# 使用 **Trident**

Trident

NetApp
April 08, 2026

# 目錄

# 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-100.2506.0.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
```

## 準備配置 SMB Volume

您可以使用 `ontap-nas` 驅動程式配置 SMB 磁碟區。

> 您必須在 SVM 上同時設定 NFS 和 SMB/CIFS 通訊協定，才能為 ONTAP 內部部署叢集建立 `ontap-nas-economy` SMB Volume。若未設定其中任一通訊協定，將導致 SMB Volume 建立失敗。

> `autoExportPolicy` 不支援 SMB 磁碟區。

開始之前

在配置 SMB 磁碟區之前、您必須具備以下條件。

- Kubernetes 叢集包含一個 Linux 控制器節點和至少一個執行 Windows Server 2022 的 Windows 工作節點。Trident 僅支援掛載到在 Windows 節點上執行的 pod 的 SMB 磁碟區。
- 至少需要一個包含您的 Active Directory 憑證的 Trident 金鑰。要產生金鑰 `smbcreds`：

```
kubectl create secret generic smbcreds --from-literal username=user
--from-literal password='password'
```

- CSI Proxy 設定為 Windows 服務。若要設定 `csi-proxy`，請參閱"GitHub：CSI Proxy"或"GitHub：適用於 Windows 的 CSI Proxy"以瞭解在 Windows 上執行的 Kubernetes 節點。

步驟

1. 對於內部部署 ONTAP、您可以選擇性地建立 SMB 共用區、或 Trident 可以為您建立一個。

   > Amazon FSx for ONTAP 需要 SMB 共用。

   您可以透過兩種方式建立 SMB 管理共用：使用 "Microsoft Management Console" 共用資料夾嵌入式管理單元或使用 ONTAP CLI。若要使用 ONTAP CLI 建立 SMB 共用：

   a. 如有必要、請建立共用區的目錄路徑結構。

      此 `vserver cifs share create`指令會檢查在建立共用時透過 -path 選項指定的路徑。如果指定的路徑不存在，則命令執行失敗。

   b. 建立與指定 SVM 相關聯的 SMB 共用：

```
vserver cifs share create -vserver vserver_name -share-name
share_name -path path [-share-properties share_properties,...]
[other_attributes] [-comment text]
```

   c. 確認共用已建立：

