



使用 Trident

Trident

NetApp
July 01, 2026

目錄

使用 Trident	1
準備工作節點	1
選擇合適的工具	1
節點服務探索	1
NFS 磁碟區	2
iSCSI 磁碟區	2
NVMe/TCP Volume	6
SCSI over FC 磁碟區	7
準備配置 SMB Volume	9
設定和管理後端	10
配置後端	10
Azure NetApp Files	10
Google Cloud NetApp Volumes	29
設定 NetApp HCI 或 SolidFire 後端	56
ONTAP SAN 驅動程式	61
ONTAP NAS 驅動程式	88
Amazon FSx for NetApp ONTAP	123
使用 kubectl 建立後端	157
管理後端	163
建立和管理儲存類別	173
建立儲存類別	173
管理儲存類別	176
配置和管理磁碟區	178
配置磁碟區	178
擴充磁碟區	182
了解 RWX NVMe 子系統限制	193
控制器擴充性	194
自動磁碟區擴充	199
管理 Autogrow 策略	205
匯入磁碟區	214
自訂磁碟區名稱和標籤	226
跨命名空間共享 NFS Volume	229
跨命名空間複製磁碟區	233
使用 SnapMirror 複製磁碟區	235
使用 CSI 拓撲	241
使用快照	249
使用磁碟區群組快照	257

使用 Trident

準備工作節點

Kubernetes 叢集中的所有工作節點都必須能夠掛載您為 Pod 配置的磁碟區。若要準備工作節點，您必須根據所選驅動程式安裝 NFS、iSCSI、NVMe/TCP 或 FC 工具。

選擇合適的工具

如果您使用多種驅動程式，則應安裝所有驅動程式所需的工具。最新版本的 Red Hat Enterprise Linux CoreOS (RHCOS) 預設已安裝這些工具。

NFS 工具

"[安裝 NFS 工具](#)"如果您正在使用：`ontap-nas`、`ontap-nas-economy`、`ontap-nas-flexgroup` 或 `azure-netapp-files`。

iSCSI 工具

"[安裝 iSCSI 工具](#)" 如果您正在使用：`ontap-san`、`ontap-san-economy`、`solidfire-san`。

NVMe 工具

"[安裝 NVMe 工具](#)" 如果您使用 `ontap-san` 非揮發性記憶體高速介面 (NVMe) over TCP (NVMe/TCP) 協定。



NetApp 建議使用 ONTAP 9.12 或更新版本來使用 NVMe/TCP。

透過 FC 進行 SCSI 的工具

如需有關設定 FC 和 FC-NVMe SAN 主機的詳細資訊，請參閱 "[配置 FC 和 FC-NVMe SAN 主機的方法](#)"。

"[安裝 FC 工具](#)" 如果您使用 `ontap-san` 搭配 `sanType fcp` (透過 FC 連線的 SCSI)。

注意事項：* SCSI over FC 在 OpenShift 和 KubeVirt 環境中受到支援。* SCSI over FC 不支援 Docker。* iSCSI 自癒功能不適用於 SCSI over FC。

SMB 工具

"[準備配置 SMB Volume](#)" 如果您正在使用：`ontap-nas` 來配置 SMB Volume。

節點服務探索

Trident 會嘗試自動偵測節點是否可以執行 iSCSI 或 NFS 服務。



節點服務發現功能可以辨識已發現的服務，但並不能保證這些服務配置正確。反之，未發現服務也不代表磁碟區掛載一定會失敗。

檢閱事件

Trident 會為節點建立事件，以識別已發現的服務。若要檢視這些事件，請執行：

```
kubectl get event -A --field-selector involvedObject.name=<Kubernetes node name>
```

檢閱探索到的服務

Trident 會辨識 Trident 節點 CR 上每個節點啟用的服務。若要檢視已探索的服務、請執行：

```
tridentctl get node -o wide -n <Trident namespace>
```

NFS 磁碟區

使用適用於您作業系統的命令安裝 NFS 工具。確保 NFS 服務在開機時啟動。

RHEL 8+

```
sudo yum install -y nfs-utils
```

Ubuntu

```
sudo apt-get install -y nfs-common
```



安裝 NFS 工具後重新啟動工作節點，以防止將磁碟區附加到容器時發生故障。

iSCSI 磁碟區

Trident 可以自動建立 iSCSI 工作階段、掃描 LUN、探索多重路徑裝置、格式化裝置並將其掛載到 Pod。

iSCSI 自癒能力

對於 ONTAP 系統、Trident 每五分鐘執行一次 iSCSI 自我修復、以：

1. *識別*所需的 iSCSI 工作階段狀態和目前的 iSCSI 工作階段狀態。
2. *比較*所需狀態與目前狀態，以識別所需的修復。Trident 會決定修復優先順序以及何時優先進行修復。
3. 執行必要的修復，使目前的 iSCSI 工作階段狀態恢復到所需的 iSCSI 工作階段狀態。



自癒活動的日誌位於對應 Daemonset pod 上的 `trident-main` 容器中。若要查看日誌，您必須在 Trident 安裝過程中將 `debug` 設為「true」。

Trident iSCSI 自我修復功能可協助防止：

- 網路連線問題後可能會出現過期或不健康的 iSCSI 工作階段。如果工作階段過期，Trident 會等待七分鐘，然後登出並重新與入口網站建立連線。



例如，如果儲存控制器上的 CHAP 金鑰進行了輪換，而網路連線中斷，則舊的（過時的）CHAP 金鑰可能會繼續存在。自癒機制可以識別這種情況，並自動重新建立工作階段以套用更新後的 CHAP 金鑰。

- 缺少 iSCSI 工作階段
- 缺少 LUN

升級 Trident 前需考量的要點

- 如果僅使用每個節點的 igroup（在 23.04+ 中引入）、則 iSCSI 自癒功能將啟動 SCSI 匯流排上所有裝置的 SCSI 重新掃描。
- 如果僅使用後端範圍的 igroup（自 23.04 版本起已棄用）、則 iSCSI 自癒功能將啟動 SCSI 重新掃描、以尋找 SCSI 匯流排中的確切 LUN ID。
- 如果同時使用每節點 igroup 和後端範圍 igroup，iSCSI 自我修復將針對 SCSI 匯流排中的確切 LUN ID 啟動 SCSI 重新掃描。

安裝 iSCSI 工具

使用適用於您作業系統的命令安裝 iSCSI 工具。

開始之前

- Kubernetes 叢集中的每個節點都必須有一個唯一的 IQN。這是必要的前提條件。
- 如果使用 RHCOS 4.5 或更高版本、或其他與 RHEL 相容的 Linux 發行版本，並搭配 `solidfire-san` 驅動程式和 Element OS 12.5 或更早版本，請確保在 `/etc/iscsi/iscsid.conf` 中將 CHAP 驗證演算法設定為 MD5。
◦ Element 12.7 提供了符合 FIPS 標準的安全 CHAP 演算法 SHA1、SHA-256 和 SHA3-256。

```
sudo sed -i 's/^\(node.session.auth.chap_algs\).*\/\1 = MD5/'  
/etc/iscsi/iscsid.conf
```

- 當使用執行 RHEL/Red Hat Enterprise Linux CoreOS (RHCOS) 的工作節點並搭配 iSCSI PV 時，請在 StorageClass 中指定 discard mountOption 以執行即時空間回收。請參閱 ["Red Hat 說明文件"](#)。
- 請確保您已升級至最新版本的 multipath-tools。

RHEL 8+

1. 安裝下列系統套件：

```
sudo yum install -y lsscsi iscsi-initiator-utils device-mapper-  
multipath
```

2. 檢查 iscsi-initiator-utils 版本是否為 6.2.0.874-2.el7 或更新版本：

```
rpm -q iscsi-initiator-utils
```

3. 將掃描方式設定為手動：

```
sudo sed -i 's/^\(node.session.scan\) .*/\1 = manual/'  
/etc/iscsi/iscsid.conf
```

4. 啟用多重路徑：

```
sudo mpathconf --enable --with_multipathd y --find_multipaths n
```



請確保 `/etc/multipath.conf` 包含 `find_multipaths no` 在 `defaults` 之下。

5. 確保 `iscsid` 和 `multipathd` 正在運行：

```
sudo systemctl enable --now iscsid multipathd
```

6. 啟用並啟動 `iscsi`：

```
sudo systemctl enable --now iscsi
```

Ubuntu

1. 安裝下列系統套件：

```
sudo apt-get install -y open-iscsi lsscsi sg3-utils multipath-tools  
scsitools
```

2. 請檢查 `open-iscsi` 版本是否為 2.0.874-5ubuntu2.10 或更高版本（適用於 bionic）或 2.0.874-7.1ubuntu6.1 或更高版本（適用於 focal）：

```
dpkg -l open-iscsi
```

3. 將掃描方式設定為手動：

```
sudo sed -i 's/^\(node.session.scan\).*\/\1 = manual/'  
/etc/iscsi/iscsid.conf
```

4. 啟用多重路徑：

```
sudo tee /etc/multipath.conf <<-EOF  
defaults {  
    user_friendly_names yes  
    find_multipaths no  
}  
EOF  
sudo systemctl enable --now multipath-tools.service  
sudo service multipath-tools restart
```



請確保 `/etc/multipath.conf` 包含 `find_multipaths no` 在 `defaults` 之下。

5. 確保 `open-iscsi` 和 `multipath-tools` 已啟用並正在運行：

```
sudo systemctl status multipath-tools  
sudo systemctl enable --now open-iscsi.service  
sudo systemctl status open-iscsi
```



對於 Ubuntu 18.04，您必須使用 `iscsiadm` 探索目標連接埠，然後再啟動 `open-iscsi`，iSCSI 精靈才能啟動。或者，您也可以修改 `iscsi` 服務，使其自動啟動 `iscsid`。

設定或停用 iSCSI 自我修復

您可以設定以下 Trident iSCSI 自我修復設定來修復過時的工作階段：

- **iSCSI 自癒間隔**：決定 iSCSI 自癒功能的呼叫頻率（預設值：5 分鐘）。您可以設定較小的數值來提高呼叫頻率，或設定較大的數值來降低呼叫頻率。



將 iSCSI 自癒間隔設為 0 將完全停止 iSCSI 自癒功能。我們不建議停用 iSCSI 自癒功能；僅當 iSCSI 自癒功能無法按預期工作或出於偵錯目的時才應停用它。

- **iSCSI 自癒等待時間**：決定 iSCSI 自癒在登出不健康的工作階段並嘗試重新登入之前等待的時間（預設值

: 7 分鐘)。您可以將其設定為較大的數字，以便識別為不健康的工作階段必須等待較長時間才能登出，然後嘗試重新登入；或設定為較小的數字，以便更早登出並重新登入。

Helm

若要配置或變更 iSCSI 自癒設置，請在 helm 安裝或 helm 更新期間傳遞 `iscsiSelfHealingInterval` 和 `iscsiSelfHealingWaitTime` 參數。

以下範例將 iSCSI 自我修復間隔設定為 3 分鐘，自我修復等待時間設定為 6 分鐘：

```
helm install trident trident-operator-100.2602.0.tgz --set
iscsiSelfHealingInterval=3m0s --set iscsiSelfHealingWaitTime=6m0s -n
trident
```

tridentctl

若要配置或變更 iSCSI 自癒設置，請在 tridentctl 安裝或更新期間傳遞 `iscsi-self-healing-interval` 和 `iscsi-self-healing-wait-time` 參數。

以下範例將 iSCSI 自我修復間隔設定為 3 分鐘，自我修復等待時間設定為 6 分鐘：

```
tridentctl install --iscsi-self-healing-interval=3m0s --iscsi-self
-healing-wait-time=6m0s -n trident
```

NVMe/TCP Volume

使用適用於您作業系統的命令安裝 NVMe 工具。



- NVMe 需要 RHEL 9 或更新版本。
- 如果您的 Kubernetes 節點的核心版本太舊，或者您的核心版本沒有 NVMe 套件，則您可能需要將節點的核心版本更新為包含 NVMe 套件的版本。

RHEL 9

```
sudo yum install nvme-cli
sudo yum install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

Ubuntu

```
sudo apt install nvme-cli
sudo apt -y install linux-modules-extra-$(uname -r)
sudo modprobe nvme-tcp
```

驗證安裝

安裝完成後，使用以下命令驗證 Kubernetes 叢集中的每個節點是否具有唯一的 NQN：

```
cat /etc/nvme/hostnqn
```



Trident 會修改 `ctrl_device_tmo` 值，以確保 NVMe 在路徑中斷時不會放棄連線。請勿更改此設定。

SCSI over FC 磁碟區

現在您可以使用光纖通道（FC）協定和 Trident 在 ONTAP 系統上配置和管理儲存資源。

先決條件

配置 FC 所需的網路和節點設定。

網路設定

1. 取得目標介面的 WWPN。如需詳細資訊，請參閱 "[network interface show](#)"。
2. 取得啟動器（主機）上介面的 WWPN。

請參閱對應的主機作業系統公用程式。

3. 使用主機和目標的 WWPN 在 FC 交換器上設定分區。

如需相關資訊，請參閱各交換器廠商文件。

如需詳細資訊，請參閱下列 ONTAP 文件：

- "[光纖通道和 FCoE 分區概述](#)"
- "[配置 FC 和 FC-NVMe SAN 主機的方法](#)"

安裝 FC 工具

使用適用於您作業系統的命令安裝 FC 工具。

- 當使用執行 RHEL/Red Hat Enterprise Linux CoreOS（RHCOS）的工作節點並搭配 FC PVs 時，請在 `discard StorageClass` 中指定 `mountOption` 以執行即時空間回收。請參閱 "[Red Hat 說明文件](#)"。

RHEL 8+

1. 安裝下列系統套件：

```
sudo yum install -y lsscsi device-mapper-multipath
```

2. 啟用多重路徑：

```
sudo mpathconf --enable --with_multipathd y --find_multipaths n
```



請確保 `/etc/multipath.conf` 包含 `find_multipaths no` 在 `defaults` 之下。

3. 確保 `multipathd` 正在執行：

```
sudo systemctl enable --now multipathd
```

Ubuntu

1. 安裝下列系統套件：

```
sudo apt-get install -y lsscsi sg3-utils multipath-tools scsitools
```

2. 啟用多重路徑：

```
sudo tee /etc/multipath.conf <<-EOF
defaults {
    user_friendly_names yes
    find_multipaths no
}
EOF
sudo systemctl enable --now multipath-tools.service
sudo service multipath-tools restart
```



請確保 `/etc/multipath.conf` 包含 `find_multipaths no` 在 `defaults` 之下。

3. 確保 `multipath-tools` 已啟用並正在運行：

```
sudo systemctl status multipath-tools
```


Trident 操作程式

編輯 `tridentorchestrator_cr.yaml` 並設定 `cloudProvider` 為 `"Azure"`。

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "Azure"
```

Helm

以下範例會安裝 Trident 並設定 `cloudProvider`，使用環境變數 `$CP`：

```
helm install trident trident-operator-100.2602.0.tgz --create-namespace
--namespace <trident-namespace> --set cloudProvider=$CP
```

`tridentctl`

以下範例安裝 Trident 並將 `cloud-provider` 標誌設為 `Azure`：

```
tridentctl install --cloud-provider="Azure" -n trident
```

使用 AKS 的工作負載身分識別

工作負載身分可讓 Kubernetes Pod 透過以工作負載身分進行驗證來存取 Azure 資源。

如果您使用 `tridentctl` 建立或管理 Azure NetApp Files 後端，請確保已將其配置為正確的 Azure 雲端環境。

若要使用工作負載身分，您必須具備：

- 使用 AKS 部署的 Kubernetes 叢集
- 在 AKS Kubernetes 叢集上設定 Workload identity 和 oidc-issuer
- Trident 已安裝，`cloudProvider` 設定為 `"Azure"`，`cloudIdentity` 設定為工作負載識別值

Trident 操作程式

編輯 `tridentorchestrator_cr.yaml` 並設定 `cloudProvider` 為 `"Azure"`。設定 `cloudIdentity` 為 `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxx`。

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "Azure"
  cloudIdentity: 'azure.workload.identity/client-id: xxxxxxxx-xxxx-
xxxx-xxxx-xxxxxxxxxx' # Edit
```

Helm

使用下列環境變數設定 **cloud-provider** (CP) 和 **cloud-identity** (CI) 標誌的值：

```
export CP="Azure"
export CI="'azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxx'"
```

以下範例安裝 Trident 並使用 `cloudProvider` 設定 `$CP`，並使用 `cloudIdentity` 設定 `$CI`：

```
helm install trident trident-operator-100.6.0.tgz --set
cloudProvider=$CP --set cloudIdentity="$CI"
```

`tridentctl`

請使用以下環境變數設定 **cloud provider** 和 **cloud identity** 標誌的值：

```
export CP="Azure"
export CI="azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxx"
```

以下範例安裝 Trident 並將 `cloud-provider` 設定為 `$CP`，將 `cloud-identity` 設定為 `$CI`：

```
tridentctl install --cloud-provider=$CP --cloud-identity="$CI" -n
trident
```


最小組態

這是絕對最小的後端組態。使用此配置，Trident 會發現您所有的 NetApp 帳戶、容量集區和子網路（已委派給位於已設定位置的 Azure NetApp Files），並隨機將新磁碟區放置在其中一個集區和子網路上。由於 `nasType` 被省略，`nfs` 預設值會套用，後端將會佈建 NFS 磁碟區。

如果您剛開始使用 Azure NetApp Files 並進行嘗試，這種設定是理想的，但在實務中，您需要為預配的磁碟區提供額外的範圍。

```
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
  tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
  clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
  clientSecret: SECRET
  location: eastus
```

AKS 的受管理身分識別

此後端配置省略了 subscriptionID、tenantID、clientID 和 `clientSecret`，這些在使用託管身分時是可選的。

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  capacityPools:
    - resource-group-1/netapp-account-1/ultra-pool
  resourceGroups:
    - resource-group-1
  netappAccounts:
    - resource-group-1/netapp-account-1
  virtualNetwork: resource-group-1/eastus-prod-vnet
  subnet: resource-group-1/eastus-prod-vnet/eastus-anf-subnet
```


此後端配置將磁碟區放置在 Azure 的 `eastus` 位置，並具有手動 QoS 容量集區。

```
---
version: 1
storageDriverName: azure-netapp-files
backendName: anf1
location: eastus
labels:
  clusterName: test-cluster-1
  cloud: anf
  nasType: nfs
defaults:
  qosType: Manual
storage:
  - serviceLevel: Ultra
    labels:
      performance: gold
    defaults:
      maxThroughput: 10
  - serviceLevel: Premium
    labels:
      performance: silver
    defaults:
      maxThroughput: 5
  - serviceLevel: Standard
    labels:
      performance: bronze
    defaults:
      maxThroughput: 3
```

此後端組態進一步將磁碟區放置範圍縮小至單一子網路，同時也會修改部分磁碟區資源配置預設值。

```
---  
version: 1  
storageDriverName: azure-netapp-files  
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451  
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf  
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa  
clientSecret: SECRET  
location: eastus  
serviceLevel: Ultra  
capacityPools:  
  - application-group-1/account-1/ultra-1  
  - application-group-1/account-1/ultra-2  
virtualNetwork: application-group-1/eastus-prod-vnet  
subnet: application-group-1/eastus-prod-vnet/my-subnet  
networkFeatures: Standard  
nfsMountOptions: vers=3,proto=tcp,timeo=600  
limitVolumeSize: 500Gi  
defaults:  
  exportRule: 10.0.0.0/24,10.0.1.0/24,10.0.2.100  
  snapshotDir: "true"  
  size: 200Gi  
  unixPermissions: "0777"
```



```

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gold
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=gold
allowVolumeExpansion: true

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: silver
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver
allowVolumeExpansion: true

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: bronze
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=bronze
allowVolumeExpansion: true

```

SMB 磁碟區的範例定義

使用 `nasType`、`node-stage-secret-name` 和 `node-stage-secret-namespace`，您可以指定 SMB 磁碟區並提供所需的 Active Directory 憑證。

參數	說明	預設
exportRule	新磁碟區的匯出規則。必須是以逗號分隔的 IPv4 位址列表，位址可以任意組合。	"0.0.0.0/0"
snapshotDir	存取 .snapshot 目錄	true, false (預設行為可能有所不同。請明確設定) NFSv3 設定為 "false"
snapshotReserve	為快照保留的磁碟區百分比	"" (接受預設值 0)
unixPermissions	新磁碟區的 unix 權限 (4 位八進位數字)。	""

範例組態

以下範例展示了基本配置，其中大多數參數都保留預設值。這是定義後端最簡單的方法。

最小組態

這是絕對最小的後端組態。使用此組態，Trident 會在已設定的位置中，發現所有委派給 Google Cloud NetApp Volumes 的儲存資源池，並隨機將新磁碟區放置在其中一個資源池上。由於 `nasType` 被省略，`nfs` 預設值會套用，後端將會佈建 NFS 磁碟區。

如果您剛開始使用 Google Cloud NetApp Volumes 並進行嘗試，此配置是理想的選擇，但在實踐中，您可能需要為配置的磁碟區提供額外的範圍。



請將 ``<id_value>`` 和 ``<key_value>`` 替換為您的服務帳戶憑證。

在您識別並修正組態檔的問題後、您可以刪除後端並再次執行 `create` 命令。

儲存類別定義

以下是參照上述後端的基本 `StorageClass` 定義。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-nfs-sc
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
```

使用 `parameter.selector` 欄位的範例定義：

使用 `parameter.selector` 您可以為每個 `StorageClass` 指定用於託管磁碟區的 "虛擬資源池"。此磁碟區將具有所選儲存池中定義的設定項。


```

cat cert-backend.json
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "SanBackend",
  "managementLIF": "1.2.3.4",
  "svm": "vserver_test",
  "clientCertificate": "Faaaakkkkeeee...Vaaalllluuueeeee",
  "clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
  "trustedCACertificate": "QNFinfO...SiqOyN",
  "storagePrefix": "myPrefix_"
}

tridentctl create backend -f cert-backend.json -n trident
+-----+-----+-----+-----+
+-----+-----+
|   NAME   | STORAGE DRIVER |           UUID           |
STATE | VOLUMES |
+-----+-----+-----+-----+
+-----+-----+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
online |         0 |
+-----+-----+-----+-----+
+-----+-----+

```

更新驗證方法或輪換認證資料

您可以更新現有後端以使用不同的身份驗證方法或輪換其憑證。此操作雙向有效：使用使用者名稱 / 密碼的後端可以更新為使用憑證；使用憑證的後端可以更新為基於使用者名稱 / 密碼的身份驗證。為此，您必須移除現有的身份驗證方法並新增新的身份驗證方法。然後使用包含所需參數的更新後的 backend.json 檔案來執行 `tridentctl backend update`。

```

cat cert-backend-updated.json
{
"version": 1,
"storageDriverName": "ontap-san",
"backendName": "SanBackend",
"managementLIF": "1.2.3.4",
"svm": "vserver_test",
"username": "vsadmin",
"password": "password",
"storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend SanBackend -f cert-backend-updated.json -n
trident
+-----+-----+-----+
+-----+-----+
|   NAME   | STORAGE DRIVER |          UUID          |
STATE | VOLUMES |
+-----+-----+-----+
+-----+-----+
| SanBackend | ontap-san      | 586b1cd5-8cf8-428d-a76c-2872713612c1 |
online |      9 |
+-----+-----+-----+
+-----+-----+

```

註 輪換密碼時，儲存管理員必須先更新 ONTAP 上使用者的密碼。之後，後端需要進行更新。輪換證書時，可以為該使用者新增多個證書。後端隨後會更新以使用新證書，之後即可從 ONTAP 叢集中刪除舊證書。

更新後端不會中斷對已建立磁碟區的存取，也不會影響之後建立的磁碟區連線。後端更新成功表示 Trident 可以與 ONTAP 後端通訊並處理未來的磁碟區作業。

為 Trident 建立自訂 ONTAP 角色

您可以建立一個具有最低權限的 ONTAP 叢集角色，這樣您就不必使用 ONTAP 管理員角色在 Trident 中執行操作。當您在 Trident 後端組態中包含使用者名稱時，Trident 會使用您建立的 ONTAP 叢集角色來執行操作。

如需建立 Trident 自訂角色的詳細資訊，請參閱 ["Trident 自訂角色產生器"](#)。


