制器.yaml：

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

回應：

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

6. 確認已套用CRD Y反 洗錢檔案：

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

回應範例：

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

7. 確認已套用快照控制器檔案：

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

回應範例：

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

將Astra Data Store設定為儲存後端

在ads_backend.json檔案中設定儲存後端參數、然後建立Astra Data Store儲存後端。

步驟

1. 使用安全終端建立「ads_backend.json」：

```
vi ads_backend.json
```

2. 設定Json檔案：



Json範例遵循下列步驟。

- 將「叢集」值變更為Astra Data Store叢集的叢集名稱。
- 將「命名空間」值變更為您要用於建立磁碟區的命名空間。
- 除非您針對此後端設定匯出原則CR,否則請將「AutoExportPolicy」值變更為「true」。
- 將您要授予存取權的IP位址填入「AutoExportCIDR」清單。使用「0.0.0.0/0」允許所有人。
- 如需「kubeconfig」值、請執行下列動作：
 - 將.kube /組態Yaml檔案轉換為Json格式、並將其最小化、不含空格：

轉換範例：

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

ii. 將編碼為基64、並使用基64輸出作為「kubeconfig」值：

編碼範例：

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

```

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

```

3. 切換到您下載Trident安裝程式的目錄：

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

4. 建立儲存後端：

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

回應範例：

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

建立預設的Astra Data Store儲存類別

建立Astra Trident預設儲存類別、並將其套用至儲存後端。

步驟

1. 建立Trident - csi儲存類別：

a. 建立ads_sc_example.yaml：

```
vi ads_sc_example.yaml
```

範例：


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

b. 建立Trident - csi :

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

回應：

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

2. 確認已新增儲存類別：

```
kubectl get storageclass
```

回應：

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

3. 切換到您下載Trident安裝程式的目錄：

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

4. 確認Astra Trident後端已使用預設的儲存類別參數更新：

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

回應範例：

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

```

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

```

```

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

```

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

設定Astra Data Store監控

（選用）您可以設定Astra Data Store、以供其他遙測服務進行監控。如果您未使用Astra Control Center進行Astra Data Store監控、或想要將監控延伸至其他端點、建議您執行此程序。

如果您的Astra Data Store執行個體是獨立部署、使用Cloud Insights 支援程序來監控遙測、或是將記錄串流至第三方端點（例如Elastic）、您可以安裝監控操作員。



針對Astra Control Center部署、系統會自動設定監控操作員。您可以跳過下列程序的前兩個命令。

在設定監控之前、您需要在「astrads系統」命名空間中使用作用中的Astra資料儲存叢集。

步驟

1. 執行此安裝命令：

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

2. 設定Astra資料儲存區以進行監控：

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

3. 設定Astra Data Store將EMS記錄串流至彈性端點：

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

Astra Data Store Early Access Program (AP) 版本限制

Astra Data Store在早期存取方案期間有下列資源限制。

資源	最低	最大值
Astra Data Store叢集中的節點數	4.	16
每個節點的持續磁碟區數目	不適用	50
Volume大小	20億	2TiB
每個Volume的快照	0	1023

資源	最低	最大值
每個Volume的複本	0	9.
每個節點的VM數	0	50

Astra Data Store的常見問題集

尋找有關Astra Data Store Early Access Program版本安裝、設定、升級及疑難排解的常見問題解答。

一般問題

*我可以使用Astra Data Store Early Access Program正式作業嗎？*不可以雖然Astra Data Store是專為提供企業級恢復能力而設計與開發、但Astra Data Store的早期存取方案版本並不適用於正式作業工作負載。

*我可以將Astra Data Store用於虛擬機器工作負載嗎？*是。Astra Data Store同時支援Kubernetes和VMware vVol工作負載。

請參閱 ["瞭解Astra Data Store與VMware的相關資訊"](#)。

*我可以使用VMware vSphere來管理Astra Data Store嗎？*是的、Astra Data Store可以在vCenter內使用適用於VMware vSphere的NetApp Astra外掛程式進行原生管理。請參閱 ["管理VMware安裝的Astra Data Store元件"](#)。