```
vserver cifs share show -share-name share_name
```

詳情請參閱 "建立 SMB 共用區"。

2. 建立後端時，必須配置以下內容以指定 SMB 磁碟區。有關所有 FSx for ONTAP 後端設定選項，請參閱 "FSx for ONTAP 設定選項和範例"。

| 參數 | 說明 | 範例 |
|---|---|---|
| `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` |
| `nasType` | * 必須設為 `smb`。*如果為 null ，則預設為 `nfs`。 | `smb` |
| `securityStyle` | 新磁碟區的安全樣式。對於 **SMB** 磁碟區，必須設定為 **`ntfs`**`或 `**`mixed`**。 | `ntfs` 或 `mixed` 適用於 SMB 磁碟區 |
| `unixPermissions` | 新磁碟區的模式。**SMB** 磁碟區必須保留空白。 | "" |

# 設定和管理後端

## 配置後端

後端定義了 Trident 與儲存系統之間的關係。它告訴 Trident 如何與該儲存系統通訊，以及 Trident 應該如何從中配置磁碟區。

Trident 會自動從後端提供符合儲存類別定義要求的儲存資源池。瞭解如何為您的儲存系統設定後端。

- "配置 Azure NetApp Files 後端"
- "配置 Google Cloud NetApp Volumes 後端"
- "設定 NetApp HCI 或 SolidFire 後端"
- "使用 ONTAP 或 Cloud Volumes ONTAP NAS 驅動程式設定後端"
- "使用 ONTAP 或 Cloud Volumes ONTAP SAN 驅動程式設定後端"
- "將 Trident 與 Amazon FSx for NetApp ONTAP 搭配使用"

### Azure NetApp Files

#### 配置 **Azure NetApp Files** 後端

您可以將 Azure NetApp Files 設定為 Trident 的後端。您可以使用 Azure NetApp Files 後端附加 NFS 和 SMB 磁碟區。Trident 也支援使用託管識別碼對 Azure Kubernetes

Services (AKS) 叢集進行憑證管理。

#### Azure NetApp Files 驅動程式詳細資料

Trident 提供以下 Azure NetApp Files 儲存驅動程式以與叢集通訊。支援的存取模式有：*ReadWriteOnce* (RWO)、*ReadOnlyMany* (ROX)、*ReadWriteMany* (RWX)、*ReadWriteOncePod* (RWOP)。

| 驅動程式 | 傳輸協定 | **volumeMode** | 支援的存取模式 | 支援的檔案系統 |
|---|---|---|---|---|
| `azure-netapp-files` | NFS SMB | 檔案系統 | RWO 、 ROX 、 RWX 、 RWOP | `nfs, smb` |

考量事項

- Azure NetApp Files 服務不支援小於 50 GiB 的磁碟區。如果請求的磁碟區較小，Trident 會自動建立 50 GiB 的磁碟區。
- Trident 僅支援掛載到在 Windows 節點上執行的 Pod 的 SMB 磁碟區。

#### AKS 的受管理身分識別

Trident 支援 "託管身分識別" Azure Kubernetes Services 叢集。若要利用託管識別碼提供的簡化憑證管理功能，您必須具備以下條件：

- 使用 AKS 部署的 Kubernetes 叢集
- 在 AKS Kubernetes 叢集上設定的託管身分
- 已安裝的 Trident 包括 `cloudProvider`以指定 `"Azure"`。

**Trident 操作程式**

若要使用 Trident 運算子安裝 Trident，請編輯 `tridentorchestrator_cr.yaml`以將 `cloudProvider`設定為 `"Azure"`。例如：

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

**Helm**

以下範例會使用環境變數 `$CP`將 Trident sets `cloudProvider`安裝到 Azure：

```
helm install trident trident-operator-100.2506.0.tgz --create
-namespace --namespace <trident-namespace> --set cloudProvider=$CP
```

**<code>tridentctl</code>**

以下範例安裝 Trident 並將 `cloudProvider` 標誌設為 `Azure`：

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

#### AKS 的雲端身分

雲端身分使 Kubernetes Pod 能夠透過作為工作負載身分進行驗證來存取 Azure 資源，而無需提供明確的 Azure 認證。

若要在 Azure 中利用雲端身分功能，您必須具備以下條件：

- 使用 AKS 部署的 Kubernetes 叢集
- 在 AKS Kubernetes 叢集上設定 Workload identity 和 oidc-issuer
- 已安裝的 Trident 包括 `cloudProvider`以指定 `"Azure"`和 `cloudIdentity`指定工作負載身分

**Trident 操作程式**

若要使用 Trident 操作員安裝 Trident，請編輯 `tridentorchestrator_cr.yaml`以將 `cloudProvider`設定為 `"Azure"`，並將 `cloudIdentity`設定為 `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx`。

例如：

```
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-xxxxxxxxxxx' # 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'"
```

以下範例會安裝 Trident，並使用環境變數 `cloudProvider`將 `$CP`設為 `Azure`，同時使用環境變數 `cloudIdentity`設定 `$CI`：

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

**<code>tridentctl</code>**

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` 標誌設為 `$CP`，並將 `cloud-identity` 設定為 `$CI`：

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

# 在配置 Azure NetApp Files 後端之前，需要確保滿足以下要求。

### **NFS** 和 **SMB** 磁碟區的先決條件

如果您是首次使用 Azure NetApp Files 或在新的位置使用，則需要進行一些初始設定來設定 Azure NetApp Files 並建立 NFS 磁碟區。請參閱 "Azure：設定 Azure NetApp Files 並建立 NFS Volume"。

要設定和使用 "Azure NetApp Files" 後端、您需要以下元件：

> - 在 AKS 叢集上使用託管識別時，`subscriptionID`、 `tenantID`、 `clientID`、 `location` 和 `clientSecret` 是可選的。
> - 在 AKS 叢集上使用雲端身分時，`tenantID`、 `clientID` 和 `clientSecret` 是可選的。

- 容量池。請參閱 "Microsoft：為 Azure NetApp Files 建立容量池"。
- 委派給 Azure NetApp Files 的子網路。請參閱 "Microsoft ：將子網路委派給 Azure NetApp Files"。
- `subscriptionID` 來自已啟用 Azure NetApp Files 的 Azure 訂閱。
- `tenantID`、 `clientID` 和 `clientSecret` 來自 "應用程式註冊" Azure Active Directory 中具有足夠權限存取 Azure NetApp Files 服務的帳戶。應用程式註冊應使用以下任一方式：
  - 擁有者或貢獻者角色"由 Azure 預先定義"。
  - "自訂 Contributor 角色" 在訂閱層級 (`assignableScopes` 建立自訂角色，該角色擁有下列權限，這些權限僅限於 Trident 所需的權限。建立自訂角色後，"使用 Azure 入口網站指派角色"。

自訂貢獻者角色