```

cat backend-san.json
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "ontap_san_chap",
  "managementLIF": "192.168.0.135",
  "svm": "ontap_iscsi_svm",
  "useCHAP": true,
  "username": "vsadmin",
  "password": "password",
  "chapInitiatorSecret": "cl9qxUpDaTeD",
  "chapTargetInitiatorSecret": "rqxigXgkeUpDaTeD",
  "chapTargetUsername": "iJF4heBRT0TCwxyz",
  "chapUsername": "uh2aNCLsd6cNwxyz",
}

./tridentctl update backend ontap_san_chap -f backend-san.json -n trident
+-----+-----+-----+-----+
+-----+-----+
|  NAME          | STORAGE DRIVER |                               |          |
STATE | VOLUMES |                               |          |
+-----+-----+-----+-----+
+-----+-----+
| ontap_san_chap | ontap-san      | aa458f3b-ad2d-4378-8a33-1a472ffbeb5c |
online |          7 |                               |          |
+-----+-----+-----+-----+
+-----+-----+

```

現有連線將不受影響；如果 Trident 在 SVM 上更新了認證資料，這些連線將繼續保持作用中狀態。新連線使用更新的認證資料，現有連線則繼續保持作用中狀態。中斷連線並重新連線舊的 PV 將導致其使用更新的認證資料。

ONTAP SAN 設定選項和範例

了解如何在 Trident 安裝中建立和使用 ONTAP SAN 驅動程式。本節提供後端組態範例以及將後端對應至 StorageClasses 的詳細資訊。["ASA r2 系統"](#) 與其他 ONTAP 系統（ASA、AFF 和 FAS）在儲存層的實作方式上有所不同。這些差異會影響某些參數的使用方式（如註明）。["深入瞭解 ASA r2 系統與其他 ONTAP 系統之間的差異"](#)。在 Trident 後端組態中、您不需要指定系統為 ASA r2。當您選擇 `ontap-san` 作為 `storageDriverName` 時、Trident 會自動偵測 ASA r2 或其他 ONTAP 系統。如下表所示、某些後端組態參數不適用於 ASA r2 系統。

註 | ASA r2 系統僅支援 `ontap-san` 驅動程式（使用 iSCSI、NVMe/TCP 和 FC 協定）。

後端組態選項

請參閱下表以了解後端組態選項：

參數	說明	預設
version		始終為 1
storageDriveName	儲存驅動程式的名稱	ontap-san 或 ontap-san-economy
backendName	自訂名稱或儲存後端	驅動程式名稱 + "_" + dataLIF
managementLIF	<p>叢集或 SVM 管理 LIF 的 IP 位址。</p> <p>可以指定完全限定網域名稱 (FQDN)。</p> <p>如果 Trident 是使用 IPv6 旗標安裝的，則可以設定為使用 IPv6 位址。IPv6 位址必須用方括號定義，例如 [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]。</p> <p>如需無縫 MetroCluster 切換、請參閱 MetroCluster 範例。</p> <p>註：如果使用「vsadmin」認證、`managementLIF` 則必須是 SVM 的認證；如果使用「admin」認證、`managementLIF` 則必須是叢集的認證。</p>	"10.0.0.1", "[2001:1234:abcd::fefe]"
dataLIF	<p>協定 LIF 的 IP 位址。如果 Trident 是使用 IPv6 旗標安裝的，則可以設定為使用 IPv6 位址。IPv6 位址必須用方括號定義，例如 [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]。*對於 iSCSI，請勿指定此參數。*Trident 使用 "ONTAP Selective LUN Map" 來發現建立多路徑會話所需的 iSCSI LIF。如果 `dataLIF` 明確定義了此參數，則會產生警告。*對於 MetroCluster，請省略此參數。*請參閱 MetroCluster 範例。</p>	由 SVM 導出
svm	要使用的儲存虛擬機器 *MetroCluster 除外。*請參閱 MetroCluster 範例 。	如果指定了 managementLIF SVM，則衍生
useCHAP	使用 CHAP 對 ONTAP SAN 驅動程式的 iSCSI 進行驗證 [布林值]。設定為 `true` 時、Trident 會將雙向 CHAP 設定並用作後端中指定 SVM 的預設驗證。詳情請參閱 "準備使用 ONTAP SAN 驅動程式配置後端" 。不支援 FCP 或 NVMe/TCP 。	false
chapInitiatorSecret	CHAP 啟動器密碼。如果 `useCHAP=true` 則為必填項目	""
labels	要套用於磁碟區的任意 JSON 格式標籤集	""

參數	說明	預設
chapTargetInitiatorSecret	CHAP 目標啟動器密碼。如果 `useCHAP=true` 則為必填項目	""
chapUsername	入站使用者名稱。如果 `useCHAP=true` 則為必填項目	""
chapTargetUsername	目標使用者名稱。如果 `useCHAP=true` 則為必填項目	""
clientCertificate	用戶端憑證的 Base64 編碼值。用於憑證型驗證	""
clientPrivateKey	用戶端私密金鑰的 Base64 編碼值。用於憑證型驗證	""
trustedCACertificate	受信任 CA 憑證的 Base64 編碼值。此參數為可選。用於憑證型驗證。	""
username	與 ONTAP 叢集通訊所需的使用者名稱。用於基於憑證的身份驗證。有關 Active Directory 驗證，請參閱 " 使用 Active Directory 憑證對後端 SVM 進行 Trident 驗證 "。	""
password	與 ONTAP 叢集通訊所需的密碼。用於基於憑證的身份驗證。有關 Active Directory 驗證，請參閱 " 使用 Active Directory 憑證對後端 SVM 進行 Trident 驗證 "。	""
svm	要使用的儲存虛擬機器	如果指定了 managementLIF SVM，則衍生
storagePrefix	在 SVM 中配置新磁碟區時所使用的前綴。無法修改。若要更新此參數、您需要建立新的後端。	trident
aggregate	<p>用於配置的 Aggregate（選用；如果設定，則必須指派給 SVM）。對於 <code>ontap-nas-flexgroup</code> 驅動程式，此選項將被忽略。如果未指派，則可以使用任何可用的 Aggregate 來配置 FlexGroup Volume。</p> <p>註</p> <p>當 SVM 中的 Aggregate 更新時，Trident 會自動輪詢 SVM 並更新，無需重新啟動 Trident Controller。如果您已在 Trident 中設定用於配置 Volume 的特定 Aggregate，則如果該 Aggregate 被重新命名或從 SVM 中移出，後端將在輪詢 SVM Aggregate 時進入故障狀態。您必須將該 Aggregate 變更為 SVM 中已存在的 Aggregate，或將其完全移除，才能使後端恢復連線狀態。</p> <p>請勿為 ASA r2 系統指定。</p>	""

參數	說明	預設
limitAggregateUsage	如果使用率超過此百分比，則配置失敗。如果您使用的是 Amazon FSx for NetApp ONTAP 後端，請勿指定 limitAggregateUsage。提供的 `fsxadmin` 和 `vsadmin` 不包含使用 Trident 檢索 Aggregate 使用情況並加以限制所需的權限。請勿為 ASA r2 系統指定。	" (預設不強制執行)
limitVolumeSize	如果請求的磁碟區大小超過此值，則配置失敗。此外，也會限制其管理的 LUN 磁碟區大小上限。	" (預設不強制執行)
lunsPerFlexvol	每個 FlexVol 的最大 LUN 數量必須在 [50、200] 範圍內	100
debugTraceFlags	用於疑難排解的偵錯旗標。例如、{"api":false, "method":true} 除非您正在進行疑難排解並需要詳細的記錄傾印、否則請勿使用。	null
useREST	<p>布林參數，用於使用 ONTAP REST API。</p> <div style="border: 1px solid #ccc; padding: 10px; margin: 10px 0;"> <p><code>`useREST`</code> 當設定為 <code>`true`</code> 時，Trident 使用 ONTAP REST API 與後端通訊；當設定為 <code>`false`</code> 時，Trident 使用 ONTAPI (ZAPI) 呼叫與後端通訊。此功能需要 ONTAP 9.11.1 或更新版本。此外，使用的 ONTAP 登入角色必須具有 <code>`ontapi`</code> 應用程式的存取權限。預先定義的 <code>`vsadmin`</code> 和 <code>`cluster-admin`</code> 角色可滿足此要求。從 Trident 24.06 版本和 ONTAP 9.15.1 或更高版本開始，<code>`useREST`</code> 預設設定為 <code>`true`</code>；若要使用 ONTAPI (ZAPI) 呼叫，請將 <code>`useREST`</code> 變更為 <code>`false`</code>。</p> </div> <p><code>useREST</code> 完全符合 NVMe/TCP 標準。</p> <p>註 NVMe 僅支援 ONTAP REST API，不支援 ONTAPI (ZAPI)。</p> <p>如果指定、則一律設為 true (適用於 ASA r2 系統)。</p>	true 適用於 ONTAP 9.15.1 或更高版本，否則 false。
sanType	用於選擇 iscsi 適用於 iSCSI、nvme 適用於 NVMe/TCP 或 fcp 適用於 SCSI over Fibre Channel (FC)。	iscsi 如果為空

NVMe/TCP 範例

您的 ONTAP 後端必須設定一個支援 NVMe 的 SVM。這是 NVMe/TCP 的基本後端組態。

```
---  
version: 1  
backendName: NVMeBackend  
storageDriverName: ontap-san  
managementLIF: 10.0.0.1  
svm: svm_nvme  
username: vsadmin  
password: password  
sanType: nvme  
useREST: true
```

SCSI over FC (FCP) 範例

您的 ONTAP 後端必須設定一個支援 FC 的 SVM。這是 FC 的基本後端組態。

```
---  
version: 1  
backendName: fcp-backend  
storageDriverName: ontap-san  
managementLIF: 10.0.0.1  
svm: svm_fc  
username: vsadmin  
password: password  
sanType: fcp  
useREST: true
```


在這些範例中，部分儲存資源池設定了自己的 `spaceReserve`、`spaceAllocation` 和 `encryption` 值，而部分儲存資源池則覆寫了預設值。



```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSD6cNwxyz
username: vsadmin
password: <password>
defaults:
  spaceAllocation: "false"
  encryption: "false"
  qosPolicy: standard
labels:
  store: san_store
  kubernetes-cluster: prod-cluster-1
region: us_east_1
storage:
  - labels:
    protection: gold
    creditpoints: "40000"
    zone: us_east_1a
    defaults:
      spaceAllocation: "true"
      encryption: "true"
      adaptiveQosPolicy: adaptive-extreme
  - labels:
    protection: silver
    creditpoints: "20000"
    zone: us_east_1b
    defaults:
      spaceAllocation: "false"
      encryption: "true"
      qosPolicy: premium
  - labels:
    protection: bronze
    creditpoints: "5000"
    zone: us_east_1c
    defaults:
      spaceAllocation: "true"
      encryption: "false"
```

```

---
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSD6cNwxyz
username: vsadmin
password: <password>
defaults:
  spaceAllocation: "false"
  encryption: "false"
labels:
  store: san_economy_store
region: us_east_1
storage:
- labels:
  app: oracledb
  cost: "30"
  zone: us_east_1a
  defaults:
    spaceAllocation: "true"
    encryption: "true"
- labels:
  app: postgresdb
  cost: "20"
  zone: us_east_1b
  defaults:
    spaceAllocation: "false"
    encryption: "true"
- labels:
  app: mysqldb
  cost: "10"
  zone: us_east_1c
  defaults:
    spaceAllocation: "true"
    encryption: "false"
- labels:
  department: legal
  creditpoints: "5000"

```

```
zone: us_east_1c
defaults:
  spaceAllocation: "true"
  encryption: "false"
```

NVMe/TCP 範例

```
---
version: 1
storageDriverName: ontap-san
sanType: nvme
managementLIF: 10.0.0.1
svm: nvme_svm
username: vsadmin
password: <password>
useREST: true
defaults:
  spaceAllocation: "false"
  encryption: "true"
storage:
  - labels:
      app: testApp
      cost: "20"
    defaults:
      spaceAllocation: "false"
      encryption: "false"
```

將後端對應至 StorageClasses

以下 StorageClass 定義均與此相關 [\[具有虛擬資源池的後端範例\]](#)。透過該 `parameters.selector` 字段，每個 StorageClass 定義都會指定哪些虛擬池可用於託管磁碟區。磁碟區將具有所選虛擬池中定義的方面。

- 該 `protection-gold` StorageClass 將映射到 `ontap-san` 後端中的第一個虛擬資源池。這是唯一提供黃金級保護的資源池。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=gold"
  fsType: "ext4"
```

- 該 protection-not-gold StorageClass 將對應至 ontap-san 後端中的第二個和第三個虛擬資源池。這些是唯一提供金級以外保護層級的資源池。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
```

- app-mysqldb StorageClass 將對應至 `ontap-san-economy` 後端中的第三個虛擬資源池。這是唯一為 mysqldb 類型應用程式提供儲存資源池組態的資源池。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
```

- protection-silver-creditpoints-20k StorageClass 將對應至 `ontap-san` 後端的第二個虛擬資源池。這是唯一提供銀級保護和 20000 信用點數的資源池。

時、Trident 會自動偵測 AFX 系統。

註 AFX 系統僅支援 `ontap-nas` 驅動程式（使用 NFS 協定）；不支援 SMB 協定。

需求

- 對於所有 ONTAP 後端、Trident 要求至少將一個 Aggregate 指派給 SVM。
- 您可以執行多個驅動程式，並建立指向其中一個或另一個驅動程式的儲存類別。例如，您可以設定使用 `ontap-nas` 驅動程式的 Gold 類別和使用 `ontap-nas-economy` 驅動程式的 Bronze 類別。
- 所有 Kubernetes 工作節點都必須安裝適當的 NFS 工具。如需更多詳細資料，請參閱 ["這裡"](#)。
- Trident 僅支援掛載到在 Windows 節點上執行的 Pod 的 SMB 磁碟區。詳情請參閱 [準備配置 SMB Volume](#)。

驗證 ONTAP 後端

Trident 提供兩種 ONTAP 後端驗證模式。

- 基於憑證：此模式需要對 ONTAP 後端擁有足夠的權限。建議使用與預先定義安全登入角色關聯的帳戶，例如 `admin`` 或 ``vsadmin`，以確保與 ONTAP 版本的最大相容性。
- 基於憑證：此模式要求在後端安裝憑證，以便 Trident 與 ONTAP 叢集通訊。在這種情況下，後端定義必須包含客戶端憑證、金鑰以及受信任 CA 憑證（如果使用，建議）的 Base64 編碼值。

您可以更新現有後端，以在基於認證和基於憑證的方法之間移動。但是，一次只支援一種驗證方法。若要切換到不同的驗證方法，您必須從後端組態中移除現有方法。

警告

如果您嘗試同時提供憑證和憑證，則後端建立將會失敗，並出現錯誤，提示組態檔中提供了多個驗證方法。

啟用基於認證的驗證

Trident 需要 SVM 範圍 / 叢集範圍的管理員憑證才能與 ONTAP 後端通訊。建議使用標準預先定義的角色，例如 `admin`` 或 ``vsadmin`。這可確保與未來 ONTAP 版本向前相容，因為未來版本可能會公開供 Trident 版本使用的功能 API。雖然可以建立自訂安全登入角色並將其與 Trident 搭配使用，但不建議這樣做。

後端定義範例如下所示：

YAML

```
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
credentials:
  name: secret-backend-creds
```

JSON

```
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.0.0.1",
  "dataLIF": "10.0.0.2",
  "svm": "svm_nfs",
  "credentials": {
    "name": "secret-backend-creds"
  }
}
```

請注意，後端定義是唯一以純文字形式儲存認證資料的地方。建立後端之後，使用者名稱 / 密碼會使用 Base64 編碼，並儲存為 Kubernetes 機密。建立/更新後端是唯一需要知道認證資料的步驟。因此，這是僅限管理員執行的作業，由 Kubernetes/ 儲存管理員執行。

啟用基於憑證的驗證

新建和現有後端都可以使用憑證與 ONTAP 後端通訊。後端定義需要三個參數。

- `clientCertificate`：用戶端憑證的 Base64 編碼值。
- `clientPrivateKey`：關聯私密金鑰的 Base64 編碼值。
- `trustedCACertificate`：受信任 CA 憑證的 Base64 編碼值。如果使用受信任的 CA，則必須提供此參數。如果未使用受信任的 CA，則可以忽略此參數。

典型的工作流程包括以下步驟。

步驟

1. 產生客戶端憑證和金鑰。產生時，將 Common Name (CN) 設定為要進行驗證的 ONTAP 使用者。

```
openssl req -x509 -nodes -days 1095 -newkey rsa:2048 -keyout k8senv.key
-out k8senv.pem -subj "/C=US/ST=NC/L=RTP/O=NetApp/CN=vsadmin"
```

2. 將信任的 CA 憑證新增至 ONTAP 叢集。儲存管理員可能已經處理此作業。如果未使用信任的 CA，請忽略。

```
security certificate install -type server -cert-name <trusted-ca-cert-name> -vserver <vserver-name>
ssl modify -vserver <vserver-name> -server-enabled true -client-enabled true -common-name <common-name> -serial <SN-from-trusted-CA-cert> -ca <cert-authority>
```

3. 在 ONTAP 叢集上安裝用戶端憑證和金鑰（來自步驟 1）。

```
security certificate install -type client-ca -cert-name <certificate-name> -vserver <vserver-name>
security ssl modify -vserver <vserver-name> -client-enabled true
```

4. 確認 ONTAP 安全登入角色支援 cert 驗證方法。

```
security login create -user-or-group-name vsadmin -application ontapi -authentication-method cert -vserver <vserver-name>
security login create -user-or-group-name vsadmin -application http -authentication-method cert -vserver <vserver-name>
```

5. 使用產生的憑證測試驗證。將 <ONTAP Management LIF> 和 <vserver name> 替換為 Management LIF IP 和 SVM 名稱。您必須確保 LIF 的服務原則已設為 default-data-management。

```
curl -X POST -Lk https://<ONTAP-Management-LIF>/servlets/netapp.servlets.admin.XMLrequest_filer --key k8senv.key --cert ~/k8senv.pem -d '<?xml version="1.0" encoding="UTF-8"?><netapp xmlns="http://www.netapp.com/filer/admin" version="1.21" vfiler="<vserver-name>"><vserver-get></vserver-get></netapp>'
```

6. 使用 Base64 對憑證、金鑰和受信任的 CA 憑證進行編碼。

```
base64 -w 0 k8senv.pem >> cert_base64
base64 -w 0 k8senv.key >> key_base64
base64 -w 0 trustedca.pem >> trustedca_base64
```

7. 使用上一步獲得的值建立後端。

```
cat cert-backend-updated.json
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "NasBackend",
  "managementLIF": "1.2.3.4",
  "dataLIF": "1.2.3.8",
  "svm": "vserver_test",
  "clientCertificate": "Faaaakkkkeeee...Vaaalllluuueeee",
  "clientPrivateKey": "LS0tFaKE...0VaLuES0tLS0K",
  "storagePrefix": "myPrefix_"
}

#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
+-----+-----+-----+
+-----+-----+
|   NAME   | STORAGE DRIVER |           UUID           |
STATE | VOLUMES |
+-----+-----+-----+
+-----+-----+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |      9 |
+-----+-----+-----+
+-----+-----+
```

更新驗證方法或輪換認證資料

您可以更新現有後端以使用不同的身份驗證方法或輪換其憑證。此操作雙向有效：使用使用者名稱 / 密碼的後端可以更新為使用憑證；使用憑證的後端可以更新為基於使用者名稱 / 密碼的身份驗證。為此，您必須移除現有的身份驗證方法並新增新的身份驗證方法。然後使用包含所需參數的更新後的 backend.json 檔案來執行 tridentctl update backend。

```
cat cert-backend-updated.json
```

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "NasBackend",
  "managementLIF": "1.2.3.4",
  "dataLIF": "1.2.3.8",
  "svm": "vserver_test",
  "username": "vsadmin",
  "password": "password",
  "storagePrefix": "myPrefix_"
}
```

```
#Update backend with tridentctl
tridentctl update backend NasBackend -f cert-backend-updated.json -n
trident
```

NAME	STORAGE DRIVER	UUID
NasBackend	ontap-nas	98e19b74-aec7-4a3d-8dcf-128e5033b214

```
STATE | VOLUMES |
online | 9 |
```

註

輪換密碼時，儲存管理員必須先更新 ONTAP 上使用者的密碼。之後，後端需要進行更新。輪換證書時，可以為該使用者新增多個證書。後端隨後會更新以使用新證書，之後即可從 ONTAP 叢集中刪除舊證書。

更新後端不會中斷對已建立磁碟區的存取，也不會影響之後建立的磁碟區連線。後端更新成功表示 Trident 可以與 ONTAP 後端通訊並處理未來的磁碟區作業。

為 Trident 建立自訂 ONTAP 角色

您可以建立一個具有最低權限的 ONTAP 叢集角色，這樣您就不必使用 ONTAP 管理員角色在 Trident 中執行操作。當您在 Trident 後端組態中包含使用者名稱時，Trident 會使用您建立的 ONTAP 叢集角色來執行操作。

如需建立 Trident 自訂角色的詳細資訊，請參閱 ["Trident 自訂角色產生器"](#)。

使用 ONTAP CLI

1. 使用以下命令建立新角色：

```
security login role create <role_name\> -cmddirname "command" -access all  
-vserver <svm_name\>
```

2. 為 Trident 使用者建立使用者名稱：

```
security login create -username <user_name\> -application ontapi  
-authmethod <password\> -role <name_of_role_in_step_1\> -vserver  
<svm_name\> -comment "user_description"
```

3. 將角色對應至使用者：

```
security login modify username <user_name\> -vserver <svm_name\> -role  
<role_name\> -application ontapi -application console -authmethod  
<password\>
```

使用 System Manager

在 ONTAP System Manager 中執行下列步驟：

1. 建立自訂角色：

- a. 若要在叢集層級建立自訂角色，請選取 **Cluster > Settings**。

(或) 若要在 SVM 層級建立自訂角色、請選取 **Storage > Storage VMs > required svm > Settings > Users and Roles**。

- b. 選擇 **Users and Roles** 旁邊的箭頭圖示 (→)。
- c. 在 **Roles** 下選擇 **+Add**。
- d. 定義角色規則，然後點選 **Save**。

2. 將角色對應到 Trident 使用者：+ 在 **Users and Roles** 頁面上執行下列步驟：

- a. 在 **Users** 下方選擇 Add 圖示 +。
- b. 選擇所需的使用者名稱，然後在 **Role** 下拉式選單中選擇角色。
- c. 按一下 **Save**。

如需更多資訊、請參閱下列頁面：

- ["用於管理 ONTAP 的自訂角色"](#) 或 ["定義自訂角色"](#)
- ["使用角色和使用者"](#)

管理 NFS 匯出原則

Trident 使用 NFS 匯出原則來控制對其所配置之磁碟區的存取。

Trident 在處理匯出原則時提供兩種選項：

- Trident 可以動態管理匯出策略；在這種模式下，儲存管理員指定 CIDR 區塊列表，這些 CIDR 區塊代表允許的 IP 位址。Trident 會在發佈時自動將落入這些範圍內的適用節點 IP 位址新增至匯出策略。或者，如果沒有指定 CIDR，則會將磁碟區發佈到的節點上找到的所有全域作用域單播 IP 位址新增至匯出策略。
- 儲存管理員可以建立匯出原則並手動新增規則。除非在組態中指定不同的匯出原則名稱、否則 Trident 會使用預設的匯出原則。

動態管理匯出原則

Trident 提供了動態管理 ONTAP 後端匯出原則的功能。這使得儲存管理員能夠為工作節點 IP 指定允許的位址空間，而無需手動定義明確規則。這大大簡化了匯出原則的管理；對匯出原則的修改不再需要在儲存叢集上進行手動干預。此外，這還有助於將對儲存叢集的存取限制在僅掛載磁碟區且 IP 位址在指定範圍內的工作節點上，從而支援精細和自動化管理。

註

使用動態匯出策略時，請勿使用網路位址轉換（NAT）。啟用 NAT 後，儲存控制器看到的是前端 NAT 位址，而不是實際的 IP 主機位址，因此，如果在匯出規則中找不到符合項，則會拒絕存取。

範例

必須使用兩種組態選項。以下是後端定義範例：