- Astra Data Store是否因其運作而與其他NetApp產品有任何相依關係？*

是的。Astra Data Store需要NetApp SCSI驅動程式Astra Trident版本210.1及更新版本、才能部署在工作負載Kubernetes叢集上。深入瞭解 ["Astra資料儲存區需求"](#)。

Astra Control Center是啟用VMware工作流程和適用於VMware vSphere的NetApp Astra外掛程式的必要條件。

將Astra Data Store叢集當作儲存後端的應用程式可以使用 ["Astra控制中心"](#) 運用應用程式感知資料管理功能、包括資料保護、災難恢復、以及Kubernetes工作負載移轉。

*我該如何管理Astra Data Store叢集？*您可以使用kubectl命令及Kubernetes API副檔名來管理Astra Data Store資源。

「kubectl astrads」命令包含一個「-h」交換器、可提供使用方法和標記文件、方便您使用。

*我該如何監控Astra Data Store叢集指標？*您可以使用Cloud Insights 支援功能來監控Astra Data Store指標（請參閱 ["利用功能表監控指標Cloud Insights"](#)）或Kubernetes原生監控（請參閱 ["使用Prometheus和Grafana進行監控"](#)）。

您也可以監控記錄。請參閱 ["設定及監控事件記錄"](#)。

*我可以在ONTAP Kubernetes叢集中、搭配使用Astra Data Store和其他的儲存供應商嗎？*可以。Astra Data Store可與應用程式叢集中的其他儲存供應商一起使用。

*如果我從Astra Data Store移除Kubernetes叢集、是否會解除安裝Astra Trident？*如果您解除安裝Astra Data Store、則不會從叢集解除安裝Astra Trident。如果您需要解除安裝Astra Trident、則必須個別執行。

授權

- Astra Data Store是否需要授權？*是的、Astra Data Store需要評估版NetApp授權檔案（NLF）才能執行早期存取方案。

請參閱 ["Astra資料儲存區需求"](#)。

- Astra Data Store試用版授權的有效時間有多長？* Astra Data Store授權的預設期限為下載日期起算的90天。

在Kubernetes叢集上安裝及使用Astra Data Store

*我可以在裸機或虛擬機器上執行的Kubernetes叢集上安裝Astra Data Store嗎？*可以。Astra Data Store可安裝在執行裸機或vSphere VM的Kubernetes叢集上。請參閱 ["Astra資料儲存區需求"](#)。

- Astra Data Store支援的Kubernetes版本有哪些？*

Astra Data Store可搭配與v1.20及更新版本相容的Kubernetes發佈產品。不過、目前並未針對所有Kubernetes配送進行驗證。深入瞭解 ["Astra資料儲存區需求"](#)。

*我的Kubernetes叢集大於4個工作節點。我可以在其中安裝Astra Data Store嗎？*是。Astra Data Store叢集一開始必須部署在Kubernetes叢集中的4個工作節點上。部署之後、您可以將叢集擴充至最多16個工作者節點。

- Astra Data Store是否支援從私有登錄進行離線安裝？*是。Astra Data Store可從本機登錄離線安裝。請參閱 ["安裝Astra Data Store"](#)。

*我需要網際網路連線才能使用Astra Data Store嗎？*不需要、Astra Data Store Early Access Program不需要網際網路連線。不過、建議您連線至NetApp AutoSupport 功能後端、以便定期傳送遙測套件。這種連線方式可以是直接連線、也可以透過Proxy連線。

- Astra Data Store使用什麼角色和權限？*您必須是Kube管理員、才能部署Astra Data Store營運者。

Astra Data Store有一組名為「astrads-ds-nodeinfo」的特殊權限取消程式集、可用來探索用於選取節點的節點資源。

此外、營運者將使用特殊權限Kubernetes工作、在所選的工作節點上安裝儲存叢集的容器、以建置Astra Data Store儲存叢集。

*我需要更新哪些資訊清單檔案才能安裝Astra Data Store？*請從下載的Astra Data Store套裝組合下載 ["NetApp支援網站"](#)、您將獲得下列資訊清單：