```
{
  "id": "/subscriptions/<subscription-
id>/providers/Microsoft.Authorization/roleDefinitions/<role-
definition-id>",
  "properties": {
    "roleName": "custom-role-with-limited-perms",
    "description": "custom role providing limited permissions",
    "assignableScopes": [
      "/subscriptions/<subscription-id>"
    ],
    "permissions": [
      {
        "actions": [
          "Microsoft.NetApp/netAppAccounts/capacityPools/read",
          "Microsoft.NetApp/netAppAccounts/capacityPools/write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/read",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/delete",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
read",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
write",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/snapshots/
delete",

"Microsoft.NetApp/netAppAccounts/capacityPools/volumes/MountTarge
ts/read",
          "Microsoft.Network/virtualNetworks/read",
          "Microsoft.Network/virtualNetworks/subnets/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/read",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
ions/write",

"Microsoft.Features/featureProviders/subscriptionFeatureRegistrat
```

```
ions/delete",
          "Microsoft.Features/features/read",
          "Microsoft.Features/operations/read",
          "Microsoft.Features/providers/features/read",

"Microsoft.Features/providers/features/register/action",

"Microsoft.Features/providers/features/unregister/action",

"Microsoft.Features/subscriptionFeatureRegistrations/read"
        ],
        "notActions": [],
        "dataActions": [],
        "notDataActions": []
      }
    ]
  }
}
```

- 包含至少一個 `location`的 Azure "委派子網路"。自 Trident 22.01 起， `location 參數是後端組態檔頂層的必填欄位。在虛擬資源池中指定的位置值將被忽略。
- 若要使用 Cloud Identity，請從 "使用者指派的受控身分"取得 client ID，並在 `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx`中指定該 ID。

**SMB** 磁碟區的其他需求

若要建立 SMB Volume ，您必須具備以下條件：

- Active Directory 已設定並連線至 Azure NetApp Files 。請參閱 "Microsoft：建立和管理 Azure NetApp Files 的 Active Directory 連線" 。
- Kubernetes 叢集包含一個 Linux 控制器節點和至少一個執行 Windows Server 2022 的 Windows 工作節點。Trident 僅支援掛載到在 Windows 節點上執行的 pod 的 SMB 磁碟區。
- 至少需要一個包含 Active Directory 憑證的 Trident 金鑰，以便 Azure NetApp Files 可以向 Active Directory 進行驗證。要產生金鑰 smbcreds：