```

---
version: 1
storageDriverName: ontap-nas-economy
backendName: ontap_nas_auto_export
managementLIF: 192.168.0.135
svm: svm1
username: vsadmin
password: password
autoExportCIDRs:
  - 192.168.0.0/24
autoExportPolicy: true

```

註

使用此功能時，必須確保 SVM 中的根接合點已預先建立匯出原則，且該原則包含允許節點 CIDR 區塊的匯出規則（例如預設匯出原則）。請務必遵循 NetApp 建議的最佳實務做法，為 Trident 專用 SVM。

以下以上述範例為例、說明此功能的工作原理：

- autoExportPolicy 已設定為 true。這表示 Trident 會為使用此後端為 svm1 SVM 配置的每個磁碟區建立一個匯出策略，並使用 autoexportCIDRs 位址區塊處理規則的新增和刪除。在磁碟區連接到節點之前，該磁碟區使用一個空的匯出策略，其中沒有任何規則來防止對該磁碟區的未經授權的存取。當磁碟區發佈到節點時，Trident 會建立一個與底層 qtree 同名的匯出策略，該 qtree 包含指定 CIDR 區塊內的節點 IP 位址。這些 IP 位址也會加入到父 FlexVol 磁碟區使用的匯出策略中。
 - 例如：
 - 後端 UUID 403b5326-8482-40db-96d0-d83fb3f4daec

- 安全 SMB 支援 `ontap-nas` 託管磁碟區匯入，但不適用於非託管磁碟區匯入。

步驟

1. 請在 TridentBackendConfig 中指定 adAdminUser，如下例所示：

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.193.176.x
  svm: svm0
  useREST: true
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

2. 在儲存類別中新增註解。

將 `trident.netapp.io/smbShareAdUser` 註解新增至儲存類別，以啟用安全 SMB 而不會失敗。為註解 `trident.netapp.io/smbShareAdUser` 指定的使用者值應與 `smbcreds` 密鑰中指定的使用者名稱相同。您可以為 `smbShareAdUserPermission` 選擇下列其中一項：`full_control`、`change` 或 `read`。預設權限為 `full_control`。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

1. 建立 PVC。

以下範例建立 PVC：

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/snapshotDirectory: "true"
    trident.netapp.io/smbShareAccessControl: |
      read:
        - tridentADtest
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

ONTAP NAS 設定選項和範例

學習如何在 Trident 安裝中建立和使用 ONTAP NAS 驅動程式。本節提供後端組態範例以及將後端對應至 StorageClasses 的詳細資訊。從 25.10 版本開始，NetApp Trident 支援 ["NetApp AFX 儲存系統"](#)。NetApp AFX 儲存系統與其他基於 ONTAP 的系統（ASA、AFF 和 FAS）在儲存層的實作方式上有所不同。

註 | 僅支援 ontap-nas 驅動程式（使用 NFS 協定）用於 NetApp AFX 系統；不支援 SMB 協定。

後端組態選項

在 Trident 後端組態中、您不需要指定系統是 NetApp AFX 儲存系統。當您選取 `ontap-nas` 作為 `storageDriverName` 時、Trident 會自動偵測 AFX 儲存系統。部分後端組態參數不適用於 AFX 儲存系統。

下表顯示後端組態選項：

參數	說明	預設
version		始終為 1

參數	說明	預設
password	連接到叢集 / SVM 的密碼。用於基於憑證的驗證。有關 Active Directory 驗證、請參閱 " 使用 Active Directory 憑證對後端 SVM 進行 Trident 驗證 "。	
storagePrefix	<p>在 SVM 中配置新磁碟區時所使用的前綴。設定後無法更新</p> <p>註 當使用 ontap-nas-economy 並且 storagePrefix 為 24 個字元或更長時，qtree 將不會嵌入儲存前綴，儘管它會包含在磁碟區名稱中。</p>	"Trident"
aggregate	<p>用於配置的 Aggregate（選用；如果設定，則必須指派給 SVM）。對於 ontap-nas-flexgroup 驅動程式，此選項將被忽略。如果未指派，則可以使用任何可用的 Aggregate 來配置 FlexGroup Volume。</p> <p>註 當 SVM 中的 Aggregate 更新時，Trident 會自動輪詢 SVM 並更新，無需重新啟動 Trident Controller。如果您已在 Trident 中設定用於配置 Volume 的特定 Aggregate，則如果該 Aggregate 被重新命名或從 SVM 中移出，後端將在輪詢 SVM Aggregate 時進入故障狀態。您必須將該 Aggregate 變更為 SVM 中已存在的 Aggregate，或將其完全移除，才能使後端恢復連線狀態。</p> <p>請勿為 AFX 儲存系統指定此規則。</p>	""
limitAggregateUsage	如果使用率超過此百分比，則配置失敗。不適用於 Amazon FSx for ONTAP 。請勿為 AFX 儲存系統指定此規則。	"（預設不強制執行）

參數	說明	預設
flexgroupAggregateList	<p>用於配置的 Aggregate 清單（選用；如果設定，則必須指派給 SVM）。指派給 SVM 的所有 Aggregate 都將用於配置 FlexGroup Volume。ontap-nas-flexgroup 儲存驅動程式支援此功能。</p> <p>註 當 SVM 中的 Aggregate 清單更新時，Trident 會自動輪詢 SVM 來更新該清單，無需重新啟動 Trident Controller。如果您已在 Trident 中設定特定的 Aggregate 清單來配置磁碟區，若該 Aggregate 清單已重新命名或從 SVM 中移出，Trident 在輪詢 SVM Aggregate 時後端將進入故障狀態。您必須將該 Aggregate 清單變更為 SVM 中存在的清單，或將其完全移除，才能使後端恢復連線狀態。</p>	""
limitVolumeSize	如果請求的磁碟區大小超過此值，則配置失敗。	"（預設不強制執行）
debugTraceFlags	用於疑難排解的偵錯旗標。例如、{"api":false, "method":true}除非您正在進行疑難排解並需要詳細的記錄傾印、否則請勿使用 debugTraceFlags。	null
nasType	配置 NFS 或 SMB 磁碟區的建立。選項為 nfs、smb 或 null。設定為 null 則預設建立 NFS 磁碟區。如果指定，對於 AFX 儲存系統，請始終設定為 `nfs`。	nfs
nfsMountOptions	以逗號分隔的 NFS 掛載選項清單。Kubernetes 持久性磁碟區的掛載選項通常在儲存類別中指定，但如果儲存類別中未指定任何掛載選項，Trident 將回退到使用儲存後端設定檔中指定的掛載選項。如果儲存類別和設定檔中均未指定掛載選項，Trident 將不會在關聯的持久性磁碟區上設定任何掛載選項。	""
qtreesPerFlexvol	每個 FlexVol 的最大 Qtree 數量必須在 [50, 300] 範圍內	"200"
smbShare	您可以指定以下選項之一：使用 Microsoft Management Console 或 ONTAP CLI 建立的 SMB 共用名稱；允許 Trident 建立 SMB 共用的名稱；或者您可以將此參數留空以封鎖對磁碟區的公共共用存取。對於內部部署 ONTAP，此參數為選用項目。對於 Amazon FSx for ONTAP 後端，此參數為必要項目且不能為空白。	smb-share

參數	說明	預設
useREST	布林參數，用於使用 ONTAP REST API。useREST 當設定為 `true` 時，Trident 使用 ONTAP REST API 與後端通訊；當設定為 `false` 時，Trident 使用 ONTAPI (ZAPI) 呼叫與後端通訊。此功能需要 ONTAP 9.11.1 或更新版本。此外，使用的 ONTAP 登入角色必須具有 `ontapi` 應用程式的存取權限。預先定義的 `vsadmin` 和 `cluster-admin` 角色可滿足此要求。從 Trident 24.06 版本和 ONTAP 9.15.1 或更高版本開始，`useREST` 預設設定為 `true`；若要使用 ONTAPI (ZAPI) 呼叫，請將 useREST 變更為 `false`。如果指定，對於 AFX 儲存系統，請始終設定為 true 。	true 適用於 ONTAP 9.15.1 或更高版本，否則 false。
limitVolumePoolSize	在 ontap-nas-economy 後端中使用 Qtree 時可請求的最大 FlexVol 大小。	" (預設不強制執行)
denyNewVolumePools	限制 `ontap-nas-economy` 後端建立新 FlexVol 磁碟區來存放其 Qtree。僅使用預先存在的 Flexvol 來配置新的 PV。	
adAdminUser	擁有對 SMB 共用完全存取權限的 Active Directory 管理員使用者或使用者群組。使用此參數可授予對 SMB 共用的完全控制權限。	

磁碟區配置的後端組態選項

您可以使用 defaults 配置部分中的這些選項來控制預設配置。例如、請參閱下面的組態範例。

參數	說明	預設
spaceAllocation	Qtree 的空間分配	"true"
spaceReserve	空間保留模式；「none」（精簡）或「volume」（完整）	"none"
snapshotPolicy	要使用的 Snapshot 原則	"none"
qosPolicy	要為建立的磁碟區指派 QoS 策略群組。為每個儲存資源池 / 後端選擇 qosPolicy 或 adaptiveQosPolicy 其中之一	""
adaptiveQosPolicy	為建立的磁碟區指派的自適應 QoS 原則群組。每個儲存資源池/後端可選擇 qosPolicy 或 adaptiveQosPolicy 其中之一。ontap-nas-economy 不支援。	""
snapshotReserve	為快照保留的磁碟區百分比	若 `snapshotPolicy` 為「none」，則為「0」，否則為「」
splitOnClone	建立時將複本從其父項分割	"false"

參數	說明	預設
encryption	在新磁碟區上啟用 NetApp Volume Encryption (NVE)；預設值為 false。要使用此選項，叢集必須已獲得 NVE 許可並啟用 NVE。如果後端啟用了 NAE，則在 Trident 中佈建的任何磁碟區都會啟用 NAE。如需詳細資訊，請參閱： "Trident 與 NVE 和 NAE 的運作方式" 。	"false"
tieringPolicy	分層策略使用 "none"	
unixPermissions	新磁碟區模式	NFS 磁碟區為「777」；SMB 磁碟區為空（不適用）
snapshotDir	控制對 .snapshot 目錄的存取	true, false（明確設定）。
exportPolicy	要使用的匯出原則	"default"
securityStyle	新磁碟區的安全樣式。NFS 支援 `mixed` 和 `unix` 安全樣式。SMB 支援 `mixed` 和 `ntfs` 安全樣式。	NFS 預設值為 unix。SMB 預設值為 ntfs。
nameTemplate	用於建立自訂磁碟區名稱的範本。	""

註

搭配 Trident 使用 QoS 策略群組需要 ONTAP 9.8 或更新版本。您應該使用非共享的 QoS 原則群組，並確保該原則群組分別套用於每個成員。共享的 QoS 策略群組會強制限制所有工作負載的總吞吐量上限。

Volume 配置範例

以下是定義預設值的範例：

```

---
version: 1
storageDriverName: ontap-nas
backendName: customBackendName
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
labels:
  k8scluster: dev1
  backend: dev1-nasbackend
svm: trident_svm
username: cluster-admin
password: <password>
limitAggregateUsage: 80%
limitVolumeSize: 50Gi
nfsMountOptions: nfsvers=4
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: premium
  exportPolicy: myk8scluster
  snapshotPolicy: default
  snapshotReserve: "10"

```

對於 `ontap-nas` 和 `ontap-nas-flexgroups`，Trident 現在使用新的計算方法，以確保 FlexVol 的大小與 `snapshotReserve` 百分比和 PVC 正確匹配。當使用者要求 PVC 時，Trident 會使用新的計算方法建立更大的原始 FlexVol。此計算方法可確保使用者在 PVC 中獲得其請求的可寫入空間，而不是少於其請求的空間。在 v21.07 之前，當使用者請求 PVC（例如 5 GiB）且 `snapshotReserve` 百分比為 50% 時，他們只能獲得 2.5 GiB 的可寫空間。這是因為使用者要求的是整個磁碟區，而 `snapshotReserve` 是其百分比。在 Trident 21.07 中，使用者要求的是可寫入空間，Trident 將該 `snapshotReserve` 數值定義為整個磁碟區的百分比。這不適用於 `ontap-nas-economy`。請參閱以下範例以了解其工作原理：

計算方法如下：

```

Total volume size = <PVC requested size> / (1 - (<snapshotReserve
percentage> / 100))

```

對於 `snapshotReserve = 50%` 以及 PVC 請求 = 5 GiB，總磁碟區大小為 $5/0.5 = 10$ GiB，可用大小為 5 GiB，這正是使用者在 PVC 請求中所要求的大小。`volume show` 命令應顯示與此範例類似的結果：

Vserver	Volume	Aggregate	State	Type	Size	Available	Used%
	_pvc_89f1c156_3801_4de4_9f9d_034d54c395f4		online	RW	10GB	5.00GB	0%
	_pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba		online	RW	1GB	511.8MB	0%

2 entries were displayed.

升級 Trident 時，先前安裝的現有後端將按照上述說明配置磁碟區。對於升級前建立的磁碟區，您需要調整其大小才能使變更生效。例如，先前使用 `snapshotReserve=50` 建立的 2 GiB PVC 會產生提供 1 GiB 可寫入空間的磁碟區。將磁碟區大小調整為 3 GiB 後，應用程式將在 6 GiB 磁碟區上獲得 3 GiB 的可寫入空間。

最小組態範例

以下範例展示了基本配置，其中大多數參數都保留預設值。這是定義後端最簡單的方法。

註

如果您在 NetApp ONTAP 上使用 Amazon FSx 搭配 Trident，建議為 LIF 指定 DNS 名稱而非 IP 位址。

ONTAP NAS 經濟範例

```

---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password

```

ONTAP NAS FlexGroup 範例

```

---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
username: vsadmin
password: password

```

MetroCluster 範例

您可以設定後端、以避免在 "SVM 複製與復原" 期間進行切換和切換後手動更新後端定義。

為了實現無縫切換和切換回，請使用 `managementLIF` 指定 SVM 並省略 `dataLIF` 和 `svm` 參數。例如：

```
---  
version: 1  
storageDriverName: ontap-nas  
managementLIF: 192.168.1.66  
username: vsadmin  
password: password
```

SMB 磁碟區範例

```
---  
version: 1  
backendName: ExampleBackend  
storageDriverName: ontap-nas  
managementLIF: 10.0.0.1  
nasType: smb  
securityStyle: ntfs  
unixPermissions: ""  
dataLIF: 10.0.0.2  
svm: svm_nfs  
username: vsadmin  
password: password
```

基於憑證的驗證範例

這是一個最小的後端設定範例。clientCertificate、clientPrivateKey 和 trustedCACertificate（如果使用受信任的 CA，則為可選）分別填充在 backend.json 中，並分別接受客戶端憑證、私鑰和受信任的 CA 憑證的 base64 編碼值。

```
---
version: 1
backendName: DefaultNASBackend
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.15
svm: nfs_svm
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
storagePrefix: myPrefix_
```

自動匯出原則範例

本範例示範如何指示 Trident 使用動態匯出原則來自動建立和管理匯出原則。這對於 `ontap-nas-economy` 和 `ontap-nas-flexgroup` 驅動程式的運作方式相同。

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
dataLIF: 10.0.0.2
svm: svm_nfs
labels:
  k8scluster: test-cluster-east-1a
  backend: test1-nasbackend
autoExportPolicy: true
autoExportCIDRs:
- 10.0.0.0/24
username: admin
password: password
nfsMountOptions: nfsvers=4
```

IPv6 位址範例

此範例展示 `managementLIF` 使用 IPv6 位址。

```
---  
version: 1  
storageDriverName: ontap-nas  
backendName: nas_ipv6_backend  
managementLIF: "[5c5d:5edf:8f:7657:bef8:109b:1b41:d491]"  
labels:  
  k8scluster: test-cluster-east-1a  
  backend: test1-ontap-ipv6  
svm: nas_ipv6_svm  
username: vsadmin  
password: password
```

使用 SMB 磁碟區的 Amazon FSx for ONTAP 範例

使用 SMB 磁碟區的 FSx for ONTAP 需要 `smbShare` 參數。

```
---  
version: 1  
backendName: SMBBackend  
storageDriverName: ontap-nas  
managementLIF: example.mgmt.fqdn.aws.com  
nasType: smb  
dataLIF: 10.0.0.15  
svm: nfs_svm  
smbShare: smb-share  
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2  
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX  
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz  
storagePrefix: myPrefix_
```

使用 nameTemplate 的後端組態範例

```
---
version: 1
storageDriverName: ontap-nas
backendName: ontap-nas-backend
managementLIF: <ip address>
svm: svm0
username: <admin>
password: <password>
defaults:
  nameTemplate:
    "{{.volume.Name}}_{{.labels.cluster}}_{{.volume.Namespace}}_{{.vo\
      lume.RequestName}}"
  labels:
    cluster: ClusterA
  PVC: "{{.volume.Namespace}}_{{.volume.RequestName}}"
```

具有虛擬資源池的後端範例

在下方所示的範例後端定義檔中，所有儲存池都設定了特定的預設值，例如 `spaceReserve` 為 none、`spaceAllocation` 為 false 和 `encryption` 為 false。虛擬資源池在儲存區段中定義。

Trident 在「備註」欄位中設定配置標籤。備註可以針對 `ontap-nas` 在 FlexVol 上設定，或針對 `ontap-nas-flexgroup` 在 FlexGroup 上設定。Trident 在配置時會將虛擬資源池上的所有標籤複製到儲存磁碟區。為了方便起見，儲存管理員可以為每個虛擬資源池定義標籤，並按標籤將磁碟區分組。

在這些範例中，部分儲存資源池設定了自己的 `spaceReserve`、`spaceAllocation` 和 `encryption` 值，而部分儲存資源池則覆寫了預設值。

```
---
version: 1
storageDriverName: ontap-nas
managementLIF: 10.0.0.1
svm: svm_nfs
username: admin
password: <password>
nfsMountOptions: nfsvers=4
defaults:
  spaceReserve: none
  encryption: "false"
  qosPolicy: standard
labels:
  store: nas_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
  - labels:
    app: msoffice
    cost: "100"
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: "true"
      unixPermissions: "0755"
      adaptiveQosPolicy: adaptive-premium
  - labels:
    app: slack
    cost: "75"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
    department: legal
    creditpoints: "5000"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
```

```
  app: wordpress
  cost: "50"
  zone: us_east_1c
  defaults:
    spaceReserve: none
    encryption: "true"
    unixPermissions: "0775"
- labels:
  app: mysqlldb
  cost: "25"
  zone: us_east_1d
  defaults:
    spaceReserve: volume
    encryption: "false"
    unixPermissions: "0775"
```

ONTAP NAS FlexGroup 範例

```
---
version: 1
storageDriverName: ontap-nas-flexgroup
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: <password>
defaults:
  spaceReserve: none
  encryption: "false"
labels:
  store: flexgroup_store
  k8scluster: prod-cluster-1
region: us_east_1
storage:
- labels:
  protection: gold
  creditpoints: "50000"
  zone: us_east_1a
  defaults:
    spaceReserve: volume
    encryption: "true"
    unixPermissions: "0755"
- labels:
  protection: gold
  creditpoints: "30000"
  zone: us_east_1b
  defaults:
    spaceReserve: none
    encryption: "true"
    unixPermissions: "0755"
- labels:
  protection: silver
  creditpoints: "20000"
  zone: us_east_1c
  defaults:
    spaceReserve: none
    encryption: "true"
    unixPermissions: "0775"
- labels:
  protection: bronze
  creditpoints: "10000"
  zone: us_east_1d
```

```
defaults:  
  spaceReserve: volume  
  encryption: "false"  
  unixPermissions: "0775"
```

```
---
version: 1
storageDriverName: ontap-nas-economy
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: <password>
defaults:
  spaceReserve: none
  encryption: "false"
labels:
  store: nas_economy_store
region: us_east_1
storage:
  - labels:
    department: finance
    creditpoints: "6000"
    zone: us_east_1a
    defaults:
      spaceReserve: volume
      encryption: "true"
      unixPermissions: "0755"
  - labels:
    protection: bronze
    creditpoints: "5000"
    zone: us_east_1b
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0755"
  - labels:
    department: engineering
    creditpoints: "3000"
    zone: us_east_1c
    defaults:
      spaceReserve: none
      encryption: "true"
      unixPermissions: "0775"
  - labels:
    department: humanresource
    creditpoints: "2000"
    zone: us_east_1d
    defaults:
```

```
spaceReserve: volume
encryption: "false"
unixPermissions: "0775"
```

將後端對應至 StorageClasses

以下 StorageClass 定義均指涉[具有虛擬資源池的後端範例](#)。透過該 `parameters.selector` 字段，每個 StorageClass 定義都會指定哪些虛擬池可用於託管磁碟區。磁碟區將具有所選虛擬池中定義的方面。

- `protection-gold` StorageClass 將對應至 `ontap-nas-flexgroup` 後端的第一個和第二個虛擬資源池。這些是唯一提供金級保護的資源池。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=gold"
  fsType: "ext4"
```

- 該 `protection-not-gold` StorageClass 將映射到 `ontap-nas-flexgroup` 後端的第三和第四個虛擬資源池。這些是唯一提供 gold 以外保護等級的資源池。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-not-gold
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection!=gold"
  fsType: "ext4"
```

- `app-mysqldb` StorageClass 將對應至 `ontap-nas` 後端的第四個虛擬資源池。這是唯一為 `mysqldb` 類型應用程式提供儲存資源池組態的資源池。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: app-mysqldb
provisioner: csi.trident.netapp.io
parameters:
  selector: "app=mysqldb"
  fsType: "ext4"
```

- protection-silver-creditpoints-20k StorageClass 將對應到 ontap-nas-flexgroup 後端中的第三個虛擬資源池。這是唯一提供銀級保護和 20000 信用點數的資源池。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: protection-silver-creditpoints-20k
provisioner: csi.trident.netapp.io
parameters:
  selector: "protection=silver; creditpoints=20000"
  fsType: "ext4"
```

- creditpoints-5k StorageClass 將對應至 ontap-nas 後端的第三個虛擬資源池和 ontap-nas-economy 後端的第二個虛擬資源池。這些是唯一提供 5000 creditpoints 的資源池。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: creditpoints-5k
provisioner: csi.trident.netapp.io
parameters:
  selector: "creditpoints=5000"
  fsType: "ext4"
```

Trident 將決定選擇哪個虛擬資源池，並確保符合儲存需求。

初始配置後更新 dataLIF

初始設定完成後，您可以執行以下命令來變更 dataLIF，以提供包含更新 dataLIF 的新後端 JSON 檔案。

```
tridentctl update backend <backend-name> -f <path-to-backend-json-file-
with-updated-dataLIF>
```

註

如果 PVC 連接到一個或多個 Pod、則必須關閉所有對應的 Pod、然後再將其重新啟動、以便讓新的 dataLIF 生效。

安全 SMB 範例

使用 **ontap-nas** 驅動程式進行後端組態

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

使用 **ontap-nas-economy** 驅動程式進行後端組態

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas-economy
  managementLIF: 10.0.0.1
  svm: svm2
  nasType: smb
  defaults:
    adAdminUser: tridentADtest
  credentials:
    name: backend-tbc-ontap-invest-secret
```

後端配置與儲存資源池

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.0.0.1
  svm: svm0
  useREST: false
  storage:
  - labels:
      app: msoffice
    defaults:
      adAdminUser: tridentADuser
  nasType: smb
  credentials:
    name: backend-tbc-ontap-invest-secret
```

使用 **ontap-nas** 驅動程式的儲存類別範例

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADtest
parameters:
  backendType: ontap-nas
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

註

請務必新增 `annotations` 以啟用安全 SMB。無論後端或 PVC 中如何配置、如果沒有這些註釋、安全 SMB 都無法正常運作。

使用 **ontap-nas-economy** 驅動程式的儲存類別範例

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-smb-sc
  annotations:
    trident.netapp.io/smbShareAdUserPermission: change
    trident.netapp.io/smbShareAdUser: tridentADuser3
parameters:
  backendType: ontap-nas-economy
  csi.storage.k8s.io/node-stage-secret-name: smbcreds
  csi.storage.k8s.io/node-stage-secret-namespace: trident
  trident.netapp.io/nasType: smb
provisioner: csi.trident.netapp.io
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

包含單一 AD 使用者的 PVC 範例

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-pvc4
  namespace: trident
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      change:
        - tridentADtest
      read:
        - tridentADuser
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-smb-sc
```