- astradscluster . yaml
- astradsoper.yaml
- astradsversion.yaml
- 監控_opuler.yaml

您需要使用部署專屬的組態來更新「astradscluster . yaml」資訊清單。請參閱 ["安裝Astra Data Store"](#)。

疑難排解與支援

有了Astra Data Store、您可以使用NetApp Container Slack通路來存取社群支援。此通路由NetApp支援與我們

的技術行銷工程師監控。

["NetApp Container Slack通路"](#)

請參閱 ["Astra Data Store支援營運"](#)。

*我要如何提出支援案例、或是詢問有關快速問題的說明？*提出支援案例、或是取得快速問題的說明、並在上回報您的問題或問題 ["NetApp Container Slack通路"](#)。NetApp支援部門將與您密切合作、以盡力提供協助。

*如何提出新功能的申請？*如果您對支援的組態或功能有任何疑問、[請聯絡astra.feedback@netapp.com](mailto:astra.feedback@netapp.com)。

*如何產生支援記錄套裝組合？*請參閱 ["產生支援服務組合"](#) 以取得有關為Astra Data Store設定及下載支援記錄套裝組合的說明。

- Astra Data Store找不到Kubernetes節點。如何修正此問題？*請參閱 ["安裝Astra Data Store"](#)。
- IPv6位址是否可用於管理、資料及叢集網路？*否、Astra Data Store僅支援IPV4位址。未來Astra Data Store版本將新增IPv6支援。

在**Astra Data Store**上配置磁碟區時、會使用什麼**NFS**版本？Astra Data Store支援NFS v4.1、以供Kubernetes應用程式配置的所有磁碟區使用、以及為VMware工作負載配置的所有磁碟區使用NFSv3。

請參閱 ["Astra資料儲存區需求"](#) 和 ["Astra資料儲存區的限制"](#)。

升級**Astra Data Store**

*我可以從Astra Data Store預覽版本升級嗎？*是。您可以從Astra Data Store Early Access Program版本升級至未來版本。

版權資訊

Copyright©2022 NetApp、Inc.版權所有。美國印製本文件中版權所涵蓋的任何部分、不得以任何形式或任何方式（包括影印、錄製、在未事先取得版權擁有者書面許可的情況下、在電子擷取系統中進行錄音或儲存。

衍生自受版權保護之NetApp資料的軟體必須遵守下列授權與免責聲明：

本軟體係由NetApp「依現狀」提供、不含任何明示或暗示的保證、包括但不限於適售性及特定用途適用性的暗示保證、特此聲明。在任何情況下、NetApp均不對任何直接、間接、偶發、特殊、示範、或衍生性損害（包括但不限於採購替代商品或服務；使用損失、資料或利潤損失；或業務中斷）、無論是在合約、嚴格責任或侵權行為（包括疏忽或其他）中、無論是因使用本軟體而產生的任何責任理論（包括疏忽或其他）、即使已被告知可能造成此類損害。

NetApp保留隨時變更本文所述之任何產品的權利、恕不另行通知。除非NetApp以書面明確同意、否則NetApp不承擔因使用本文所述產品而產生的任何責任或責任。使用或購買本產品並不代表NetApp擁有任何專利權利、商標權利或任何其他智慧財產權。

本手冊所述產品可能受到一或多個美國國家/地區的保護專利、國外專利或申請中。

限制權利圖例：政府使用、複製或揭露受DFARS 252.277-7103（1988年10月）和FAR 52-227-19（1987年6月）技術資料與電腦軟體權利條款（c）（1）（ii）分段所述限制。

商標資訊

NetApp、NetApp標誌及所列的標章 <http://www.netapp.com/TM> 為NetApp、Inc.的商標。其他公司和產品名稱可能為其各自所有者的商標。