```
kubectl create secret generic smbcreds --from-literal username=user
--from-literal password='password'
```

- CSI Proxy 設定為 Windows 服務。若要設定 csi-proxy，請參閱"GitHub：CSI Proxy"或"GitHub：適用於 Windows 的 CSI Proxy"以瞭解在 Windows 上執行的 Kubernetes 節點。

**Azure NetApp Files** 後端組態選項和範例

了解 Azure NetApp Files 的 NFS 和 SMB 後端設定選項，並查看設定範例。

後端組態選項

Trident 使用您的後端設定（子網路、虛擬網路、服務等級和位置），在要求的位置中可用的容量池上建立 Azure NetApp Files 磁碟區，並與要求的服務等級和子網路相符。

Azure NetApp Files 後端提供以下設定選項。

| 參數 | 說明 | 預設 |
| --- | --- | --- |
| `version` | | 始終為 1 |
| `storageDriverName` | 儲存驅動程式的名稱 | "azure-netapp-files" |
| `backendName` | 自訂名稱或儲存後端 | 驅動程式名稱 + "_" + 隨機字元 |
| `subscriptionID` | Azure 訂閱的訂閱 ID（在 AKS 叢集上啟用託管識別時為選用）。 | |
| `tenantID` | 在 AKS 叢集上使用託管身分或雲端身分時，來自應用程式註冊的租用戶 ID 為選用項目。 | |
| `clientID` | 在 AKS 叢集上使用託管身分或雲端身分時，來自 App Registration 的用戶端 ID 為選用項目。 | |
| `clientSecret` | 在 AKS 叢集上使用託管身分或雲端身分時，來自應用程式註冊的用戶端密碼為選用項目。 | |
| `serviceLevel` | 其中之一 `Standard`、 `Premium` 或 `Ultra` | "（隨機） |
| `location` | 將在其中建立新磁碟區的 Azure 位置名稱在 AKS 叢集上啟用託管身分識別時為選用。 | |
| `resourceGroups` | 用於篩選已發現資源的資源群組清單 | "[]"（無濾鏡） |
| `netappAccounts` | 用於篩選已發現資源的 NetApp 帳戶列表 | "[]"（無濾鏡） |
| `capacityPools` | 用於篩選已發現資源的容量集區清單 | "[]"（無過濾，隨機） |
| `virtualNetwork` | 具有委派子網路的虛擬網路名稱 | "" |
| `subnet` | 委派給 `Microsoft.Netapp/volumes` 的子網路名稱 | "" |
| `networkFeatures` | 磁碟區的 VNet 功能集，可以是 `Basic`或 `Standard`。網路功能並非在所有區域都可用，可能需要在訂閱中啟用。指定 `networkFeatures`時，如果未啟用此功能，會導致磁碟區佈建失敗。 | "" |

| 參數 | 說明 | 預設 |
| --- | --- | --- |
| `nfsMountOptions` | 對 NFS 掛載選項進行精細控制。SMB 卷將忽略此設定。若要使用 NFS 版本 4.1 掛載卷，請在以逗號分隔的掛載選項清單中新增 `nfsvers=4` 以選擇 NFS v4.1。在儲存類別定義中設定的掛載選項會覆蓋後端配置中設定的掛載選項。 | "nfsvers=3" |
| `limitVolumeSize` | 如果請求的磁碟區大小大於此值，則配置失敗 | " （預設不強制執行） |
| `debugTraceFlags` | 疑難排解時使用的偵錯旗標。例如 `\{"api": false, "method": true, "discovery": true}`。除非您正在進行疑難排解並需要詳細的記錄傾印，否則請勿使用此功能。 | 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。 | `nfs` |
| `supportedTopologies` | 表示此後端支援的區域和區域清單。如需詳細資訊，請參閱 "使用 CSI 拓撲"。 | |
| `qosType` | 表示 QoS 類型：自動或手動。 | 自動 |
| `maxThroughput` | 設定允許的最大處理量（單位：MiB/ 秒）。僅支援手動 QoS 容量集區。 | `4 MiB/sec` |

如需網路功能的詳細資訊，請參閱 "設定 Azure NetApp Files 磁碟區的網路功能"。

所需權限和資源

如果在建立 PVC 時收到「未找到容量池」錯誤，則可能是您的應用程式註冊缺少所需的權限和資源（子網路、虛擬網路、容量池）。如果啟用了偵錯模式，Trident 會在建立後端時記錄發現的 Azure 資源。請確認是否使用了適當的角色。

`resourceGroups`、`netappAccounts`、`capacityPools`、`virtualNetwork` 和 `subnet`
的值可以使用短名稱或完全限定名稱來指定。大多數情況下建議使用完全限定名稱，因為短名稱可能會符合多個同名資源。

如果 vNet 位於與 Azure NetApp Files（ANF）儲存帳戶不同的資源群組中，則在設定後端的 resourceGroups 清單時，請為虛擬網路指定資源群組。

`resourceGroups`、 `netappAccounts` 和 `capacityPools`
值是過濾器，用於將發現的資源集限制為此儲存後端可用的資源，並且可以以任意組合指定。完全限定名稱遵循以下格式：

| 類型 | 格式 |
| --- | --- |
| 資源群組 | <resource group> |
| NetApp 帳戶 | <resource group>/<netapp account> |
| 容量池 | <resource group>/<netapp account>/<capacity pool> |
| 虛擬網路 | <resource group>/<virtual network> |
| 子網路 | <resource group>/<virtual network>/<subnet> |

**Volume** 資源配置

您可以透過在設定檔特定部分中指定以下選項來控制預設磁碟區配置。詳情請參閱 [範例組態]。

| 參數 | 說明 | 預設 |
| --- | --- | --- |
| `exportRule` | 新磁碟區的匯出規則。 `exportRule` 必須是以逗號分隔的 IPv4 位址或 CIDR 表示法的 IPv4 子網路的任意組合清單。SMB 磁碟區會忽略此規則。 | "0.0.0.0/0" |
| `snapshotDir` | 控制 .snapshot 目錄的可見性 | NFSv4 為 "true" ， NFSv3 為 "false" |
| `size` | 新磁碟區的預設大小 | "100G" |
| `unixPermissions` | 新磁碟區的 Unix 權限（4 位八進位數字）。SMB 磁碟區忽略此設定。 | " （預覽功能，需在訂閱中加入白名單） |

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

這是最基本的後端配置。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`省略，因此 `nfs`預設值將套用，後端將為 NFS 磁碟區進行佈建。

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
```