包含多個 AD 使用者的 PVC 範例

```

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-test-pvc
  annotations:
    trident.netapp.io/smbShareAccessControl: |
      full_control:
        - tridentTestuser
        - tridentuser
        - tridentTestuser1
        - tridentuser1
      change:
        - tridentADuser
        - tridentADuser1
        - tridentADuser4
        - tridentTestuser2
      read:
        - tridentTestuser2
        - tridentTestuser3
        - tridentADuser2
        - tridentADuser3
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi

```

Amazon FSx for NetApp ONTAP

將 **Trident** 與 **Amazon FSx for NetApp ONTAP** 搭配使用

"Amazon FSx for NetApp ONTAP" 是一項完全託管的 AWS 服務，可執行由 NetApp ONTAP 儲存作業系統提供支援的檔案系統。它提供 ONTAP 的功能、效能和管理能力，同時兼具 AWS 的可擴充性和營運簡易性。檔案系統是 Amazon FSx 中的主要資源，類似於內部部署的 ONTAP 叢集。每個檔案系統包含一個或多個儲存虛擬機器 (SVM)，每個 SVM 包含一個或多個用於儲存檔案和目錄的磁碟區。這種整合使得在 Amazon Elastic Kubernetes Service (EKS) 中執行的 Kubernetes 叢集能夠為區塊和檔案工作負載佈建由 ONTAP 支援的持續磁碟區。

需求

此外 "**Trident 需求**"，要將 FSx for ONTAP 與 Trident 整合，您還需要：

- 已安裝 `kubectl` 的現有 Amazon EKS 叢集或自我管理 Kubernetes 叢集。

尋找更多資訊

- ["Amazon FSx for NetApp ONTAP 文件"](#)
- ["關於 Amazon FSx for NetApp ONTAP 的部落格文章"](#)

建立 IAM 角色和 AWS Secret

您可以設定 Kubernetes Pod 以透過 AWS IAM 角色進行驗證來存取 AWS 資源，而不是提供明確的 AWS 憑證。

註 若要使用 AWS IAM 角色進行驗證、您必須擁有使用 EKS 部署的 Kubernetes 叢集。

建立 AWS Secrets Manager 密碼

由於 Trident 將針對 FSx vserver 發出 API 來為您管理儲存設備，因此需要相應的認證資料。傳遞這些認證資料的安全方法是透過 AWS Secrets Manager 密碼。因此，如果您還沒有密碼，則需要建立一個包含 vsadmin 帳戶認證資料的 AWS Secrets Manager 密碼。

此範例建立一個 AWS Secrets Manager 密碼來儲存 Trident CSI 認證：

```
aws secretsmanager create-secret --name trident-secret --description
"Trident CSI credentials"\
  --secret-string
"{\"username\": \"vsadmin\", \"password\": \"<svmpassword>\"}"
```

建立 IAM 政策

Trident 也需要 AWS 權限才能正常運作。因此、您需要建立一個原則、授予 Trident 所需的權限。

以下範例使用 AWS CLI 建立 IAM 原則：

```
aws iam create-policy --policy-name AmazonFSxNCSIDriverPolicy --policy
-document file://policy.json
  --description "This policy grants access to Trident CSI to FSxN and
Secrets manager"
```

Policy JSON 範例：

```

{
  "Statement": [
    {
      "Action": [
        "fsx:DescribeFileSystems",
        "fsx:DescribeVolumes",
        "fsx:CreateVolume",
        "fsx:RestoreVolumeFromSnapshot",
        "fsx:DescribeStorageVirtualMachines",
        "fsx:UntagResource",
        "fsx:UpdateVolume",
        "fsx:TagResource",
        "fsx>DeleteVolume"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Action": "secretsmanager:GetSecretValue",
      "Effect": "Allow",
      "Resource": "arn:aws:secretsmanager:<aws-region>:<aws-account-id>:secret:<aws-secret-manager-name>*"
    }
  ],
  "Version": "2012-10-17"
}

```

建立 Pod Identity 或 IAM 角色以關聯服務帳戶 (IRSA)

您可以使用 EKS Pod Identity 或 IAM role for Service account association (IRSA) 設定 Kubernetes 服務帳戶來承擔 AWS Identity and Access Management (IAM) 角色。任何已設定為使用該服務帳戶的 Pod 都可以存取該角色有權存取的任何 AWS 服務。

Pod Identity

Amazon EKS Pod Identity 關聯可讓您管理應用程式的憑證，類似於 Amazon EC2 執行個體設定檔向 Amazon EC2 執行個體提供憑證的方式。

在 **EKS 叢集** 上安裝 **Pod Identity**：

您可以透過 AWS 控制台建立 Pod 身分，也可以使用下列 AWS CLI 命令：

```
aws eks create-addon --cluster-name <EKS_CLUSTER_NAME> --addon-name
eks-pod-identity-agent
```

如需詳細資訊，請參閱 ["設定 Amazon EKS Pod Identity Agent"](#)。

建立 **trust-relationship.json**：

建立 trust-relationship.json 檔案，使 EKS Service Principal 能夠承擔 Pod Identity 的此角色。然後使用此信任政策建立角色：

```
aws iam create-role \
  --role-name fsxn-csi-role --assume-role-policy-document file://trust-
relationship.json \
  --description "fsxn csi pod identity role"
```

trust-relationship.json 檔案：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "pods.eks.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:TagSession"
      ]
    }
  ]
}
```

將角色原則附加到 **IAM** 角色：

將上一個步驟中的角色原則附加到已建立的 IAM 角色：

```
aws iam attach-role-policy \  
  --policy-arn arn:aws:iam::aws:111122223333:policy/fsxn-csi-policy \  
  --role-name fsxn-csi-role
```

建立 **Pod** 身分關聯：

在 IAM 角色和 Trident 服務帳戶 (trident-controller) 之間建立 Pod 身分關聯

```
aws eks create-pod-identity-association \  
  --cluster-name <EKS_CLUSTER_NAME> \  
  --role-arn arn:aws:iam::111122223333:role/fsxn-csi-role \  
  --namespace trident --service-account trident-controller
```

服務帳戶關聯 (IRSA) 的 IAM 角色

使用 **AWS CLI**：

```
aws iam create-role --role-name AmazonEKS_FSxN_CSI_DriverRole \  
  --assume-role-policy-document file://trust-relationship.json
```

trust-relationship.json 檔案：

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Principal": {  
        "Federated": "arn:aws:iam::<account_id>:oidc-  
provider/<oidc_provider>"  
      },  
      "Action": "sts:AssumeRoleWithWebIdentity",  
      "Condition": {  
        "StringEquals": {  
          "<oidc_provider>:aud": "sts.amazonaws.com",  
          "<oidc_provider>:sub":  
"system:serviceaccount:trident:trident-controller"  
        }  
      }  
    }  
  ]  
}
```


- EKS 附加元件

如果您想使用快照功能，請安裝 CSI 快照控制器外掛程式。請參閱 "[為 CSI 磁碟區啟用快照功能](#)" 以取得更多資訊。

透過 Helm 安裝 Trident

管理主控台

1. 在 <https://console.aws.amazon.com/eks/home#/clusters> 開啟 Amazon EKS 主控台。
2. 在左側導覽窗格中、選取 **Clusters**。
3. 選擇要為其配置 NetApp Trident CSI 附加元件的叢集名稱。
4. 選擇 **Add-ons**，然後選擇 **Get more add-ons**。
5. 請依照以下步驟選擇附加元件：
 - a. 向下捲動至 **AWS Marketplace** 附加元件 部分，然後在搜尋框中輸入 **"Trident"**。
 - b. 選取 Trident by NetApp 方塊右上角的核取方塊。
 - c. 選擇 **Next**。
6. 在 **Configure selected add-ons** 設定頁面上，執行以下操作：

註 如果您使用的是 **Pod Identity association**，請跳過這些步驟。

- a. 選取您要使用的 **Version**。
- b. 如果您使用 IRSA 身份驗證，請確保設定可選配置設定中提供的配置值：
 - 選取您要使用的 **Version**。
 - 依照 **Add-on configuration schema** 進行操作，並將 **configurationValues** 參數在 **Configuration values** 部分設定為您在上一個步驟建立的 role-arn（值應採用下列格式）：

```
{  
  
  "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'",  
  "cloudProvider": "AWS"  
  
}
```

+

如果選擇「覆蓋」作為衝突解決方式，則現有外掛程式的一個或多個設定可能會被 Amazon EKS 外掛程式的設定覆蓋。如果未啟用此選項且與現有設定有衝突，則操作將會失敗。您可以利用產生的錯誤訊息來排查衝突。選擇此選項之前，請確保 Amazon EKS 外掛程式沒有管理您需要自行管理的設定。

7. 選擇 **Next**。
8. 在 **Review and add** 頁面上，選擇 **Create**。

附加元件安裝完成後，您會看到已安裝的附加元件。

AWS CLI

*1. 建立 `add-on.json` 檔案 *：

對於 **Pod Identity**，請使用以下格式：

註 | 使用

```
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v25.6.0-eksbuild.1",
}
```

對於 **IRSA** 認證、請使用以下格式：

```
{
  "clusterName": "<eks-cluster>",
  "addonName": "netapp_trident-operator",
  "addonVersion": "v25.6.0-eksbuild.1",
  "serviceAccountRoleArn": "<role ARN>",
  "configurationValues": {
    "cloudIdentity": "'eks.amazonaws.com/role-arn: <role ARN>'",
    "cloudProvider": "AWS"
  }
}
```

註 | 將 `<role ARN>` 替換為上一步驟中建立的角色 ARN。

2. 安裝 Trident EKS 附加元件。

```
aws eks create-addon --cli-input-json file://add-on.json
```

eksctl

以下範例命令安裝 Trident EKS 附加元件：

```
eksctl create addon --name netapp_trident-operator --cluster
<cluster_name> --force
```

更新 **Trident EKS** 附加元件

管理主控台

1. 開啟 Amazon EKS 主控台 <https://console.aws.amazon.com/eks/home#/clusters>。
2. 在左側導覽窗格中、選取 **Clusters**。
3. 選擇要為其更新 NetApp Trident CSI 外掛程式的叢集名稱。
4. 選擇 **Add-ons** 標籤。
5. 選取 **Trident by NetApp**，然後選取 **Edit**。
6. 在 **Configure Trident by NetApp** 頁面上、執行以下操作：
 - a. 選取您要使用的 **Version**。
 - b. 展開 **Optional configuration settings** 並根據需要進行修改。
 - c. 選擇 **Save changes**。

AWS CLI

以下範例更新 EKS 附加元件：

```
aws eks update-addon --cluster-name <eks_cluster_name> --addon-name  
netapp_trident-operator --addon-version v25.6.0-eksbuild.1 \  
--service-account-role-arn <role-ARN> --resolve-conflict preserve \  
--configuration-values "{\"cloudIdentity\":  
\"'eks.amazonaws.com/role-arn: <role ARN>'\"}"
```

eksctl

- 檢查您的 FSxN Trident CSI 外掛程式的目前版本。將 `my-cluster` 替換為您的叢集名稱。

```
eksctl get addon --name netapp_trident-operator --cluster my-cluster
```

範例輸出：

NAME	VERSION	STATUS	ISSUES
IAMROLE	UPDATE AVAILABLE	CONFIGURATION VALUES	
netapp_trident-operator	v25.6.0-eksbuild.1	ACTIVE	0
{\"cloudIdentity\": \"'eks.amazonaws.com/role-arn: arn:aws:iam::139763910815:role/AmazonEKS_FSXN_CSI_DriverRole'\"}			

- 將附加元件更新至上一步驟輸出中 UPDATE AVAILABLE 下傳回的版本。

```
eksctl update addon --name netapp_trident-operator --version  
v25.6.0-eksbuild.1 --cluster my-cluster --force
```

如果您移除 `--force` 選項，且任何 Amazon EKS 外掛程式設定與您現有的設定衝突，則更新 Amazon EKS 外掛程式將會失敗；您會收到錯誤訊息，以協助您解決衝突。在指定此選項之前，請確保 Amazon EKS 外掛程式不管理您需要管理的設定，因為這些設定會被此選項覆寫。有關此設定的其他選項的更多資訊，請參閱"[外掛程式](#)"。有關 Amazon EKS Kubernetes 欄位管理的更多資訊，請參閱"[Kubernetes 欄位管理](#)"。

解除安裝 / 移除 Trident EKS 附加元件

您有兩種選項可移除 Amazon EKS 附加元件：

- 保留叢集上的附加軟體 – 此選項將移除 Amazon EKS 對所有設定的管理。它還會移除 Amazon EKS 通知您更新以及在您啟動更新後自動更新 Amazon EKS 附加軟體的功能。但是，它會保留叢集上的附加軟體。此選項使附加軟體成為自我管理安裝，而不是 Amazon EKS 附加軟體。使用此選項，附加軟體不會出現停機時間。在命令中保留 `--preserve` 選項以保留附加軟體。
- 從叢集完全移除附加元件軟體 — NetApp 建議僅在您的叢集上沒有任何依賴該附加元件的資源時，才從叢集移除 Amazon EKS 附加元件。從 `--preserve` 指令中移除 `delete` 選項以移除附加元件。

註 | 如果附加元件關聯了 IAM 帳戶，則不會移除該 IAM 帳戶。

管理主控台

1. 在 <https://console.aws.amazon.com/eks/home#/clusters> 開啟 Amazon EKS 主控台。
2. 在左側導覽窗格中、選取 **Clusters**。
3. 選擇要移除 NetApp Trident CSI 外掛程式的叢集名稱。
4. 選取 **Add-ons** 標籤，然後選取 **Trident by NetApp**。
5. 選擇 **Remove**。
6. 在 **Remove netapp_trident-operator confirmation** 對話方塊中、執行下列操作：
 - a. 如果您希望 Amazon EKS 停止管理外掛程式的設置，請選擇 **Preserve on cluster**。如果您希望將外掛程式軟體保留在叢集上，以便您可以自行管理外掛程式的所有設置，請執行此操作。
 - b. 輸入 **netapp_trident-operator**。
 - c. 選擇 **Remove**。

AWS CLI

將 `my-cluster` 替換為您的叢集名稱，然後執行以下命令。

```
aws eks delete-addon --cluster-name my-cluster --addon-name  
netapp_trident-operator --preserve
```

eksctl

以下命令會解除安裝 Trident EKS 附加元件：

```
eksctl delete addon --cluster K8s-arm --name netapp_trident-operator
```

設定儲存類別

此 "Kubernetes StorageClass 對象"可識別資源配置程式，並指示資源配置程式如何配置磁碟區。本節說明如何設定 Kubernetes StorageClass 物件，以指定 Trident 作為資源配置程式。

建立 StorageClass 物件

當您為 FSx for ONTAP 建立 StorageClass 時，Trident 將自動建立後端組態。

註

如果您想要手動設定儲存後端，請參閱[\[create-a-kubernetes-storageclass-without-automatic-backend-configuration\]](#)章節，瞭解如何分別建立 Trident 後端和儲存類別。

指定所需的 StorageClass 參數

建立 StorageClass 時需要定義以下三個參數：

參數	必填	類型	說明
fsxFilesystemID	是的	字串	FSx for NetApp ONTAP 檔案系統 ID
storageDriverName	是的	字串	Trident 儲存驅動程式（例如、ontap-nas 或 ontap-san）
credentialsName	是的	字串	包含 FSx for ONTAP 憑證的 Kubernetes Secret 名稱

指定選用參數

您可以透過 StorageClass 傳遞可選的後端參數。在 StorageClass parameters 部分中將所有可選值定義為字串。有關後端參數的完整列表、請參閱："FSx for NetApp ONTAP 後端組態"。

範例 StorageClass 設定檔。

以下範例展示 StorageClass 如何觸發自動後端配置。

YAML

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-fsx-demo
  annotations:
    description: "Demo StorageClass for FSx for NetApp ONTAP"
provisioner: csi.trident.netapp.io
parameters:
  fsxFilesystemID: "fs-0abc123"
  storageDriverName: "ontap-nas"
  credentialsName: trident-fsx-credentials
allowVolumeExpansion: true
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

JSON

```
{
  "apiVersion": "storage.k8s.io/v1",
  "kind": "StorageClass",
  "metadata": {
    "name": "ontap-fsx-demo",
    "annotations": {
      "description": "Demo StorageClass for FSx for NetApp ONTAP"
    }
  },
  "provisioner": "csi.trident.netapp.io",
  "parameters": {
    "fsxFilesystemID": "fs-0abc123",
    "storageDriverName": "ontap-nas",
    "credentialsName": "trident-fsx-credentials"
  },
  "allowVolumeExpansion": true,
  "reclaimPolicy": "Delete",
  "volumeBindingMode": "Immediate"
}
```

建立 StorageClass

建立設定檔案後，執行以下命令以建立儲存類別。

```
kubectl create -f storage-class-ontapnas.yaml
```

現在您應該在 Kubernetes 和 Trident 中看到 **basic-csi** 儲存類別，而 Trident 應該已經發現了後端上的儲存池。

```
kubectl get sc basic-csi
```

NAME	PROVISIONER	AGE
basic-csi	csi.trident.netapp.io	15h

套用 StorageClass 後，Trident 會自動建立後端。然後您可以建立 PersistentVolumeClaims 來參照此 StorageClass。

驗證後端組態狀態

Trident 會將後端建立的結果記錄在 StorageClass 註解中。

註解	說明
trident.netapp.io/configuratorStatus	組態結果 (Success 或 Failure)
trident.netapp.io/configuratorMessage	詳細狀態或錯誤訊息
trident.netapp.io/configuratorName	內部配置器資源的名稱
trident.netapp.io/managed	表示 StorageClass 由 Trident 管理
trident.netapp.io/additionalStoragePools	為此後端建立的儲存資源池

若要驗證狀態、請執行：

```
kubectl get storageclass ontap-fsx-demo -o yaml
```

確認 trident.netapp.io/configuratorStatus 是否已設定為 Success。如果值為 Failure，請檢查 trident.netapp.io/configuratorMessage 以瞭解錯誤。

新增其他 FSxN 檔案系統

如果您在繼續使用相同 StorageClass 的情況下需要額外的儲存容量，請新增額外的 FSxN 檔案系統 ID。

編輯 StorageClass 並新增以下註釋：

```
metadata:
  annotations:
    trident.netapp.io/additionalFsxnFileSystemID: '["fs-
xxxxxxxxxxxxxxxxxxxx"]'
```

套用變更後，Trident 會更新後端組態並更新 StorageClass 註釋。

操作考量與限制

- 刪除具有自動後端組態的 StorageClass 通常會刪除相關的 Trident 後端。這可能會中斷儲存連線並導致執行中的工作負載中斷。在刪除託管的 StorageClass 之前，請先驗證其影響。
- 僅 AWS FSx for NetApp ONTAP 支援自動後端組態。

建立一個不帶自動後端配置的 **Kubernetes StorageClass**

如果您想單獨建立 Trident 後端和 StorageClass，請依照下列步驟操作。

了解自動後端組態的運作方式

Trident 從 StorageClass 定義中派生後端配置。當您套用 StorageClass 時，Trident 會驗證必要的參數、建立後端，並使用狀態標記 StorageClass。

Trident 只會建立一次 VolumeSnapshotClass。Trident 會為後續的 StorageClasses 重複使用相同的 VolumeSnapshotClass。

建立 **Trident** 後端

要建立 Trident 後端，您需要建立一個 JSON 或 YAML 格式的設定檔案。該檔案需要指定所需的儲存類型（NAS 或 SAN）、檔案系統、要從中取得資料的 SVM 以及如何進行驗證。以下範例展示如何定義基於 NAS 的儲存，以及如何使用 AWS Secret 來儲存要使用的 SVM 的憑證：

YAML

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  backendName: tbc-ontap-nas
  svm: svm-name
  aws:
    fsxFilesystemID: fs-xxxxxxxxxx
  credentials:
    name: "arn:aws:secretsmanager:us-west-2:xxxxxxx:secret:secret-
name"
    type: awsarn
```

JSON

```
{
  "apiVersion": "trident.netapp.io/v1",
  "kind": "TridentBackendConfig",
  "metadata": {
    "name": "backend-tbc-ontap-nas"
    "namespace": "trident"
  },
  "spec": {
    "version": 1,
    "storageDriverName": "ontap-nas",
    "backendName": "tbc-ontap-nas",
    "svm": "svm-name",
    "aws": {
      "fsxFilesystemID": "fs-xxxxxxxxxx"
    },
    "managementLIF": null,
    "credentials": {
      "name": "arn:aws:secretsmanager:us-west-2:xxxxxxx:secret:secret-
name",
      "type": "awsarn"
    }
  }
}
```

FSx for ONTAP 驅動程式詳細資料

您可以使用下列驅動程式將 Trident 與 Amazon FSx for NetApp ONTAP 整合：

驅動程式名稱	說明
ontap-san	每個已配置的 PV 都是其自身 Amazon FSx for NetApp ONTAP 磁碟區中的一個 LUN。建議用於區塊儲存設備。
ontap-nas	每個已配置的 PV 都是一個完整的 Amazon FSx for NetApp ONTAP 磁碟區。推薦用於 NFS 和 SMB。
ontap-san-economy	每個已配置的 PV 都是一個 LUN，每個 Amazon FSx for NetApp ONTAP 磁碟區可設定 LUN 的數量。
ontap-nas-economy	每個已配置的 PV 都是一個 qtree，每個 Amazon FSx for NetApp ONTAP 磁碟區可配置的 qtree 數量。
ontap-nas-flexgroup	每個已配置的 PV 都是一個完整的 Amazon FSx for NetApp ONTAP FlexGroup 磁碟區。

有關驅動程式詳細資料、請參閱 "[NAS 驅動程式](#)" 和 "[SAN 驅動程式](#)"。

建立後端

建立設定檔案後，執行以下命令來建立和驗證 Trident 後端設定（TBC）：

- 從 yaml 檔案建立 Trident 後端組態（TBC），並執行下列命令：

```
kubectl create -f backendconfig.yaml -n trident
```

```
tridentbackendconfig.trident.netapp.io/backend-tbc-ontap-nas created
```

- 驗證 Trident 後端組態（TBC）是否已成功建立：

```
Kubectl get tbc -n trident
```

NAME	BACKEND NAME	BACKEND UUID
backend-tbc-ontap-nas b9ff-f96d916ac5e9 Bound	tbc-ontap-nas Success	933e0071-66ce-4324-

如需其他組態選項的詳細資訊，請參閱下方的[\[Backend-advanced-configuration-and-examples\]](#)區段。

在*不使用*自動後端配置的情況下配置儲存類別

以下是與 Trident 和 FSx for ONTAP 一起使用的儲存類別設定範例。

NFS 儲存類別

您可以參考此範例來設定使用 NFS 的磁碟區 StorageClass（有關完整的屬性列表，請參閱下面的 Trident 屬性部分）：

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  provisioningType: "thin"
  snapshots: "true"
```

iSCSI 儲存類別

請使用以下範例設定使用 iSCSI 的磁碟區的 StorageClass：

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
  provisioningType: "thin"
  snapshots: "true"
```

使用 NFSv3 和 AWS Bottlerocket 的儲存類別

若要在 AWS Bottlerocket 上配置 NFSv3 磁碟區，請將所需內容新增 `mountOptions` 至儲存類別：

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  media: "ssd"
  provisioningType: "thin"
  snapshots: "true"
mountOptions:
  - nfsvers=3
  - nolock