**AKS** 的雲端身分

此後端配置省略了 `tenantID`、 `clientID`和 `clientSecret`，在使用雲端身分時它們是可選的。

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-anf-1
  namespace: trident
spec:
  version: 1
  storageDriverName: azure-netapp-files
  capacityPools:
    - ultra-pool
  resourceGroups:
    - aks-ami-eastus-rg
  netappAccounts:
    - smb-na
  virtualNetwork: eastus-prod-vnet
  subnet: eastus-anf-subnet
  location: eastus
  subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
```

具有容量資源池篩選器的特定服務等級組態

此後端組態會將磁碟區放置在 Azure 的 `eastus`位置中的 `Ultra`容量集區。Trident 會自動探索該位置中委派給 Azure NetApp Files 的所有子網路，並隨機將新磁碟區放置在其中一個子網路上。

```
---
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
```

後端範例，包含手動 **QoS** 容量集區

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
```

進階組態

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"
```

虛擬資源池組態

此後端配置在單一檔案中定義了多個儲存池。當您有多個容量池支援不同的服務級別，並且希望在 Kubernetes 中建立代表這些容量池的儲存類別時，此配置非常有用。虛擬池標籤用於根據 `performance` 來區分這些池。

```
---
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
resourceGroups:
  - application-group-1
networkFeatures: Basic
nfsMountOptions: vers=3,proto=tcp,timeo=600
labels:
  cloud: azure
storage:
  - labels:
      performance: gold
    serviceLevel: Ultra
    capacityPools:
      - application-group-1/netapp-account-1/ultra-1
      - application-group-1/netapp-account-1/ultra-2
    networkFeatures: Standard
  - labels:
      performance: silver
    serviceLevel: Premium
    capacityPools:
      - application-group-1/netapp-account-1/premium-1
  - labels:
      performance: bronze
    serviceLevel: Standard
    capacityPools:
      - application-group-1/netapp-account-1/standard-1
      - application-group-1/netapp-account-1/standard-2
```

支援的拓撲組態

Trident 能夠根據區域和可用區為工作負載配置磁碟區。 `supportedTopologies`此後端配置中的程式碼區塊用於為每個後端提供區域和可用區清單。此處指定的區域和可用區值必須與每個 Kubernetes 叢集節點標籤中的區域和可用區值相符。這些區域和可用區代表儲存類別中可以提供的允許值清單。對於包含後端提供的區域和可用區子集的儲存類， Trident 會在指定的區域和可用區中建立磁碟區。如需詳細資訊，請參閱"使用 CSI 拓撲"。