```

Trident StorageClass 屬性

這些參數決定了應使用哪些 Trident 管理的儲存資源池來配置給定類型的磁碟區。

屬性	類型	價值觀	優惠	要求	支援者
媒體 ¹	字串	HDD、混合式、SSD	Pool 包含此類型的媒體；混合型表示兩者兼具	指定的媒體類型	ontap-nas、ontap-nas-economy、ontap-nas-flexgroup、ontap-san、solidfire-san
provisioningType	字串	薄、厚	資源池支援此佈建方法	已指定佈建方法	thick：所有 ONTAP；thin：所有 ONTAP 和 SolidFire-SAN
backendType	字串	ontap-nas、ontap-nas-economy、ontap-nas-flexgroup、ontap-san、solidfire-san、azure-netapp-files、ontap-san-economy	Pool 屬於這種類型的後端	指定後端	所有驅動程式
快照	布林值	true、false	Pool 支援帶快照的磁碟區	已啟用快照的磁碟區	ontap-nas、ontap-san、solidfire-san
複製	布林值	true、false	儲存池支援複製磁碟區	已啟用複本的磁碟區	ontap-nas、ontap-san、solidfire-san

參數	說明	範例
storageDriverName	儲存驅動程式的名稱	ontap-nas, ontap-nas-economy, ontap-nas-flexgroup, ontap-san, ontap-san-economy
backendName	自訂名稱或儲存後端	驅動程式名稱 + "_" + dataLIF
managementLIF	叢集或 SVM 管理 LIF 的 IP 位址，可以指定完整網域名稱 (FQDN)。如果 Trident 是使用 IPv6 旗標安裝的，則可以設定為使用 IPv6 位址。IPv6 位址必須以方括號定義，例如 [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]。如果您在 fsxFilesystemID 下的 aws 欄位提供了 managementLIF，則不需要再提供 managementLIF，因為 Trident 會從 AWS 擷取 SVM 資訊。因此，您必須為 SVM 下的使用者 (例如：vsadmin) 提供認證，且該使用者必須具有 vsadmin 角色。	"10.0.0.1", "[2001:1234:abcd::fefe]"
dataLIF	協定 LIF 的 IP 位址。 ONTAP NAS 驅動程式：NetApp 建議指定 dataLIF。如果未提供，Trident 將從 SVM 取得 dataLIF。您可以指定一個完全限定網域名稱 (FQDN) 用於 NFS 掛載作業，從而建立輪詢 DNS 以在多個 dataLIF 之間進行負載平衡。初始設定後可以變更。 ONTAP SAN 驅動程式：iSCSI 無需指定。Trident 使用 ONTAP Selective LUN Map 來探索建立多路徑工作階段所需的 iSCSI LIF。如果明確定義了 dataLIF，則會產生警告。如果 Trident 是使用 IPv6 旗標安裝的，則可以設定為使用 IPv6 位址。IPv6 位址必須以方括號定義，例如 [28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]。	
autoExportPolicy	啟用自動匯出原則建立和更新 [布林值]。使用 `autoExportPolicy` 和 `autoExportCIDRs` 選項，Trident 可以自動管理匯出原則。	false
autoExportCIDRs	啟用 `autoExportPolicy` 時用於篩選 Kubernetes 節點 IP 的 CIDR 清單。使用 `autoExportPolicy` 和 `autoExportCIDRs` 選項，Trident 可以自動管理匯出原則。	"["0.0.0.0/0", ":::0]"

參數	說明	範例
labels	要套用於磁碟區的任意 JSON 格式標籤集	""
clientCertificate	用戶端憑證的 Base64 編碼值。用於憑證型驗證	""
clientPrivateKey	用戶端私密金鑰的 Base64 編碼值。用於憑證型驗證	""
trustedCACertificate	受信任 CA 憑證的 Base64 編碼值。此參數為可選。用於憑證型驗證。	""
username	用於連接叢集或 SVM 的使用者名稱。用於基於憑證的身份驗證。例如，vsadmin。	
password	連接叢集或 SVM 的密碼。用於基於憑證的身份驗證。	
svm	要使用的儲存虛擬機器	如果指定了 SVM 管理 LIF，則會衍生。
storagePrefix	在 SVM 中配置新磁碟區時所使用的前綴。建立後無法修改。若要更新此參數、您需要建立新的後端。	trident
limitAggregateUsage	*請勿為 Amazon FSx for NetApp ONTAP 指定。*提供的 `fsxadmin` 和 `vsadmin` 不包含使用 Trident 檢索 Aggregate 使用情況並加以限制所需的權限。	請勿使用。
limitVolumeSize	如果請求的磁碟區大小超過此值，則配置失敗。此外，它還限制了其管理的 qtree 和 LUN 卷的最大大小，並且該 `qtreesPerFlexvol` 選項允許自訂每個 FlexVol 磁碟區的最大 qtree 數量	"（預設不強制執行）
lunsPerFlexvol	每個 FlexVol volume 的最大 LUN 數量必須在 [50, 200] 範圍內。僅限 SAN。	"100"
debugTraceFlags	用於疑難排解的偵錯旗標。例如、{"api":false, "method":true}除非您正在進行疑難排解並需要詳細的記錄傾印、否則請勿使用 debugTraceFlags。	null

參數	說明	範例
nfsMountOptions	以逗號分隔的 NFS 掛載選項清單。Kubernetes 持久性磁碟區的掛載選項通常在儲存類別中指定，但如果儲存類別中未指定任何掛載選項，Trident 將回退到使用儲存後端設定檔中指定的掛載選項。如果儲存類別和設定檔中均未指定掛載選項，Trident 將不會在關聯的持久性磁碟區上設定任何掛載選項。	""
nasType	配置 NFS 或 SMB 磁碟區的建立。選項為 nfs、`smb` 或 null。*必須設定為 `smb` 才能建立 SMB 磁碟區。*設定為 null 則預設建立 NFS 磁碟區。	nfs
qtreesPerFlexvol	每個 FlexVol volume 的最大 Qtree 數量必須在 [50, 300] 範圍內	"200"
smbShare	您可以指定下列名稱之一：使用 Microsoft Management Console 或 ONTAP CLI 建立的 SMB 共用名稱，或允許 Trident 建立 SMB 共用的名稱。此參數對於 Amazon FSx for ONTAP 後端是必要的。	smb-share
useREST	布林參數，用於使用 ONTAP REST API。當設定為 `true` 時，Trident 將使用 ONTAP REST API 與後端通訊。此功能需要 ONTAP 9.11.1 或更新版本。此外，使用的 ONTAP 登入角色必須具有 `ontap` 應用程式的存取權限。預先定義的 `vsadmin` 和 `cluster-admin` 角色可滿足此要求。	false
aws	您可以在 AWS FSx for ONTAP 的組態檔中指定下列項目： - fsxFilesystemID：指定 AWS FSx 檔案系統的 ID。 - apiRegion：AWS API 區域名稱。 - apikey：AWS API 金鑰。 - secretKey：AWS 秘密金鑰。	"" "" ""
credentials	指定要儲存在 AWS Secrets Manager 中的 FSx SVM 認證。 - name：包含 SVM 認證的密碼的 Amazon Resource Name (ARN)。 - type：設定為 awsarn。請參閱 " 建立 AWS Secrets Manager 密碼 " 以取得更多資訊。	

您可以使用 `defaults` 配置部分中的這些選項來控制預設配置。例如、請參閱下面的組態範例。

參數	說明	預設
<code>spaceAllocation</code>	LUN 的空間分配	<code>true</code>
<code>spaceReserve</code>	空間保留模式；「 <code>none</code> 」（精簡）或「 <code>volume</code> 」（完整）	<code>none</code>
<code>snapshotPolicy</code>	要使用的 Snapshot 原則	<code>none</code>
<code>qosPolicy</code>	要為建立的磁碟區指派 QoS 策略群組。每個儲存資源池或後端可選擇 <code>qosPolicy</code> 或 <code>adaptiveQosPolicy</code> 其中之一。搭配 Trident 使用 QoS 策略群組需要 ONTAP 9.8 或更新版本。您應該使用非共享的 QoS 策略群組，並確保該策略群組單獨套用至每個成員。共享的 QoS 策略群組會強制限制所有工作負載的總吞吐量上限。	""
<code>adaptiveQosPolicy</code>	為建立的磁碟區指派的自適應 QoS 原則群組。每個儲存資源池或後端可選擇 <code>qosPolicy</code> 或 <code>adaptiveQosPolicy</code> 其中之一。ontap-nas-economy 不支援。	""
<code>snapshotReserve</code>	為快照保留的磁碟區百分比 "0"	如果 <code>snapshotPolicy</code> 是 <code>none</code> ， <code>else</code> ""
<code>splitOnClone</code>	建立時將複本從其父項分割	<code>false</code>
<code>encryption</code>	在新磁碟區上啟用 NetApp Volume Encryption (NVE)；預設值為 <code>false</code> 。要使用此選項，叢集必須已獲得 NVE 許可並啟用 NVE。如果後端啟用了 NAE，則在 Trident 中佈建的任何磁碟區都會啟用 NAE。如需詳細資訊，請參閱： "Trident 與 NVE 和 NAE 的運作方式" 。	<code>false</code>
<code>luksEncryption</code>	啟用 LUKS 加密。請參閱 "使用 Linux Unified Key Setup (LUKS)" 。僅限 SAN。	""
<code>tieringPolicy</code>	要使用的分層原則 <code>none</code>	
<code>unixPermissions</code>	新磁碟區的模式。 SMB 磁碟區請保留空白。	""
<code>securityStyle</code>	新磁碟區的安全樣式。NFS 支援 <code>mixed</code> 和 <code>unix</code> 安全樣式。SMB 支援 <code>mixed</code> 和 <code>ntfs</code> 安全樣式。	NFS 預設值為 <code>unix</code> 。SMB 預設值為 <code>ntfs</code> 。

配置 PVC

本節包含如何建立 PersistentVolumeClaim (PVC) 使用已設定 Kubernetes StorageClass 來請求 PV 的說明。如果成功，則可以將 PV 掛載到 Pod 中。

建立 PVC

```
https://kubernetes.io/docs/concepts/storage/persistent-volumes["_PersistentVolumeClaim_"] (PVC) 是對叢集上 PersistentVolume 的存取請求。PVC 可以配置為請求特定大小的儲存空間或存取模式。透過關聯的 StorageClass，叢集管理員不僅可以控制 PersistentVolume 大小和存取模式，還可以控制效能或服務等級等更多參數。
```

建立 Trident 後端和 StorageClass 後，即可建立 PVC。建立 PVC 後，即可將該磁碟區掛載到 pod 中。

範例資訊清單

以下範例展示了 PVC 的基本組態選項。

具有 RWX 存取權限的 PVC

此範例顯示了一個與名為 basic-csi 的 StorageClass 關聯的具有 RWX 存取權限的基本 PVC。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-storage
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: ontap-gold
```

使用 iSCSI 的 PVC 範例

此範例展示了一個與名為 `protection-gold` 的 StorageClass 關聯的、具有 RWO 存取權限的 iSCSI 基本 PVC。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-san
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: protection-gold
```

建立 PVC

步驟

1. 建立 PVC。

```
kubectl create -f pvc.yaml
```

2. 核實 PVC 狀態。

```
kubectl get pvc
```

NAME	STATUS	VOLUME	CAPACITY	ACCESS MODES	STORAGECLASS	AGE
pvc-storage	Bound	pv-name	2Gi	RWO		5m

請參閱["Kubernetes 和 Trident 物件"](#)以瞭解儲存類別如何與 `PersistentVolumeClaim` 互動，以及控制 Trident 配置磁碟區的參數詳細資訊。

部署應用程式

建立儲存類別和 PVC 後，即可將 PV 掛載到 Pod 上。本節列出將 PV 連接到 Pod 的範例命令和組態。

部署範例應用程式

步驟

1. 在 pod 中掛載 volume。

```
kubectl create -f pv-pod.yaml
```

以下範例展示了將 PVC 連接到 pod 的基本組態：基本組態：

```
kind: Pod
apiVersion: v1
metadata:
  name: pv-pod
spec:
  volumes:
  - name: pv-storage
    persistentVolumeClaim:
      claimName: basic
  containers:
  - name: pv-container
    image: nginx
    ports:
    - containerPort: 80
      name: "http-server"
    volumeMounts:
    - mountPath: "/my/mount/path"
      name: pv-storage
```

註 | 您可以使用 `kubectl get pod --watch` 監控進度。

2. 確認磁碟區已掛載到 `/my/mount/path`。

```
kubectl exec -it pv-pod -- df -h /my/mount/path
```

```
Filesystem                                Size
Used Avail Use% Mounted on
192.168.188.78:/trident_pvc_ae45ed05_3ace_4e7c_9080_d2a83ae03d06 1.1G
320K 1.0G 1% /my/mount/path
```

現在您可以刪除 Pod 了。Pod 應用程式將不復存在，但卷將保留。

```
kubectl delete pod pv-pod
```

在 EKS 叢集上設定 **Trident EKS** 附加元件

NetApp Trident 簡化了 Kubernetes 中 Amazon FSx for NetApp ONTAP 儲存的管理，讓您的開發人員和管理員能夠專注於應用程式部署。NetApp Trident EKS 外掛程式包含最新的安全性修補程式和錯誤修復，並經過 AWS 驗證，可與 Amazon EKS 搭配使用。EKS 附加

元件可讓您持續確保 Amazon EKS 叢集的安全性和穩定性，並減少安裝、設定和更新附加元件所需的工作量。

先決條件

在為 AWS EKS 設定 Trident 附加元件之前，請確保您已具備以下條件：

- 具有使用附加元件權限的 Amazon EKS 叢集帳戶。請參閱"[Amazon EKS 附加元件](#)"。
- AWS 對 AWS Marketplace 的權限：
"aws-marketplace:ViewSubscriptions",
"aws-marketplace:Subscribe",
"aws-marketplace:Unsubscribe"
- AMI 類型：Amazon Linux 2 (AL2_x86_64) 或 Amazon Linux 2 Arm (AL2_ARM_64)
- 節點類型：AMD 或 ARM
- 現有的 Amazon FSx for NetApp ONTAP 檔案系統

步驟

1. 請務必建立 IAM 角色和 AWS 密鑰，以使 EKS Pod 能夠存取 AWS 資源。如需相關指示，請參閱 "[建立 IAM 角色和 AWS Secret](#)"。
2. 在 EKS Kubernetes 叢集上，導覽至 **Add-ons** 標籤。

The screenshot shows the AWS EKS console interface for a cluster named 'tri-env-eks'. At the top, there are buttons for 'Delete cluster', 'Upgrade version', and 'View dashboard'. A notification banner at the top indicates that standard support for Kubernetes version 1.30 ends on July 28, 2025, with an 'Upgrade now' button. Below this, the 'Cluster info' section displays: Status: Active; Kubernetes version: 1.30; Support period: Standard support until July 28, 2025; Provider: EKS. There are also indicators for 'Cluster health issues' and 'Upgrade insights', both showing 0 issues. A navigation bar below the cluster info shows tabs for Overview, Resources, Compute, Networking, Add-ons (1), Access, Observability, Update history, and Tags. A notification banner below the navigation bar states 'New versions are available for 1 add-on.' The 'Add-ons (3)' section is active, showing a search bar, filters for 'Any category' and 'Any status', and a 'Get more add-ons' button. It indicates there are 3 matches and a page number of 1.

3. 前往 **AWS Marketplace** 外掛程式，然後選擇 *storage* 類別。

AWS Marketplace add-ons (1) ↻

Discover, subscribe to and configure EKS add-ons to enhance your EKS clusters.

Filtering options

Any category ▾ NetApp, Inc. ▾ Any pricing model ▾ [Clear filters](#)

NetApp, Inc. ✕ < 1 >

NetApp **NetApp Trident** ☐

NetApp Trident streamlines Amazon FSx for NetApp ONTAP storage management in Kubernetes to let your developers and administrators focus on application deployment. FSx for ONTAP flexibility, scalability, and integration capabilities make it the ideal choice for organizations seeking efficient containerized storage workflows. [Product details](#)

Standard Contract

Category storage	Listed by NetApp, Inc.	Supported versions 1.31, 1.30, 1.29, 1.28, 1.27, 1.26, 1.25, 1.24, 1.23	Pricing starting at View pricing details
----------------------------	--	---	--

[Cancel](#) [Next](#)

4. 找到 **NetApp Trident**，選取 Trident 外掛程式的複選框，然後按一下 **Next**。

5. 選擇所需的附加元件版本。

Configure selected add-ons settings
Configure the add-ons for your cluster by selecting settings.

NetApp Trident [Remove add-on](#)

Listed by NetApp	Category storage	Status 🟢 Ready to install
----------------------------	---------------------	------------------------------

📘 You're subscribed to this software [View subscription](#) ✕
You can view the terms and pricing details for this product or choose another offer if one is available.

Version
Select the version for this add-on.

▶ **Optional configuration settings**

[Cancel](#) [Previous](#) [Next](#)

6. 配置所需的附加元件設定。

Review and add

Step 1: Select add-ons

Edit

Selected add-ons (1)

Find add-on

Add-on name	Type	Status
netapp_trident-operator	storage	Ready to install

Step 2: Configure selected add-ons settings

Edit

Selected add-ons version (1)

Add-on name	Version	IAM role for service account (IRSA)
netapp_trident-operator	v24.10.0-eksbuild.1	Not set

EKS Pod Identity (0)

Add-on name	IAM role	Service account
No Pod Identity associations None of the selected add-on(s) have Pod Identity associations.		

Cancel

Previous

Create

- 如果您正在使用 IRSA（服務帳戶的 IAM 角色），請參閱其他組態步驟"這裡"。
- 選擇 **Create**。
- 確認附加元件的狀態為 *Active*。

Add-ons (1) Info

View details Edit Remove Get more add-ons

netapp

Any categ... Any status 1 match

NetApp **NetApp Trident**

NetApp Trident streamlines Amazon FSx for NetApp ONTAP storage management in Kubernetes to let your developers and administrators focus on application deployment. FSx for ONTAP flexibility, scalability, and integration capabilities make it the ideal choice for organizations seeking efficient containerized storage workflows. [Product details](#)

Category	Status	Version	EKS Pod Identity	IAM role for service account (IRSA)
storage	Active	v24.10.0-eksbuild.1	-	Not set

Listed by [NetApp, Inc.](#)

View subscription

- 執行以下命令以驗證 Trident 是否已正確安裝在叢集上：

```
kubectl get pods -n trident
```

11. 繼續進行設定並配置儲存後端。如需相關資訊、請參閱 ["配置儲存後端"](#)。

使用 CLI 安裝/解除安裝 Trident EKS 附加元件

使用 CLI 安裝 NetApp Trident EKS 外掛程式：

以下範例指令安裝 Trident EKS 外掛程式：

```
eksctl create addon --cluster clusterName --name netapp_trident-operator  
--version v25.6.0-eksbuild.1 (使用專用版本)
```

以下範例指令安裝 Trident EKS 外掛程式版本 25.6.1：

```
eksctl create addon --cluster clusterName --name netapp_trident-operator  
--version v25.6.1-eksbuild.1 (使用專用版本)
```

以下範例指令安裝 Trident EKS 外掛程式版本 25.6.2：

```
eksctl create addon --cluster clusterName --name netapp_trident-operator  
--version v25.6.2-eksbuild.1 (使用專用版本)
```

使用 CLI 卸載 NetApp Trident EKS 外掛程式：

以下命令會解除安裝 Trident EKS 附加元件：

```
eksctl delete addon --cluster K8s-arm --name netapp_trident-operator
```

使用 kubectl 建立後端

後端定義了 Trident 與儲存系統之間的關係。它告訴 Trident 如何與儲存系統通信，以及 Trident 應該如何從中配置磁碟區。安裝 Trident 後，下一步是建立後端。

`TridentBackendConfig` 自訂資源定義 (CRD) 可讓您直接透過 Kubernetes 介面建立和管理 Trident 後端。您可以使用 `kubectl` 或適用於您的 Kubernetes 發行版的等效 CLI 工具來完成此操作。

TridentBackendConfig

TridentBackendConfig (tbc tbconfig tbackendconfig) 是一個前端命名空間 CRD，可讓您使用 kubectl 管理 Trident 後端。Kubernetes 和儲存管理員現在可以直接透過 Kubernetes CLI 建立和管理後端，而無需專用的命令列實用程式 ((tridentctl))。

建立 TridentBackendConfig 物件時，會發生以下情況：

- Trident 根據您提供的設定自動建立後端。這在內部表示為 TridentBackend (tbc tridentbackend) CR。
- TridentBackendConfig 唯一綁定到由 Trident 建立的 TridentBackend。

每個 `TridentBackendConfig` 都與 `TridentBackend` 保持一對一的映射關係。前者是提供給使用者設計和配置後端的介面；後者是 Trident 表示實際後端物件的方式。

警告

TridentBackend CR 由 Trident 自動建立。您*不應該*修改它們。如果您想要更新後端、請透過修改 `TridentBackendConfig` 物件來執行此作業。

請參閱以下 TridentBackendConfig CR 格式範例：

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret
```

您也可以查看 "[trident-installer](#)" 目錄中的範例，以了解所需儲存平台/服務的範例配置。

`spec` 接受後端特定的組態參數。在本例中、後端使用 `ontap-san` 儲存驅動程式、並使用此處表格中列出的組態參數。如需所需儲存驅動程式的組態選項清單、請參閱 [link:backends.html\["儲存驅動程式的後端組態資訊"\]](#)。

`spec` 區段也包含 `credentials` 和 `deletionPolicy` 欄位，這些欄位是在 `TridentBackendConfig` CR 中新引入的：

- `credentials`：此參數為必填欄位，包含用於向儲存系統/服務進行驗證的認證資料。此參數設定為使用者建立的 Kubernetes Secret。認證資料不能以純文字形式傳遞，否則將導致錯誤。
- `deletionPolicy`：此欄位定義 TridentBackendConfig 刪除時應執行的操作。它可以取以下兩個值之一：
 - `delete`：這將導致 TridentBackendConfig CR 及其關聯的後端都被刪除。這是預設值。
 - `retain`：當 TridentBackendConfig CR 被刪除時，後端定義仍會存在，並可透過 `tridentctl` 進行管理。將刪除原則設為 `retain` 可讓使用者降級至較早版本（21.04 之前版本）並保留已建立的後端。建立 `TridentBackendConfig` 之後，可以更新此欄位的值。

註

後端名稱透過 `spec.backendName` 設定。如果未指定，後端名稱將設定為 TridentBackendConfig 物件的名稱（`metadata.name`）。建議使用 `spec.backendName` 明確設定後端名稱。

提示

使用 `tridentctl` 建立的後端沒有關聯的 `TridentBackendConfig` 物件。您可以選擇透過建立 `TridentBackendConfig` CR，使用 `kubectl` 來管理這些後端。必須注意指定相同的組態參數（例如 `spec.backendName`、`spec.storagePrefix`、`spec.storageDriverName` 等）。Trident 會自動將新建立的 `TridentBackendConfig` 與現有的後端綁定。

步驟概述

要使用 `kubectl` 建立新的後端，您應該執行以下操作：

1. 建立 "Kubernetes Secret"。此密鑰包含 Trident 與儲存叢集/服務通訊所需的認證資料。
2. 建立 `TridentBackendConfig` 物件。其中包含儲存叢集 / 服務的詳細資訊、以及在上一步中建立的機密參照。

建立後端後，您可以使用 `kubectl get tbc <tbc-name> -n <trident-namespace>` 來觀察其狀態並收集更多詳細資訊。

步驟 1：建立 Kubernetes Secret

建立一個包含後端存取認證的 Secret。每個儲存服務 / 平台都是唯一的。以下是一個範例：

```
kubectl -n trident create -f backend-tbc-ontap-san-secret.yaml
```

```
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-san-secret
type: Opaque
stringData:
  username: cluster-admin
  password: password
```

下表總結了每個儲存平台 Secret 中必須包含的欄位：

儲存平台 Secret Fields 描述	機密	欄位說明
Azure NetApp Files	clientID	應用程式註冊中的用戶端 ID
Element (NetApp HCI/SolidFire)	端點	具有租戶認證資料的 SolidFire 叢集的 MVIP
ONTAP	使用者名稱	用於連接叢集 / SVM 的使用者名稱。用於基於憑證的身份驗證

儲存平台 Secret Fields 描述	機密	欄位說明
ONTAP	密碼	連接到叢集 / SVM 的密碼。用於基於憑證的身份驗證
ONTAP	clientPrivateKey	用戶端私密金鑰的 Base64 編碼值。用於基於憑證的驗證。
ONTAP	chapUsername	入站使用者名稱。如果 useCHAP=true 則為必填項。適用於 ontap-san 和 `ontap-san-economy`
ONTAP	chapInitiatorSecret	CHAP 啟動器密碼。如果 useCHAP=true 則為必填項。適用於 ontap-san 和 `ontap-san-economy`
ONTAP	chapTargetUsername	目標使用者名稱。如果 useCHAP=true 則為必填項。適用於 ontap-san 和 `ontap-san-economy`
ONTAP	chapTargetInitiatorSecret	CHAP 目標啟動器密碼。如果 useCHAP=true 則為必填項。適用於 ontap-san 和 `ontap-san-economy`

在此步驟中建立的 Secret，將會在下一步建立的 TridentBackendConfig 物件的 spec.credentials 欄位中被引用。

步驟 2：建立 TridentBackendConfig CR

您現在可以建立 TridentBackendConfig CR 了。在本例中，使用 ontap-san 驅動程式的後端是使用以下所示的 TridentBackendConfig 物件建立的：

```
kubectl -n trident create -f backend-tbc-ontap-san.yaml
```

```

apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-san
spec:
  version: 1
  backendName: ontap-san-backend
  storageDriverName: ontap-san
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  svm: trident_svm
  credentials:
    name: backend-tbc-ontap-san-secret

```

步驟 3：驗證 TridentBackendConfig CR 的狀態

建立 TridentBackendConfig CR 後、您可以驗證其狀態。請參閱以下範例：

```

kubectl -n trident get tbc backend-tbc-ontap-san

```

NAME	PHASE	STATUS	BACKEND NAME	BACKEND UUID
backend-tbc-ontap-san		Bound	ontap-san-backend	8d24fce7-6f60-4d4a-8ef6-bab2699e6ab8
			Success	

後端已成功建立並綁定到 TridentBackendConfig CR。

階段可以採用下列其中一個值：

- **Bound**：TridentBackendConfig CR 與後端關聯，且該後端包含 configRef 設定為 TridentBackendConfig CR 的 uid 的值。
- **Unbound**：使用 "" 表示。TridentBackendConfig 物件未綁定到後端。所有新建的 TridentBackendConfig CR 預設都處於此階段。階段變更後，無法再恢復為 Unbound 狀態。
- **Deleting**：TridentBackendConfig CR 的 deletionPolicy 被設定為刪除。當 TridentBackendConfig CR 被刪除時，它會進入「正在刪除」狀態。
 - 如果後端不存在持久性磁碟區宣告 (PVC)，則刪除 TridentBackendConfig 將導致 Trident 刪除後端以及 TridentBackendConfig CR。
 - 如果後端存在一個或多個 PVC，則後端進入刪除狀態。TridentBackendConfig CR 隨後也進入刪除階段。後端和 TridentBackendConfig 只有在所有 PVC 都被刪除後才會刪除。
- **Lost**：與 TridentBackendConfig CR 關聯的後端被意外或故意刪除，而 TridentBackendConfig CR 仍然保留對已刪除後端的參考。無論 TridentBackendConfig 值為何，deletionPolicy CR 仍然可以被刪除。
- **Unknown**：Trident 無法確定與 TridentBackendConfig CR 關聯的後端的狀態或是否存在。例如，如果 API 伺服器沒有回應或 tridentbackends.trident.netapp.io CRD 缺失。則可能需要介入。

至此，後端已成功創建！還可以處理一些其他操作，例如 ["後端更新和後端刪除"](#)。

(選用) 步驟 4：取得更多詳細資料

您可以執行以下命令來獲取有關後端的更多資訊：

```
kubectl -n trident get tbc backend-tbc-ontap-san -o wide
```

NAME	BACKEND NAME	BACKEND UUID
PHASE STATUS STORAGE DRIVER DELETION POLICY		
backend-tbc-ontap-san	ontap-san-backend	8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8	Bound Success ontap-san	delete

此外、您還可以獲得 YAML/JSON 轉儲 TridentBackendConfig。

```
kubectl -n trident get tbc backend-tbc-ontap-san -o yaml
```

```

apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  creationTimestamp: 2021-04-21T20:45:11Z
  finalizers:
    - trident.netapp.io
  generation: 1
  name: backend-tbc-ontap-san
  namespace: trident
  resourceVersion: "947143"
  uid: 35b9d777-109f-43d5-8077-c74a4559d09c
spec:
  backendName: ontap-san-backend
  credentials:
    name: backend-tbc-ontap-san-secret
  managementLIF: 10.0.0.1
  dataLIF: 10.0.0.2
  storageDriverName: ontap-san
  svm: trident_svm
  version: 1
status:
  backendInfo:
    backendName: ontap-san-backend
    backendUUID: 8d24fce7-6f60-4d4a-8ef6-bab2699e6ab8
  deletionPolicy: delete
  lastOperationStatus: Success
  message: Backend 'ontap-san-backend' created
  phase: Bound

```

backendInfo 包含 backendName 和 backendUUID 回應 TridentBackendConfig CR 而建立的後端。lastOperationStatus 欄位表示 TridentBackendConfig CR 最後一次操作的狀態，該操作可以由使用者觸發（例如，使用者變更了 spec 中的某些內容）或由 Trident 觸發（例如，在 Trident 重新啟動期間）。其值可以是 Success 或 Failed。phase 表示 TridentBackendConfig CR 與後端之間關係的狀態。在上面的範例中，phase 的值為 Bound，這意味著 TridentBackendConfig CR 已與後端關聯。

您可以執行 `kubectl -n trident describe tbc <tbc-cr-name>` 命令來取得事件日誌的詳細資訊。

警告

您無法使用 `TridentBackendConfig` 物件透過 `tridentctl` 來更新或刪除包含關聯的後端。若要了解在 `tridentctl` 與 `TridentBackendConfig` 之間切換所需的步驟，請["請看這裡"](#)。

管理後端

使用 **kubectl** 執行後端管理

了解如何使用 `kubectl` 執行後端管理操作。

刪除後端

刪除 `TridentBackendConfig` 會指示 Trident 刪除/保留後端（取決於 `deletionPolicy`）。若要刪除後端，請確保 `deletionPolicy` 設定為刪除。若要僅刪除 `TridentBackendConfig`，請確保 `deletionPolicy` 設定為保留。這樣可以確保後端仍然存在，並且可以使用 `tridentctl` 進行管理。

執行下列命令：

```
kubectl delete tbc <tbc-name> -n trident
```

Trident 不會刪除 Kubernetes 正在使用的 Secret `TridentBackendConfig`。Kubernetes 用戶負責清理 Secret。刪除 Secret 時必須格外小心。只有當 Secret 不再被後端使用時，才應刪除。

檢視現有的後端

執行下列命令：

```
kubectl get tbc -n trident
```

您也可以執行 `tridentctl get backend -n trident` 或 `tridentctl get backend -o yaml -n trident` 來取得所有現有後端的清單。此清單還將包括使用 `tridentctl` 建立的後端。

更新後端

更新後端的原因可能有很多：

- 儲存系統的憑證已更改。若要更新憑證，必須更新 `TridentBackendConfig` 物件中使用的 Kubernetes Secret。Trident 將使用提供的最新憑證自動更新後端。執行以下命令更新 Kubernetes Secret:

```
kubectl apply -f <updated-secret-file.yaml> -n trident
```

- 需要更新參數（例如正在使用的 ONTAP SVM 的名稱）。
 - 您可以使用以下命令直接透過 Kubernetes 更新 `TridentBackendConfig` 物件：

```
kubectl apply -f <updated-backend-file.yaml>
```

- 或者，您可以使用以下命令對現有 `TridentBackendConfig` CR 進行更改：

```
kubectl edit tbc <tbc-name> -n trident
```

註

- 如果後端更新失敗，後端將繼續保持其上次已知的配置。您可以透過執行 `kubectl get tbc <tbc-name> -o yaml -n trident` 或 `kubectl describe tbc <tbc-name> -n trident` 來查看日誌以確定原因。
- 在您識別並修正組態檔的問題後，您可以重新執行更新命令。

使用 **tridentctl** 執行後端管理

了解如何使用 `tridentctl` 執行後端管理操作。

建立後端

建立 "後端組態檔" 後，執行以下命令：

```
tridentctl create backend -f <backend-file> -n trident
```

如果後端建立失敗，則表示後端配置存在問題。您可以執行以下命令查看日誌以確定原因：

```
tridentctl logs -n trident
```

在您發現並修正設定檔中的問題後，您可以再次執行 `create` 命令。

刪除後端

若要從 Trident 刪除後端、請執行下列操作：

1. 取得後端名稱：

```
tridentctl get backend -n trident
```

2. 刪除後端：

```
tridentctl delete backend <backend-name> -n trident
```

註

如果 Trident 已從此後端配置了仍然存在的磁碟區和快照，則刪除該後端將阻止其配置新的磁碟區。該後端將繼續以「正在刪除」狀態存在。

檢視現有的後端

若要檢視 Trident 已知的後端、請執行下列步驟：

- 若要取得摘要、請執行下列命令：

```
tridentctl get backend -n trident
```

- 若要取得所有詳細資料、請執行下列命令：

```
tridentctl get backend -o json -n trident
```

更新後端

建立新的後端組態檔後，執行以下命令：

```
tridentctl update backend <backend-name> -f <backend-file> -n trident
```

如果後端更新失敗，則表示後端配置存在問題，或者您嘗試了無效的更新操作。您可以執行以下命令查看日誌以確定原因：

```
tridentctl logs -n trident
```

在您發現並修正設定檔中的問題後，您可以再次執行 `update` 命令。

識別使用後端的儲存類別

這是一個可以使用 `tridentctl` 為後端物件輸出的 JSON 資料來回答的問題範例。這需要用到 `jq` 實用程式，您需要先安裝該程式。

```
tridentctl get backend -o json | jq '[.items[] | {backend: .name, storageClasses: [.storage[].storageClasses]|unique}]'
```

這也適用於使用 `TridentBackendConfig` 建立的後端。

在後端管理選項之間移動

了解在 Trident 中管理後端的不同方式。

管理後端的選項

隨著 `TridentBackendConfig` 的推出，管理員現在有兩種獨特的後端管理方式。這引發了以下問題：

- 使用 `tridentctl` 建立的後端可以用 `TridentBackendConfig` 來管理嗎？
- 使用 `TridentBackendConfig` 建立的後端可以使用 `tridentctl` 來管理嗎？

使用 `tridentctl` 後端管理 `TridentBackendConfig`

本節說明如何通過 Kubernetes 介面直接建立 `tridentctl` 並建立 `TridentBackendConfig` 物件來管理後端的步驟。

這適用於以下情況：

- 已存在的後端，沒有 `TridentBackendConfig`，因為它們是用 `tridentctl` 建立的。
- 使用 `tridentctl` 建立的新後端，同時存在其他 `TridentBackendConfig` 物件。

無論哪種情況，後端都將繼續存在，Trident 負責排程磁碟區並對其進行操作。管理員此時有兩種選擇：

- 繼續使用 `tridentctl` 來管理使用它建立的後端。
- 將使用 `tridentctl` 建立的後端繫結至新的 `TridentBackendConfig` 物件。這樣做意味著後端將使用 `kubectl` 而非 `tridentctl` 進行管理。

若要使用 `kubectl` 管理現有後端，您需要建立一個 `TridentBackendConfig` 綁定到現有後端。以下概述了其工作原理：

1. 建立 Kubernetes Secret。此 Secret 包含 Trident 與儲存叢集/服務通訊所需的認證資料。
2. 建立 `TridentBackendConfig` 物件。其中包含儲存叢集 / 服務的詳細資訊、以及在上一步中建立的機密參照。必須小心指定相同的組態參數（例如 `spec.backendName`、`spec.storagePrefix`、`spec.storageDriverName` 等）。`spec.backendName` 必須設定為現有後端的名稱。

步驟 0：識別後端

若要建立 `TridentBackendConfig` 綁定到現有後端，您需要取得後端配置。在本例中，我們假設後端是使用以下 JSON 定義建立的：

```
tridentctl get backend ontap-nas-backend -n trident
+-----+-----+
+-----+-----+-----+-----+
|          NAME          | STORAGE DRIVER |          UUID          |
| STATE  | VOLUMES  |          |          |
+-----+-----+-----+-----+
| ontap-nas-backend    | ontap-nas      | 52f2eb10-e4c6-4160-99fc- |
| 96b3be5ab5d7 | online |          25 |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```

```
cat ontap-nas-backend.json
```

```

{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "10.10.10.1",
  "dataLIF": "10.10.10.2",
  "backendName": "ontap-nas-backend",
  "svm": "trident_svm",
  "username": "cluster-admin",
  "password": "admin-password",
  "defaults": {
    "spaceReserve": "none",
    "encryption": "false"
  },
  "labels": {
    "store": "nas_store"
  },
  "region": "us_east_1",
  "storage": [
    {
      "labels": {
        "app": "msoffice",
        "cost": "100"
      },
      "zone": "us_east_1a",
      "defaults": {
        "spaceReserve": "volume",
        "encryption": "true",
        "unixPermissions": "0755"
      }
    },
    {
      "labels": {
        "app": "mysqldb",
        "cost": "25"
      },
      "zone": "us_east_1d",
      "defaults": {
        "spaceReserve": "volume",
        "encryption": "false",
        "unixPermissions": "0775"
      }
    }
  ]
}

```

步驟 1：建立 Kubernetes Secret

建立一個包含後端憑證的 Secret，如下例所示：

```
cat tbc-ontap-nas-backend-secret.yaml
```

```
apiVersion: v1
kind: Secret
metadata:
  name: ontap-nas-backend-secret
type: Opaque
stringData:
  username: cluster-admin
  password: admin-password
```

```
kubectl create -f tbc-ontap-nas-backend-secret.yaml -n trident
secret/backend-tbc-ontap-san-secret created
```

步驟 2：建立 TridentBackendConfig CR

下一步是建立 TridentBackendConfig CR，使其自動綁定到現有 ontap-nas-backend（如本例所示）。請確保滿足以下要求：

- 相同的後端名稱已在 `spec.backendName` 中定義。
- 配置參數與原始後端相同。
- 虛擬資源池（如果存在）必須保持與原始後端相同的順序。
- 認證資訊透過 Kubernetes Secret 提供，而非以純文字形式提供。

在這種情況下，`TridentBackendConfig` 看起來會像這樣：

```
cat backend-tbc-ontap-nas.yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-ontap-nas-backend
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: 10.10.10.1
  dataLIF: 10.10.10.2
  backendName: ontap-nas-backend
  svm: trident_svm
  credentials:
    name: mysecret
  defaults:
    spaceReserve: none
    encryption: 'false'
  labels:
    store: nas_store
    region: us_east_1
  storage:
  - labels:
      app: msoffice
      cost: '100'
      zone: us_east_1a
      defaults:
        spaceReserve: volume
        encryption: 'true'
        unixPermissions: '0755'
  - labels:
      app: mysqlldb
      cost: '25'
      zone: us_east_1d
      defaults:
        spaceReserve: volume
        encryption: 'false'
        unixPermissions: '0775'
```

```
kubectl create -f backend-tbc-ontap-nas.yaml -n trident
tridentbackendconfig.trident.netapp.io/tbc-ontap-nas-backend created
```

步驟 3：驗證 TridentBackendConfig CR 的狀態

在 TridentBackendConfig 建立之後，其階段必須為 Bound。它還應反映與現有後端相同的後端名稱和 UUID。

```

kubect1 get tbc tbc-ontap-nas-backend -n trident
NAME                                BACKEND NAME                BACKEND UUID
PHASE    STATUS
tbc-ontap-nas-backend  ontap-nas-backend          52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7    Bound    Success

#confirm that no new backends were created (i.e., TridentBackendConfig did
not end up creating a new backend)
tridentctl get backend -n trident
+-----+-----+
+-----+-----+-----+-----+
|          NAME          | STORAGE DRIVER |          UUID
| STATE  | VOLUMES |
+-----+-----+-----+-----+
| ontap-nas-backend    | ontap-nas      | 52f2eb10-e4c6-4160-99fc-
96b3be5ab5d7 | online |          25 |
+-----+-----+-----+-----+
+-----+-----+-----+-----+

```

現在後端將完全使用 tbc-ontap-nas-backend TridentBackendConfig 物件進行管理。

使用 TridentBackendConfig 後端管理 `tridentctl`

`tridentctl` 可用於列出使用 `TridentBackendConfig` 建立的後端。此外，管理員也可以選擇透過 `tridentctl` 完全管理這些後端，方法是刪除 `TridentBackendConfig` 並確保 `spec.deletionPolicy` 設定為 `retain`。

步驟 0：識別後端

例如，假設使用 TridentBackendConfig 建立了以下後端：

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                                BACKEND NAME          BACKEND UUID
PHASE  STATUS  STORAGE DRIVER  DELETION POLICY
backend-tbc-ontap-san  ontap-san-backend    81abcb27-ea63-49bb-b606-
0a5315ac5f82  Bound  Success  ontap-san          delete

tridentctl get backend ontap-san-backend -n trident
+-----+-----+
+-----+-----+-----+-----+-----+
|          NAME          | STORAGE DRIVER |                               UUID
| STATE  | VOLUMES |
+-----+-----+-----+-----+-----+
| ontap-san-backend | ontap-san      | 81abcb27-ea63-49bb-b606-
0a5315ac5f82 | online |          33 |
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
```

從輸出結果可以看出，`TridentBackendConfig` 已成功創建，並綁定到後端（觀察後端的 UUID）。

步驟 1：確認 `deletionPolicy` 已設定為 `retain`

讓我們來看一下 `deletionPolicy` 的值。這需要設定為 `retain`。這可確保刪除 `TridentBackendConfig` CR 時，後端定義仍會存在，並可使用 `tridentctl` 進行管理。

```
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                                BACKEND NAME          BACKEND UUID
PHASE  STATUS  STORAGE DRIVER  DELETION POLICY
backend-tbc-ontap-san  ontap-san-backend    81abcb27-ea63-49bb-b606-
0a5315ac5f82  Bound  Success  ontap-san          delete

# Patch value of deletionPolicy to retain
kubectl patch tbc backend-tbc-ontap-san --type=merge -p
'{"spec":{"deletionPolicy":"retain"}}' -n trident
tridentbackendconfig.trident.netapp.io/backend-tbc-ontap-san patched

#Confirm the value of deletionPolicy
kubectl get tbc backend-tbc-ontap-san -n trident -o wide
NAME                                BACKEND NAME          BACKEND UUID
PHASE  STATUS  STORAGE DRIVER  DELETION POLICY
backend-tbc-ontap-san  ontap-san-backend    81abcb27-ea63-49bb-b606-
0a5315ac5f82  Bound  Success  ontap-san          retain
```

註 | 除非 `deletionPolicy` 設定為 `retain`，否則不要進行下一步。

步驟 2：刪除 TridentBackendConfig CR

最後一步是刪除 TridentBackendConfig CR。確認 `deletionPolicy` 設定為 `retain` 後，即可執行刪除操作：

```
kubectl delete tbc backend-tbc-ontap-san -n trident
tridentbackendconfig.trident.netapp.io "backend-tbc-ontap-san" deleted

tridentctl get backend ontap-san-backend -n trident
+-----+-----+
+-----+-----+-----+-----+
|          NAME          | STORAGE DRIVER |          UUID          |
| STATE  | VOLUMES |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
| ontap-san-backend | ontap-san      | 81abcb27-ea63-49bb-b606-
0a5315ac5f82 | online |          33 |
+-----+-----+-----+-----+
+-----+-----+-----+-----+
```

刪除 TridentBackendConfig 物件時，Trident 只是將其移除，而不會實際刪除後端本身。

建立和管理儲存類別

建立儲存類別

配置 Kubernetes StorageClass 物件並建立儲存類別，以指示 Trident 如何配置磁碟區。

配置 Kubernetes StorageClass 物件

此 "[Kubernetes StorageClass 對象](#)" 識別 Trident 為該類別使用的佈建程式，並指示 Trident 如何佈建磁碟區。例如：

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-gold
provisioner: csi.trident.netapp.io
mountOptions:
  - nfsvers=3
  - nolock
parameters:
  backendType: "ontap-nas"
  media: "ssd"
allowVolumeExpansion: true
volumeBindingMode: Immediate
```

請參閱"[Kubernetes 和 Trident 物件](#)"以瞭解儲存類別如何與 `PersistentVolumeClaim` 互動，以及控制 Trident 配置磁碟區的參數詳細資訊。

建立儲存類別

建立 StorageClass 物件後，即可建立儲存類別。[\[儲存類別範例\]](#) 提供了一些可供使用或修改的基本範例。

步驟

1. 這是一個 Kubernetes 物件，因此請使用 `kubectl` 在 Kubernetes 中建立它。

```
kubectl create -f sample-input/storage-class-basic-csi.yaml
```

2. 現在您應該在 Kubernetes 和 Trident 中看到 **basic-csi** 儲存類別，而 Trident 應該已經發現了後端上的儲存池。

```
kubectl get sc basic-csi
```

NAME	PROVISIONER	AGE
basic-csi	csi.trident.netapp.io	15h

```
./tridentctl -n trident get storageclass basic-csi -o json
```

```

{
  "items": [
    {
      "Config": {
        "version": "1",
        "name": "basic-csi",
        "attributes": {
          "backendType": "ontap-nas"
        },
        "storagePools": null,
        "additionalStoragePools": null
      },
      "storage": {
        "ontapnas_10.0.0.1": [
          "aggr1",
          "aggr2",
          "aggr3",
          "aggr4"
        ]
      }
    }
  ]
}

```

儲存類別範例

Trident 提供 "特定後端的簡單儲存類別定義"。

或者、您可以編輯安裝程式隨附的 `sample-input/storage-class-csi.yaml.template` 檔案、並將 `BACKEND_TYPE` 替換為儲存驅動程式名稱。


```
"nameTemplate": "pool_{{ .config.StoragePrefix }}_{{ .volume.Name }}_{{ slice .volume.RequestName 1 5 }}"
```

需要考慮的要點

1. 對於磁碟區匯入，僅當現有磁碟區的標籤採用特定格式時，才會更新標籤。例如：
`{"provisioning":{"Cluster":"ClusterA", "PVC": "pvcname"}}`
2. 對於託管磁碟區匯入，磁碟區名稱遵循後端定義中根層級定義的名稱範本。
3. Trident 不支援將切片運算子與儲存前置字元一起使用。
4. 如果範本無法產生唯一的磁碟區名稱，Trident 將附加一些隨機字元以建立唯一的磁碟區名稱。
5. 如果 NAS 經濟型磁碟區的自訂名稱長度超過 64 個字元，Trident 將依照現有的命名規則命名磁碟區。對於所有其他 ONTAP 驅動程式，如果磁碟區名稱超過名稱限制，則磁碟區建立程序將會失敗。