```
---
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
supportedTopologies:
  - topology.kubernetes.io/region: eastus
    topology.kubernetes.io/zone: eastus-1
  - topology.kubernetes.io/region: eastus
    topology.kubernetes.io/zone: eastus-2
```

儲存類別定義

以下 `StorageClass`定義是指上述儲存資源池。

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池。該磁碟區將具有所選池中定義的屬性。

```
---
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。

預設命名空間上的基本組態

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
```

每個命名空間使用不同的機密

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

每個磁碟區使用不同的機密

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: anf-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "azure-netapp-files"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: ${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

`nasType: smb` 篩選支援 SMB 磁碟區的儲存池。 `nasType: nfs` 或 `nasType: null` 篩選 NFS 儲存池。

建立後端

建立後端組態檔後，執行以下命令：

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

如果後端建立失敗，則表示後端組態有問題。您可以執行下列命令來檢視記錄以判斷原因：

```
tridentctl logs
```

在您識別並修正組態檔的問題後、您可以再次執行 create 命令。

## Google Cloud NetApp Volumes

### 配置 **Google Cloud NetApp Volumes** 後端

現在您可以將 Google Cloud NetApp Volumes 設定為 Trident 的後端。您可以使用 Google Cloud NetApp Volumes 後端附加 NFS 和 SMB 磁碟區。

#### **Google Cloud NetApp Volumes** 驅動程式詳情

Trident 提供 `google-cloud-netapp-volumes`驅動程式來與叢集通訊。支援的存取模式包括：*ReadWriteOnce*（RWO）、*ReadOnlyMany*（ROX）、*ReadWriteMany*（RWX）、*ReadWriteOncePod*（RWOP）。

| 驅動程式 | 傳輸協定 | **volumeMode** | 支援的存取模式 | 支援的檔案系統 |
|---|---|---|---|---|
| `google-cloud-netapp-volumes` | NFS SMB | 檔案系統 | RWO 、 ROX 、 RWX 、 RWOP | `nfs`, `smb` |

#### **GKE** 的雲端身分

雲端身分可讓 Kubernetes Pod 透過以工作負載身分進行驗證來存取 Google Cloud 資源，而無需提供明確的 Google Cloud 認證。

若要在 Google Cloud 中利用雲端身分功能，您必須具備以下條件：

- 使用 GKE 部署的 Kubernetes 叢集。
- 在 GKE 叢集上配置工作負載身分，並在節點池上配置 GKE MetaData Server。
- 具有 Google Cloud NetApp Volumes 管理員（roles/netapp.admin）角色或自訂角色的 GCP 服務帳戶。
- Trident 已安裝，其中包括 cloudProvider 指定「GCP」並 cloudIdentity 指定新的 GCP 服務帳戶。以下提供範例。

**Trident 操作程式**

若要使用 Trident 操作員安裝 Trident，請編輯 `tridentorchestrator_cr.yaml`以將 `cloudProvider`設定為 `"GCP"`，並將 `cloudIdentity`設定為 `iam.gke.io/gcp-service-account: cloudvolumes-admin-sa@mygcpproject.iam.gserviceaccount.com`。

例如：

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  imagePullPolicy: IfNotPresent
  cloudProvider: "GCP"
  cloudIdentity: 'iam.gke.io/gcp-service-account: cloudvolumes-admin-sa@mygcpproject.iam.gserviceaccount.com'
```

**Helm**

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

```
export CP="GCP"
export ANNOTATION="'iam.gke.io/gcp-service-account: cloudvolumes-admin-sa@mygcpproject.iam.gserviceaccount.com'"
```

以下範例安裝 Trident 並使用環境變數 `$CP`將 `cloudProvider`設為 `GCP`，並使用環境變數 `$ANNOTATION`設置 `cloudIdentity`：