跨命名空間共享 NFS Volume

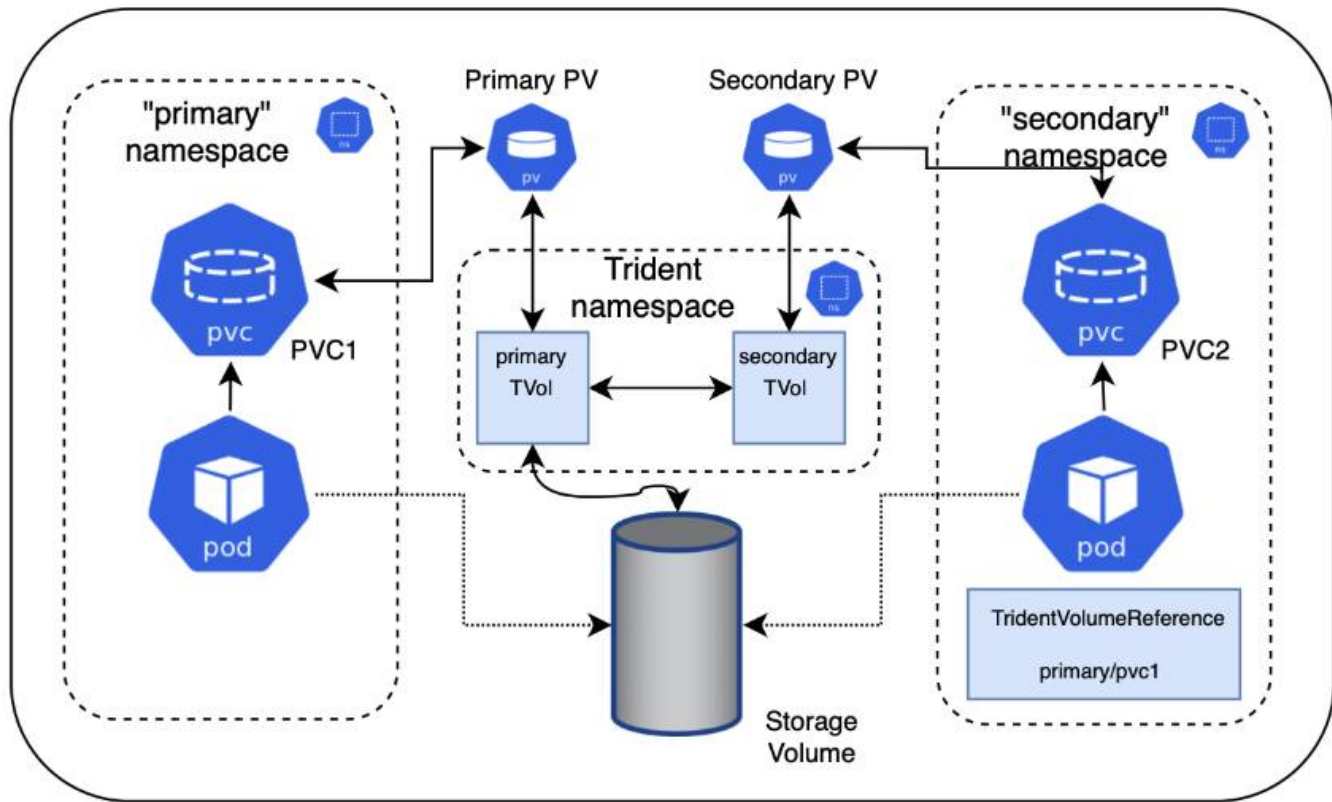
使用 Trident、您可以在主命名空間中建立磁碟區、並在一個或多個次要命名空間中共用該磁碟區。

功能

TridentVolumeReference CR 可讓您在一個或多個 Kubernetes 命名空間之間安全地共用 ReadWriteMany (RWX) NFS 磁碟區。這種 Kubernetes 原生解決方案具有以下優勢：

- 多層級存取控制以確保安全性
- 適用於所有 Trident NFS Volume 驅動程式
- 不依賴 tridentctl 或任何其他非原生 Kubernetes 功能

此圖展示了跨兩個 Kubernetes 命名空間的 NFS 磁碟區共用。



快速入門

只需幾個步驟即可設定 NFS 磁碟區共用。

1

配置來源 **PVC** 以共用磁碟區

來源命名空間擁有者授予存取來源 PVC 中資料的權限。

2

授予在目標命名空間中建立 **CR** 的權限

叢集管理員授予目標命名空間的擁有者建立 TridentVolumeReference CR 的權限。

3

在目標命名空間中建立 **TridentVolumeReference**

目標命名空間的擁有者建立 TridentVolumeReference CR 以引用來源 PVC。

4

在目標命名空間中建立從屬 **PVC**

目標命名空間的擁有者建立從屬 PVC，以使用來源 PVC 中的資料來源。

設定來源和目的地命名空間

為確保安全，跨命名空間共用需要來源命名空間擁有者、叢集管理員和目標命名空間擁有者的協作和操作。每個步驟都會指定使用者角色。

步驟

1. 來源命名空間擁有者：在來源命名空間中建立 PVC(pvc1，該 PVC 授予與目標命名空間(namespace2) 共享的權限，使用 `shareToNamespace` 註解。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1
  namespace: namespace1
  annotations:
    trident.netapp.io/shareToNamespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

Trident 會建立 PV 及其後端 NFS 儲存磁碟區。

註

- 您可以使用逗號分隔的清單將 PVC 共用給多個命名空間。例如，
trident.netapp.io/shareToNamespace: namespace2, namespace3, namespace4 ◦
- 您可以使用 * 共用到所有命名空間。例如，
trident.netapp.io/shareToNamespace: *
- 您可以隨時更新 PVC 以包含 `shareToNamespace` 註釋。

2. 叢集管理員：確保已部署適當的基於角色的存取控制 (RBAC)，以授予目標命名空間擁有者在目標命名空間中建立 TridentVolumeReference CR 的權限。
3. 目標命名空間擁有者：在目標命名空間中建立一個 TridentVolumeReference CR，引用來源命名空間 pvc1。

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. 目標命名空間擁有者：在目標命名空間 (namespace2) 中建立一個 PVC (pvc2)，並使用 shareFromPVC 註解來指定來源 PVC。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  annotations:
    trident.netapp.io/shareFromPVC: namespace1/pvc1
  name: pvc2
  namespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

註 目的地 PVC 的大小必須小於或等於來源 PVC。

結果

Trident 讀取 `shareFromPVC` 目標 PVC 上的註解，並將目標 PV 建立為從屬磁碟區，該磁碟區本身沒有儲存資源，指向來源 PV 並共用來源 PV 的儲存資源。目標 PVC 和 PV 看起來就像正常綁定一樣。

刪除共享磁碟區

您可以刪除跨多個命名空間共享的磁碟區。Trident 將移除來源命名空間對該磁碟區的存取權限，並保留其他共用該磁碟區的命名空間的存取權限。當所有引用該磁碟區的命名空間都被移除後，Trident 才會刪除該磁碟區。

使用 `tridentctl get` 查詢從屬磁碟區

使用 [tridentctl 公用程式](tridentctl)，您可以執行 get 命令來取得從屬磁碟區。如需詳細資訊、請參閱 [tridentctl 命令和選項](#)。

```
Usage:
  tridentctl get [option]
```

標記：

- `-h`, `--help`：有關磁碟區的說明。
- `--parentOfSubordinate string`：將查詢限制在從屬來源磁碟區。
- `--subordinateOf string`：將查詢限制在 Volume 的下級。

限制

- Trident 無法阻止目的地命名空間寫入共用磁碟區。您應該使用檔案鎖定或其他程序來防止覆寫共用磁碟區資料。

- 您無法透過移除 `shareToNamespace` 或 `shareFromNamespace` 註解或刪除 `TridentVolumeReference` CR 來撤銷對來源 PVC 的存取權限。若要撤銷存取權限，您必須刪除從屬 PVC。
- 從屬磁碟區無法進行 Snapshot、複本和鏡射操作。

如需更多資訊

若要深入瞭解跨命名空間磁碟區存取：

- 觀看 "[NetAppTV](#)" 上的示範影片。

跨命名空間複製磁碟區

使用 Trident、您可以利用同一 Kubernetes 叢集中不同命名空間內的現有磁碟區或磁碟區快照建立新磁碟區。

先決條件

在複製磁碟區之前、請確保來源和目的地後端的類型相同、且具有相同的儲存類別。

註 跨命名空間複製僅支援 `ontap-san` 和 `ontap-nas` 儲存驅動程式。不支援唯讀複製。

快速入門

只需幾個步驟即可設定磁碟區複製。

1

配置來源 PVC 以複製磁碟區

來源命名空間擁有者授予存取來源 PVC 中資料的權限。

2

授予在目標命名空間中建立 CR 的權限

叢集管理員授予目標命名空間的擁有者建立 `TridentVolumeReference` CR 的權限。

3

在目標命名空間中建立 **TridentVolumeReference**

目標命名空間的擁有者建立 `TridentVolumeReference` CR 以引用來源 PVC。

4

在目標命名空間中建立複製 PVC

目標命名空間的擁有者建立 PVC 以複製來源命名空間中的 PVC。

設定來源和目的地命名空間

為確保安全，跨命名空間複製磁碟區需要來源命名空間擁有者、叢集管理員和目標命名空間擁有者的協作和操作。每個步驟都會指定使用者角色。

步驟

1. 來源命名空間擁有者：在來源命名空間(namespace1)中建立 PVC(pvc1)，並使用 cloneToNamespace 註解授權與目標命名空間(namespace2)共享。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1
  namespace: namespace1
  annotations:
    trident.netapp.io/cloneToNamespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi
```

Trident 建立 PV 及其後端儲存磁碟區。

註

- 您可以使用逗號分隔的清單將 PVC 共用給多個命名空間。例如，trident.netapp.io/cloneToNamespace: namespace2, namespace3, namespace4。
- 您可以使用 * 共用到所有命名空間。例如，trident.netapp.io/cloneToNamespace: *
- 您可以隨時更新 PVC 以包含 cloneToNamespace 註釋。

2. 叢集管理員：確保已配置正確的 RBAC，以授予目標命名空間擁有者在目標命名空間中建立 TridentVolumeReference CR 的權限(namespace2)。
3. 目標命名空間擁有者：在目標命名空間中建立一個 TridentVolumeReference CR，引用來源命名空間 pvc1。

```
apiVersion: trident.netapp.io/v1
kind: TridentVolumeReference
metadata:
  name: my-first-tvr
  namespace: namespace2
spec:
  pvcName: pvc1
  pvcNamespace: namespace1
```

4. 目標命名空間擁有者：在目標命名空間(namespace2)中建立一個 PVC(pvc2)，使用 cloneFromPVC 或 cloneFromSnapshot，以及 cloneFromNamespace 註解來指定來源 PVC。

```

kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  annotations:
    trident.netapp.io/cloneFromPVC: pvc1
    trident.netapp.io/cloneFromNamespace: namespace1
  name: pvc2
  namespace: namespace2
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: trident-csi
  resources:
    requests:
      storage: 100Gi

```

限制

- 對於使用 ontap-nas-economy 驅動程式配置的 PVC、不支援唯讀複本。

使用 SnapMirror 複製磁碟區

Trident 支援在叢集上的來源磁碟區和對等叢集上的目標磁碟區之間建立鏡像關係，以複製資料進行災難復原。您可以使用名為 Trident Mirror Relationship (TMR) 的命名空間自訂資源定義 (CRD) 來執行下列操作：

- 在磁碟區 (PVC) 之間建立鏡像關係
- 移除磁碟區之間的鏡射關係
- 打破鏡像關係
- 在災難情況下 (容錯移轉) 提升次要磁碟區
- 在計劃內故障轉移或遷移期間，實現應用程式在叢集間的無損遷移

複寫的前提條件

在開始之前、請確保符合下列先決條件：

ONTAP 叢集

- **Trident**：使用 ONTAP 作為後端的來源 Kubernetes 叢集和目標 Kubernetes 叢集上必須存在 Trident 版本 22.10 或更新版本。
- **授權**：必須在來源和目的地 ONTAP 叢集上啟用使用資料保護套件的 ONTAP SnapMirror 非同步授權。如需詳細資訊，請參閱 "[SnapMirror 授權總覽 \(ONTAP\)](#)"。

從 ONTAP 9.10.1 開始，所有授權均以 NetApp 授權檔案 (NLF) 的形式提供，這是一個可啟用多項功能的單一檔案。請參閱 "[ONTAP One 隨附的授權](#)" 以取得更多資訊。

註 | 僅支援 SnapMirror 非同步保護。

對等

- 叢集和 SVM：ONTAP 儲存後端必須建立對等連線。如需詳細資訊，請參閱["叢集和 SVM 對等連接概述"](#)。

重要 | 確保兩個 ONTAP 叢集之間複寫關係中使用的 SVM 名稱是唯一的。

- Trident 和 SVM：對等遠端 SVM 必須可供目的地叢集上的 Trident 使用。

支援的驅動程式

NetApp Trident 支援使用下列驅動程式支援的儲存類別、透過 NetApp SnapMirror 技術進行磁碟區複寫：

ontap-nas：NFS **ontap-san**：iSCSI **ontap-san**：FC **ontap-san**：NVMe/TCP（需要最低 ONTAP 版本 9.15.1）

註

ASA r2 系統不支援使用 SnapMirror 的磁碟區複寫。如需 ASA r2 系統的相關資訊、請參閱["了解 ASA r2 儲存系統"](#)。

建立鏡射 PVC

請依照這些步驟並使用 CRD 範例，在主磁碟區和次要磁碟區之間建立鏡像關係。

步驟

1. 在主要 Kubernetes 叢集上執行下列步驟：
 - a. 建立一個帶有 `trident.netapp.io/replication: true` 參數的 StorageClass 物件。

範例

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-nas
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  fsType: "nfs"
  trident.netapp.io/replication: "true"
```

- b. 使用先前建立的 StorageClass 建立 PVC。

範例

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  storageClassName: csi-nas
```

- c. 建立包含本機資訊的 MirrorRelationship CR。

範例

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: promoted
  volumeMappings:
    - localPVCName: csi-nas
```

Trident 取得磁碟區的內部資訊和磁碟區的目前資料保護 (DP) 狀態，然後填入 MirrorRelationship 的狀態欄位。

- d. 取得 TridentMirrorRelationship CR 以取得 PVC 的內部名稱和 SVM。

```
kubectl get tmr csi-nas
```

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
  generation: 1
spec:
  state: promoted
  volumeMappings:
    - localPVCName: csi-nas
status:
  conditions:
    - state: promoted
      localVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
      localPVCName: csi-nas
      observedGeneration: 1
```

2. 在次要 Kubernetes 叢集上執行下列步驟：

a. 建立 StorageClass 並設定 trident.netapp.io/replication: true 參數。

範例

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: csi-nas
provisioner: csi.trident.netapp.io
parameters:
  trident.netapp.io/replication: true
```

b. 建立包含目標和來源資訊的 MirrorRelationship CR。

範例

```
kind: TridentMirrorRelationship
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  state: established
  volumeMappings:
    - localPVCName: csi-nas
      remoteVolumeHandle:
"datavserver:trident_pvc_3bedd23c_46a8_4384_b12b_3c38b313c1e1"
```

Trident 將使用已設定的關係原則名稱（或 ONTAP 的預設值）建立 SnapMirror 關係並將其初始化。

- c. 建立一個 PVC，將先前建立的 StorageClass 作為輔助（SnapMirror 目標）。

範例

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: csi-nas
  annotations:
    trident.netapp.io/mirrorRelationship: csi-nas
spec:
  accessModes:
  - ReadWriteMany
resources:
  requests:
    storage: 1Gi
storageClassName: csi-nas
```

Trident 將檢查 TridentMirrorRelationship CRD，如果關係不存在，則建立磁碟區失敗。如果關係存在，Trident 將確保將新 FlexVol 磁碟區放置在與 MirrorRelationship 中定義的遠端 SVM 對等的 SVM 上。

Volume 複寫狀態

Trident 鏡像關係（TMR）是一種 CRD，它表示 PVC 之間複製關係的一端。目標 TMR 具有一個狀態，該狀態告知 Trident 所需的狀態。目標 TMR 具有以下狀態：

- 已建立：本地 PVC 是鏡像關係的目標磁碟區，這是一個新的關係。
- **Promoted**：本機 PVC 為 ReadWrite 且可掛載，目前沒有鏡像關係生效。
- 重新建立：本機 PVC 是鏡像關係的目的地 Volume，且先前也處於該鏡像關係中。
 - 如果目標 volume 曾經與來源 volume 有關聯，則必須使用重新建立的狀態，因為它會覆寫目標 volume 的內容。
 - 如果磁碟區之前未與來源建立關係，則重新建立的狀態將會失敗。

在非計劃性容錯移轉期間提升次要 **PVC**

在次要 Kubernetes 叢集上執行下列步驟：

- 將 TridentMirrorRelationship 的 `spec.state` 欄位更新為 `promoted`。

在計劃故障切換期間推廣次要 **PVC**

在計劃性容錯移轉（移轉）期間，執行以下步驟以提升次要 PVC：

步驟

1. 在主 Kubernetes 叢集上、建立 PVC 的快照、並等待快照建立完成。
2. 在主 Kubernetes 叢集上、建立 SnapshotInfo CR 以取得內部詳細資訊。

範例

```
kind: SnapshotInfo
apiVersion: trident.netapp.io/v1
metadata:
  name: csi-nas
spec:
  snapshot-name: csi-nas-snapshot
```

3. 在輔助 Kubernetes 叢集上，將 *TridentMirrorRelationship* CR 的 *spec.state* 欄位更新為 *promoted*，並將 *spec.promotedSnapshotHandle* 更新為快照的 *internalName*。
4. 在輔助 Kubernetes 叢集上，確認 *TridentMirrorRelationship* 的狀態 (*status.state* 欄位) 是否已提升。

故障轉移後還原鏡像關係

在還原鏡射關係之前、請選擇您要做為新主要端的一端。

步驟

1. 在輔助 Kubernetes 叢集上，確保 *TridentMirrorRelationship* 上 *spec.remoteVolumeHandle* 欄位的值已更新。
2. 在輔助 Kubernetes 叢集上，將 *TridentMirrorRelationship* 的 *spec.mirror* 欄位更新為 *reestablished*。

其他作業

Trident 支援對主要磁碟區和次要磁碟區執行下列操作：

將主要 PVC 複寫到新的次要 PVC

請確保您已備有主要 PVC 和次要 PVC。

步驟

1. 從已建立的輔助 (目標) 叢集中刪除 *PersistentVolumeClaim* 和 *TridentMirrorRelationship* CRD。
2. 從主 (來源) 叢集中刪除 *TridentMirrorRelationship* CRD。
3. 在主 (來源) 叢集上為要建立的新輔助 (目標) PVC 建立一個新的 *TridentMirrorRelationship* CRD。

調整鏡像、主要或次要 PVC 的大小

PVC 可以像往常一樣調整大小，如果資料量超過目前大小，ONTAP 將自動擴展任何目標 *FlexVol*。

從 PVC 移除複寫

若要移除複寫，請對目前的次要磁碟區執行下列其中一項作業：

- 刪除輔助 PVC 上的 *MirrorRelationship*。這將破壞複寫關係。

- 或者，將 `spec.state` 欄位更新為 `promoted`。

刪除一個 PVC（之前已鏡像）

Trident 會檢查是否有複寫的 PVC，並在嘗試刪除磁碟區之前釋放複寫關係。

刪除 TMR

刪除鏡像關係一側的 TMR 會導致剩餘的 TMR 在 Trident 完成刪除作業之前轉換為 `promoted` 狀態。如果選擇刪除的 TMR 已處於 `promoted` 狀態，表示不存在鏡像關係，此時 TMR 將被移除，Trident 會將本機 PVC 提升為 `ReadWrite` 狀態。此刪除操作會釋放 ONTAP 中本機磁碟區的 `SnapMirror` 中繼資料。如果將來在鏡像關係中使用此磁碟區，則在建立新鏡像關係時必須使用狀態為 `established` 的磁碟區複寫新 TMR。

ONTAP 上線時更新鏡像關係

鏡像關係建立後可隨時更新。您可以使用 ``state: promoted`` 或 ``state: reestablished`` 欄位來更新關係。將目標磁碟區提升為常規 `ReadWrite` 磁碟區時、您可以使用 `promotedSnapshotHandle` 指定要將目前磁碟區還原到的特定快照。

ONTAP 離線時更新鏡像關係

您可以使用 CRD 執行 `SnapMirror` 更新，而無需 Trident 與 ONTAP 叢集直接連線。請參考以下 `TridentActionMirrorUpdate` 範例格式：

範例

```
apiVersion: trident.netapp.io/v1
kind: TridentActionMirrorUpdate
metadata:
  name: update-mirror-b
spec:
  snapshotHandle: "pvc-1234/snapshot-1234"
  tridentMirrorRelationshipName: mirror-b
```

`status.state` 反映 `TridentActionMirrorUpdate` CRD 的狀態。它可以取值於 `Succeeded`、`In Progress` 或 `Failed`。

使用 CSI 拓撲

Trident 可以利用 "[CSI Topology 功能](#)" 選擇性地建立磁碟區並將其附加到 Kubernetes 叢集中的節點。

概況

使用 CSI Topology 功能，可以根據區域和可用區將磁碟區的存取權限制在部分節點上。如今，雲端供應商允許 Kubernetes 管理員建立基於可用區的節點。節點可以位於同一區域內的不同可用區，也可以跨越多個區域。為了方便在多可用區架構中為工作負載配置磁碟區，Trident 使用 CSI Topology。

提示 | 深入瞭解 CSI Topology 功能 ["這裡"](#)。

Kubernetes 提供兩種獨特的磁碟區繫結模式：

- 將 `VolumeBindingMode` 設為 `Immediate` 時、Trident 會在不感知拓撲的情況下建立磁碟區。磁碟區繫結和動態資源配置會在建立 PVC 時處理。這是預設 `VolumeBindingMode`、適用於不強制執行拓撲限制的叢集。持續磁碟區的建立不會依賴要求 Pod 的排程需求。
- 將 `VolumeBindingMode` 設為 `WaitForFirstConsumer` 時、系統會延遲建立 PVC 的持續磁碟區並將其繫結、直到排程並建立使用該 PVC 的 Pod 為止。如此一來、建立的磁碟區就能符合拓撲需求所強制執行的排程限制。

註 | `WaitForFirstConsumer` 綁定模式不需要拓撲標籤。它可以獨立於 CSI 拓撲功能使用。

您需要準備的項目

若要用 CSI Topology，您需要下列項目：

- 執行 "[支援的 Kubernetes 版本](#)" 的 Kubernetes 叢集

```
kubectl version
Client Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedaefd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:50:19Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"19",
GitVersion:"v1.19.3",
GitCommit:"1e11e4a2108024935ecfcb2912226cedaefd99df",
GitTreeState:"clean", BuildDate:"2020-10-14T12:41:49Z",
GoVersion:"go1.15.2", Compiler:"gc", Platform:"linux/amd64"}
```

- 叢集中的節點應該帶有用於引入拓撲感知的標籤(`topology.kubernetes.io/region` 和 `topology.kubernetes.io/zone`)。這些標籤*應該在安裝 Trident 之前就存在於叢集中的節點上*，以便 Trident 能夠感知拓撲。

```

kubect1 get nodes -o=jsonpath='{range .items[*]}[{"metadata.name"},
{"metadata.labels}]{"\n"}{end}' | grep --color "topology.kubernetes.io"
[node1,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kubernete
s.io/arch":"amd64","kubernetes.io/hostname":"node1","kubernetes.io/
os":"linux","node-
role.kubernetes.io/master":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-a"}}
[node2,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kubernete
s.io/arch":"amd64","kubernetes.io/hostname":"node2","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-b"}}
[node3,
{"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kubernete
s.io/arch":"amd64","kubernetes.io/hostname":"node3","kubernetes.io/
os":"linux","node-
role.kubernetes.io/worker":"","topology.kubernetes.io/region":"us-
east1","topology.kubernetes.io/zone":"us-east1-c"}}

```

步驟 1：建立拓樸感知後端

Trident 儲存後端可設計為根據可用區選擇性地配置磁碟區。每個後端都可以包含一個可選的 `supportedTopologies` 資料區塊，用於指定支援的區域和可用區清單。對於使用此類後端的 `StorageClasses`，僅當調度到受支援區域/可用區的應用程式請求時，才會建立相應的磁碟區。

以下是後端定義範例：

YAML

```
---
version: 1
storageDriverName: ontap-san
backendName: san-backend-us-east1
managementLIF: 192.168.27.5
svm: iscsi_svm
username: admin
password: password
supportedTopologies:
  - topology.kubernetes.io/region: us-east1
    topology.kubernetes.io/zone: us-east1-a
  - topology.kubernetes.io/region: us-east1
    topology.kubernetes.io/zone: us-east1-b
```

JSON

```
{
  "version": 1,
  "storageDriverName": "ontap-san",
  "backendName": "san-backend-us-east1",
  "managementLIF": "192.168.27.5",
  "svm": "iscsi_svm",
  "username": "admin",
  "password": "password",
  "supportedTopologies": [
    {
      "topology.kubernetes.io/region": "us-east1",
      "topology.kubernetes.io/zone": "us-east1-a"
    },
    {
      "topology.kubernetes.io/region": "us-east1",
      "topology.kubernetes.io/zone": "us-east1-b"
    }
  ]
}
```

註 supportedTopologies 用於為每個後端提供區域和可用區清單。這些區域和可用區代表可以在 StorageClass 中提供的允許值清單。對於包含後端提供的區域和可用區子集的 StorageClasses，Trident 會在後端建立一個磁碟區。

您也可以按儲存池定義 supportedTopologies。請參閱以下範例：

```
---
version: 1
storageDriverName: ontap-nas
backendName: nas-backend-us-centrall
managementLIF: 172.16.238.5
svm: nfs_svm
username: admin
password: password
supportedTopologies:
  - topology.kubernetes.io/region: us-centrall
    topology.kubernetes.io/zone: us-centrall-a
  - topology.kubernetes.io/region: us-centrall
    topology.kubernetes.io/zone: us-centrall-b
storage:
  - labels:
      workload: production
    supportedTopologies:
      - topology.kubernetes.io/region: us-centrall
        topology.kubernetes.io/zone: us-centrall-a
  - labels:
      workload: dev
    supportedTopologies:
      - topology.kubernetes.io/region: us-centrall
        topology.kubernetes.io/zone: us-centrall-b
```

在此範例中、region 和 zone 標籤代表儲存資源池的位置。topology.kubernetes.io/region 和 topology.kubernetes.io/zone 則指定可從何處使用儲存資源池。

步驟 2：定義可感知拓撲的 StorageClasses

根據提供給叢集中節點的拓撲標籤、StorageClasses 可定義為包含拓撲資訊。這將決定哪些儲存資源池可作為 PVC 要求的候選對象、以及哪些節點子集可以使用 Trident 所配置的磁碟區。

請參閱以下範例：

```

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: netapp-san-us-east1
provisioner: csi.trident.netapp.io
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
  - matchLabelExpressions:
    - key: topology.kubernetes.io/zone
      values:
        - us-east1-a
        - us-east1-b
    - key: topology.kubernetes.io/region
      values:
        - us-east1
parameters:
  fsType: ext4

```

在上述提供的 StorageClass 定義中，volumeBindingMode 被設定為 WaitForFirstConsumer。使用此 StorageClass 請求的 PVC 只有在 Pod 中被引用後才會生效。而且，allowedTopologies 提供要使用的 zone 和 region。netapp-san-us-east1 StorageClass 會在上述定義的 san-backend-us-east1 後端上建立 PVC。

步驟 3：建立並使用 PVC

建立 StorageClass 並映射到後端後，您現在可以建立 PVC。

請看下面的例子 spec：

```

---
kind: PersistentVolumeClaim
apiVersion: v1
metadata: null
name: pvc-san
spec: null
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: netapp-san-us-east1

```

使用此資訊清單建立 PVC 將產生下列結果：

```

kubect1 create -f pvc.yaml
persistentvolumeclaim/pvc-san created
kubect1 get pvc
NAME          STATUS      VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS
AGE
pvc-san      Pending
2s
kubect1 describe pvc
Name:          pvc-san
Namespace:     default
StorageClass:  netapp-san-us-east1
Status:        Pending
Volume:
Labels:        <none>
Annotations:   <none>
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:
Access Modes:
VolumeMode:    Filesystem
Mounted By:    <none>
Events:
  Type      Reason              Age   From
  ----      -
Normal    WaitForFirstConsumer  6s   persistentvolume-controller
waiting
for first consumer to be created before binding

```

若要讓 Trident 建立磁碟區並將其繫結至 PVC，請在 Pod 中使用 PVC。請參閱以下範例：

```

apiVersion: v1
kind: Pod
metadata:
  name: app-pod-1
spec:
  affinity:
    nodeAffinity:
      requiredDuringSchedulingIgnoredDuringExecution:
        nodeSelectorTerms:
          - matchExpressions:
              - key: topology.kubernetes.io/region
                operator: In
                values:
                  - us-east1
            preferredDuringSchedulingIgnoredDuringExecution:
              - weight: 1
                preference:
                  matchExpressions:
                    - key: topology.kubernetes.io/zone
                      operator: In
                      values:
                        - us-east1-a
                        - us-east1-b
      securityContext:
        runAsUser: 1000
        runAsGroup: 3000
        fsGroup: 2000
    volumes:
      - name: voll
        persistentVolumeClaim:
          claimName: pvc-san
    containers:
      - name: sec-ctx-demo
        image: busybox
        command: [ "sh", "-c", "sleep 1h" ]
        volumeMounts:
          - name: voll
            mountPath: /data/demo
        securityContext:
          allowPrivilegeEscalation: false

```

這個 podSpec 指示 Kubernetes 將 pod 排程到 us-east1 區域中存在的節點上，並從 us-east1-a 或 us-east1-b 區域中存在的任何節點中進行選擇。

請參閱下列輸出：

```

kubect1 get pods -o wide
NAME             READY   STATUS    RESTARTS   AGE   IP             NODE
NOMINATED NODE  READINESS GATES
app-pod-1       1/1     Running   0           19s   192.168.25.131 node2
<none>          <none>
kubect1 get pvc -o wide
NAME             STATUS    VOLUME                                     CAPACITY
ACCESS MODES     STORAGECLASS          AGE   VOLUMEMODE
pvc-san          Bound     pvc-ecb1e1a0-840c-463b-8b65-b3d033e2e62b 300Mi
RWO               netapp-san-us-east1  48s   Filesystem

```

更新後端以包含 `supportedTopologies`

可以更新現有後端，使其包含 `supportedTopologies` 清單，使用 `tridentctl backend update`。這不會影響已配置的磁碟區，並且僅用於後續的 PVC。

尋找更多資訊

- ["管理容器資源"](#)
- ["nodeSelector"](#)
- ["親和性與反親和性"](#)
- ["污點與容忍度"](#)

使用快照

Kubernetes 持久磁碟區 (PV) 的磁碟區快照功能可以建立磁碟區的特定時間點副本。您可以建立使用 Trident 建立的磁碟區快照、匯入在 Trident 外部建立的快照、從現有快照建立新磁碟區，以及從快照還原磁碟區資料。

概況

Volume snapshot 受 `ontap-nas`、`ontap-nas-flexgroup`、`ontap-san`、`ontap-san-economy`、`solidfire-san`、`azure-netapp-files` 和 `google-cloud-netapp-volumes` 驅動程式支援。

開始之前

若要使用快照，您必須擁有外部快照控制器和自訂資源定義 (CRD)。這是 Kubernetes 編排器 (例如：`Kubeadm`、`GKE`、`OpenShift`) 的職責。

如果您的 Kubernetes 發行版不包含快照控制器和 CRD，請參閱 [部署 Volume Snapshot Controller](#)。

註

如果要在 GKE 環境中建立按需磁碟區快照，請勿建立快照控制器。GKE 使用內建的隱藏快照控制器。

建立 Volume Snapshot

步驟

1. 建立 VolumeSnapshotClass。如需詳細資訊、請參閱 "VolumeSnapshotClass"。
 - driver 指向 Trident CSI 驅動程式。
 - deletionPolicy 可以是 Delete 或 Retain。設定為 Retain 時，即使 VolumeSnapshot 物件被刪除，儲存叢集上的底層實體快照也會被保留。

範例

```
cat snap-sc.yaml
```

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: csi-snapclass
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

2. 建立現有 PVC 的快照。

範例

- 此範例會建立現有 PVC 的快照。

```
cat snap.yaml
```

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: pvc1-snap
spec:
  volumeSnapshotClassName: csi-snapclass
  source:
    persistentVolumeClaimName: pvc1
```

- 此範例為名為 pvc1 的 PVC 建立磁碟區快照物件，並將快照名稱設為 pvc1-snap
 - VolumeSnapshot 類似於 PVC，並與 VolumeSnapshotContent 物件相關聯，該物件代表實際快照。

```
kubectl create -f snap.yaml
volumesnapshot.snapshot.storage.k8s.io/pvc1-snap created

kubectl get volumesnapshots
NAME                                AGE
pvc1-snap                           50s
```

- 您可以透過描述該 VolumeSnapshotContent 物件來識別 pvc1-snap VolumeSnapshot。該 Snapshot Content Name 會識別提供此快照的 VolumeSnapshotContent 物件。該 Ready To Use 參數表示此快照可用於建立新的 PVC。

```
kubectl describe volumesnapshots pvc1-snap
Name:          pvc1-snap
Namespace:    default
...
Spec:
  Snapshot Class Name:    pvc1-snap
  Snapshot Content Name:  snapcontent-e8d8a0ca-9826-11e9-9807-
525400f3f660
  Source:
    API Group:
    Kind:      PersistentVolumeClaim
    Name:      pvc1
Status:
  Creation Time:  2019-06-26T15:27:29Z
  Ready To Use:  true
  Restore Size:  3Gi
...
```

從磁碟區快照建立 PVC

您可以使用 `dataSource` 來建立 PVC，使用名為 `` 的 VolumeSnapshot 作為資料來源。建立 PVC 後，可以將其附加到 pod 上，並像使用其他 PVC 一樣使用它。

警告

PVC 將在與來源磁碟區相同的後端建立。請參閱["知識庫：無法在備用後端從 Trident PVC 快照建立 PVC"](#)。

以下範例使用 pvc1-snap 作為資料來源建立 PVC。

```
cat pvc-from-snap.yaml
```

```

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: golden
  resources:
    requests:
      storage: 3Gi
  dataSource:
    name: pvcl-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io

```

匯入磁碟區快照

Trident 支援"[Kubernetes 預先配置快照流程](#)"讓叢集管理員建立 `VolumeSnapshotContent` 物件並匯入在 Trident 外部建立的快照。

開始之前

Trident 必須已建立或匯入快照的父 Volume 。

步驟

- 叢集管理員：建立一個 `VolumeSnapshotContent` 物件，參照後端快照。這會在 Trident 中啟動快照工作流程。
 - 在 annotations 中將後端快照的名稱指定為 `trident.netapp.io/internalSnapshotName: <"backend-snapshot-name">`。
 - 請在 <name-of-parent-volume-in-trident>/<volume-snapshot-content-name> 中指定 `snapshotHandle`。這是外部快照工具在 `ListSnapshots` 呼叫中提供給 Trident 的唯一資訊。

註

<volumeSnapshotContentName> 由於 CR 命名限制，無法始終與後端快照名稱相符。

範例

以下範例建立了一個 `VolumeSnapshotContent` 物件，該物件引用後端快照 `snap-01`。

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotContent
metadata:
  name: import-snap-content
  annotations:
    trident.netapp.io/internalSnapshotName: "snap-01" # This is the
name of the snapshot on the backend
spec:
  deletionPolicy: Retain
  driver: csi.trident.netapp.io
  source:
    snapshotHandle: pvc-f71223b5-23b9-4235-bbfe-e269ac7b84b0/import-
snap-content # <import PV name or source PV name>/<volume-snapshot-
content-name>
  volumeSnapshotRef:
    name: import-snap
    namespace: default
```

2. **Cluster admin**：建立 VolumeSnapshot CR，參照 VolumeSnapshotContent 物件。這會要求存取權限，以便在指定的命名空間中使用 VolumeSnapshot。

範例

以下範例會建立一個 VolumeSnapshot CR，名為 import-snap，該 CR 會參照 VolumeSnapshotContent，名為 import-snap-content。

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: import-snap
spec:
  # volumeSnapshotClassName: csi-snapclass (not required for pre-
provisioned or imported snapshots)
  source:
    volumeSnapshotContentName: import-snap-content
```

3. ***內部處理（無需操作）**：*外部快照程式識別新建立的 VolumeSnapshotContent 並執行 ListSnapshots 呼叫。Trident 建立 TridentSnapshot。
 - 外部 snapshotter 會將 VolumeSnapshotContent 設為 readyToUse，並將 VolumeSnapshot 設為 true。
 - Trident 返回 readyToUse=true。
4. **任何使用者**：建立一個 PersistentVolumeClaim 以參照新的 VolumeSnapshot，其中 spec.dataSource（或 spec.dataSourceRef）名稱是 VolumeSnapshot 名稱。

範例

以下範例建立了一個 PVC，參考名為 `VolumeSnapshot` 的 `import-snap`。

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snap
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: simple-sc
  resources:
    requests:
      storage: 1Gi
  dataSource:
    name: import-snap
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
```

使用快照恢復磁碟區資料

預設情況下，快照目錄處於隱藏狀態，以確保使用 `ontap-nas` 和 `ontap-nas-economy` 驅動程式配置的磁碟區的最大相容性。啟用 `.snapshot` 目錄即可直接從快照還原資料。

使用 `volume snapshot restore ONTAP CLI` 將磁碟區還原到先前快照中記錄的狀態。

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

註

還原 Snapshot 複本時、現有的 Volume 組態會被覆寫。建立 Snapshot 複本後對 Volume 資料所做的變更將會遺失。

從快照進行就地 **Volume** 還原

Trident 使用 `TridentActionSnapshotRestore (TASR) CR` 從快照快速進行原廠磁碟區復原。此 CR 作為一項命令式 Kubernetes 操作運行，操作完成後不會持久保存。

Trident 支援在 `ontap-san`、`ontap-san-economy`、`ontap-nas`、`ontap-nas-flexgroup`、`azure-netapp-files`、`google-cloud-netapp-volumes` 和 `solidfire-san` 驅動程式上進行快照還原。

開始之前

您必須擁有已綁定的 PVC 和可用的磁碟區快照。

- 確認 PVC 狀態為已綁定。

```
kubectl get pvc
```

- 確認磁碟區快照已準備就緒可供使用。

```
kubectl get vs
```

步驟

1. 建立 TASR CR。此範例為 PVC `pvc1` 和磁碟區快照 `pvc1-snapshot` 建立 CR。

註 | TASR CR 必須位於 PVC 和 VS 存在的命名空間中。

```
cat tasr-pvc1-snapshot.yaml
```

```
apiVersion: trident.netapp.io/v1
kind: TridentActionSnapshotRestore
metadata:
  name: trident-snap
  namespace: trident
spec:
  pvcName: pvc1
  volumeSnapshotName: pvc1-snapshot
```

2. 套用 CR 從快照還原。此範例從快照還原 `pvc1`。

```
kubectl create -f tasr-pvc1-snapshot.yaml
```

```
tridentactionsnapshotrestore.trident.netapp.io/trident-snap created
```

結果

Trident 會從快照還原資料。您可以驗證快照還原狀態：

```
kubectl get tasr -o yaml
```

```
apiVersion: trident.netapp.io/v1
items:
- apiVersion: trident.netapp.io/v1
  kind: TridentActionSnapshotRestore
  metadata:
    creationTimestamp: "2023-04-14T00:20:33Z"
    generation: 3
    name: trident-snap
    namespace: trident
    resourceVersion: "3453847"
    uid: <uid>
  spec:
    pvcName: pvcl
    volumeSnapshotName: pvcl-snapshot
  status:
    startTime: "2023-04-14T00:20:34Z"
    completionTime: "2023-04-14T00:20:37Z"
    state: Succeeded
kind: List
metadata:
  resourceVersion: ""
```

註

- 在大多數情況下，Trident 不會在操作失敗後自動重試。您需要再次執行該操作。
- 沒有管理員權限的 Kubernetes 使用者可能需要管理員授予權限才能在其應用程式命名空間中建立 TASR CR。

刪除具有相關快照的 PV

刪除包含關聯快照的持久性磁碟區時，對應的 Trident 磁碟區會更新為「正在刪除」狀態。刪除磁碟區快照即可刪除 Trident 磁碟區。

部署 Volume Snapshot Controller

如果您的 Kubernetes 發行版不包含快照控制器和 CRD、您可以依照下列方式部署它們。

步驟

1. 建立 Volume Snapshot CRD。

```
cat snapshot-setup.sh
```

```
#!/bin/bash
# Create volume snapshot CRDs
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshotclasses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshotcontents.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/client/config/crd/snapshot.storage.k8s.io_volumesnapshots.yaml
```

2. 建立 Snapshot Controller。

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml
```

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-6.1/deploy/kubernetes/snapshot-controller/setup-snapshot-controller.yaml
```

註

如有必要，請打開 `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` 並更新 `namespace` 到您的命名空間。

相關連結

- ["Volume 快照"](#)
- ["VolumeSnapshotClass"](#)

使用磁碟區群組快照

Kubernetes 持久性磁碟區 (PV) 的磁碟區群組快照 NetApp Trident 提供建立多個磁碟區快照 (磁碟區快照群組) 的功能。此磁碟區群組快照代表在同一時間點從多個磁碟區取得的複本。

註

VolumeGroupSnapshot 是 Kubernetes 中的一項測試版功能，使用測試版 API。Kubernetes 1.32 是 VolumeGroupSnapshot 的最低版本要求。

建立 Volume Group 快照

以下儲存驅動程式支援 Volume group snapshot：

- `ontap-san driver` - 僅適用於 iSCSI 和 FC 協定，不適用於 NVMe/TCP 協定。
- `ontap-san-economy` - 僅適用於 iSCSI 協定。
- `ontap-nas`

註 NetApp ASA r2 或 AFX 儲存系統不支援 Volume group snapshot。

開始之前

- 請確保您的 Kubernetes 版本為 K8s 1.32 或更高版本。
- 若要使用快照，您必須擁有外部快照控制器和自訂資源定義 (CRD)。這是 Kubernetes 編排器 (例如：Kubeadm、GKE、OpenShift) 的職責。

如果您的 Kubernetes 發行版不包含外部快照控制器和 CRD，請參閱 [部署 Volume Snapshot Controller](#)。

註 如果要在 GKE 環境中建立按需磁碟區組快照，請勿建立快照控制器。GKE 使用內建的隱藏快照控制器。

- 在快照控制器 YAML 中，將 `CSIVolumeGroupSnapshot` 功能閘設定為 `'true'`，以確保啟用磁碟區群組快照。
- 在建立磁碟區群組快照之前，請先建立所需的磁碟區群組快照類別。
- 確保所有 PVC/ 磁碟區都在同一 SVM 上、才能建立 `VolumeGroupSnapshot`。

步驟

- 在建立 `VolumeGroupSnapshot` 之前，請先建立 `VolumeGroupSnapshotClass`。如需更多資訊，請參閱 ["VolumeGroupSnapshotClass"](#)。

```
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshotClass
metadata:
  name: csi-group-snap-class
  annotations:
    kubernetes.io/description: "Trident group snapshot class"
driver: csi.trident.netapp.io
deletionPolicy: Delete
```

- 使用現有的儲存類別建立具有所需標籤的 PVC，或將這些標籤新增至現有的 PVC。

以下範例使用 `pvc1-group-snap` 作為資料來源和標籤 `consistentGroupSnapshot: groupA` 建立 PVC。請根據您的需求定義標籤的鍵和值。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvcl-group-snap
  labels:
    consistentGroupSnapshot: groupA
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Mi
  storageClassName: sc1-1
```

- 建立具有與 PVC 中指定的標籤(`consistentGroupSnapshot: groupA` 相同的 VolumeGroupSnapshot。

此範例會建立磁碟區群組快照：

```
apiVersion: groupsnapshot.storage.k8s.io/v1beta1
kind: VolumeGroupSnapshot
metadata:
  name: "vgs1"
  namespace: trident
spec:
  volumeGroupSnapshotClassName: csi-group-snap-class
  source:
    selector:
      matchLabels:
        consistentGroupSnapshot: groupA
```

使用群組快照恢復磁碟區資料

您可以使用作為 Volume Group Snapshot 一部分建立的個別快照來還原個別 Persistent Volume。您無法將 Volume Group Snapshot 作為一個單元進行還原。

使用 volume snapshot restore ONTAP CLI 將磁碟區還原到先前快照中記錄的狀態。

```
cluster1::*> volume snapshot restore -vserver vs0 -volume vol3 -snapshot
vol3_snap_archive
```

註

還原 Snapshot 複本時、現有的 Volume 組態會被覆寫。建立 Snapshot 複本後對 Volume 資料所做的變更將會遺失。

從快照進行就地 **Volume** 還原

Trident 使用 `TridentActionSnapshotRestore` (TASR) CR 從快照快速進行原廠磁碟區復原。此 CR 作為一項命令式 Kubernetes 操作運行，操作完成後不會持久保存。

如需詳細資訊，請參閱 "[從快照進行就地 Volume 還原](#)"。

刪除具有相關群組快照的 **PV**

刪除群組 Volume 快照時：

- 您可以整體刪除 `VolumeGroupSnapshots`，而不是刪除群組中的單一快照。
- 如果刪除 `PersistentVolumes` 時該 `PersistentVolume` 已存在快照，Trident 會將該磁碟區移至「刪除中」狀態，因為必須先移除快照才能安全地移除該磁碟區。
- 如果使用分組快照建立了複本，然後要刪除該群組，則會開始複本分割作業，並且在分割完成之前無法刪除該群組。

部署 **Volume Snapshot Controller**

如果您的 Kubernetes 發行版不包含快照控制器和 CRD、您可以依照下列方式部署它們。

步驟

1. 建立 Volume Snapshot CRD。

```
cat snapshot-setup.sh
```

```
#!/bin/bash
# Create volume snapshot CRDs
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotclasses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotcontents.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshots.yaml
```

2. 建立 Snapshot Controller。

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-8.2/deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml
```

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/release-8.2/deploy/kubernetes/snapshot-controller/setup-snapshot-controller.yaml
```

註

如有必要，請打開 `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` 並更新 `namespace` 到您的命名空間。

相關連結

- ["VolumeGroupSnapshotClass"](#)
- ["Volume 快照"](#)

版權資訊

Copyright © 2026 NetApp, Inc. 版權所有。台灣印製。非經版權所有人事先書面同意，不得將本受版權保護文件的任何部分以任何形式或任何方法（圖形、電子或機械）重製，包括影印、錄影、錄音或儲存至電子檢索系統中。

由 NetApp 版權資料衍伸之軟體必須遵守下列授權和免責聲明：

此軟體以 NETAPP「原樣」提供，不含任何明示或暗示的擔保，包括但不限於有關適售性或特定目的適用性之擔保，特此聲明。於任何情況下，就任何已造成或基於任何理論上責任之直接性、間接性、附隨性、特殊性、懲罰性或衍生性損害（包括但不限於替代商品或服務之採購；使用、資料或利潤上的損失；或企業營運中斷），無論是在使用此軟體時以任何方式所產生的契約、嚴格責任或侵權行為（包括疏忽或其他）等方面，NetApp 概不負責，即使已被告知有前述損害存在之可能性亦然。

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。NetApp 不承擔因使用本文所述之產品而產生的責任或義務，除非明確經過 NetApp 書面同意。使用或購買此產品並不會在依據任何專利權、商標權或任何其他 NetApp 智慧財產權的情況下轉讓授權。

本手冊所述之產品受到一項（含）以上的美國專利、國外專利或申請中專利所保障。

有限權利說明：政府機關的使用、複製或公開揭露須受 DFARS 252.227-7013（2014 年 2 月）和 FAR 52.227-19（2007 年 12 月）中的「技術資料權利 - 非商業項目」條款 (b)(3) 小段所述之限制。

此處所含屬於商業產品和 / 或商業服務（如 FAR 2.101 所定義）的資料均為 NetApp, Inc. 所有。根據本協議提供的所有 NetApp 技術資料和電腦軟體皆屬於商業性質，並且完全由私人出資開發。美國政府對於該資料具有非專屬、非轉讓、非轉授權、全球性、有限且不可撤銷的使用權限，僅限於美國政府為傳輸此資料所訂合約所允許之範圍，並基於履行該合約之目的方可使用。除非本文另有規定，否則未經 NetApp Inc. 事前書面許可，不得逕行使用、揭露、重製、修改、履行或展示該資料。美國政府授予國防部之許可權利，僅適用於 DFARS 條款 252.227-7015(b)（2014 年 2 月）所述權利。

商標資訊

NETAPP、NETAPP 標誌及 <http://www.netapp.com/TM> 所列之標章均為 NetApp, Inc. 的商標。文中所涉及的所有其他公司或產品名稱，均為其各自所有者的商標，不得侵犯。