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

**<code>tridentctl</code>**

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

```
export CP="GCP"
export ANNOTATION="'iam.gke.io/gcp-service-account: cloudvolumes-admin-sa@mygcpproject.iam.gserviceaccount.com'"
```

以下範例安裝 Trident 並將 `cloud-provider` 標誌設為 `$CP`，並將 `cloud-identity` 設定為 `$ANNOTATION`：

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

準備設定 **Google Cloud NetApp Volumes** 後端

## 在配置 Google Cloud NetApp Volumes 後端之前，您需要確保滿足以下要求。

**NFS** 磁碟區的先決條件

如果您是首次使用 Google Cloud NetApp Volumes 或在新的位置使用，則需要進行一些初始配置來設定 Google Cloud NetApp Volumes 並建立 NFS 磁碟區。請參閱 "開始之前"。

在設定 Google Cloud NetApp Volumes 後端之前，請確保您已具備以下條件：

- 已設定 Google Cloud NetApp Volumes 服務的 Google Cloud 帳戶。請參閱 "Google Cloud NetApp Volumes"。
- 您的 Google Cloud 帳戶的專案編號。請參閱 "識別專案"。
- 具有 NetApp Volumes Admin (`roles/netapp.admin` 角色的 Google Cloud 服務帳號。請參閱 "身分與存取管理角色和權限"。
- 您的 GCNV 帳戶的 API 金鑰檔案。請參閱 "建立服務帳戶金鑰"
- 儲存池。請參閱 "儲存池概覽"。

有關如何設定對 Google Cloud NetApp Volumes 存取權限的詳細資訊、請參閱 "設定對 Google Cloud NetApp Volumes 的存取權限"。

**Google Cloud NetApp Volumes** 後端設定選項和範例

## 了解 Google Cloud NetApp Volumes 的後端組態選項並檢閱組態範例。

後端組態選項

每個後端都在單一 Google Cloud 區域中配置磁碟區。若要在其他區域中建立磁碟區，您可以定義其他後端。

| 參數 | 說明 | 預設 |
|---|---|---|
| `version` | | 始終為 1 |
| `storageDriverName` | 儲存驅動程式的名稱 | `storageDriverName` 的值必須指定為「google-cloud-netapp-volumes」。 |
| `backendName` | （選用）儲存後端的自訂名稱 | 驅動程式名稱 "_" API 金鑰的一部分 |
| `storagePools` | 用於指定磁碟區建立之儲存池的選用參數。 | |
| `projectNumber` | Google Cloud 帳戶專案編號。該值可在 Google Cloud 入口網站首頁找到。 | |

| 參數 | 說明 | 預設 |
| --- | --- | --- |
| location | Trident 建立 GCNV 磁碟區的 Google Cloud 位置。建立跨區域 Kubernetes 叢集時，在 `location`中建立的磁碟區可用於調度到多個 Google Cloud 區域節點上的工作負載。跨區域流量會產生額外費用。 | |
| apiKey | 用於具有 netapp.admin`角色的 Google Cloud 服務帳號的 API 金鑰。它包含 Google Cloud 服務帳號私鑰檔案的 JSON 格式內容（原封不動地複製到後端設定檔中）。 `apiKey`必須包含以下鍵的鍵值對： `type、project_id、client_email、client_id、auth_uri、token_uri、auth_provider_x509_cert_url`和 `client_x509_cert_url。 | |
| nfsMountOptions | 對 NFS 掛載選項進行精細控制。 | "nfsvers=3" |
| 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） |
| serviceLevel | 儲存池及其磁碟區的服務等級。取值為 flex、standard、premium`或 `extreme。 | |
| labels | 要套用於磁碟區的任意 JSON 格式標籤集 | "" |
| network | 用於 Google Cloud NetApp Volumes 磁碟區的 Google Cloud 網路。 | |
| debugTraceFlags | 疑難排解時使用的偵錯旗標。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，否則請勿使用此功能。 | 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。 | nfs |
| supportedTopologies | 表示此後端支援的區域和可用區列表。如需更多資訊，請參閱"使用 CSI 拓撲"。例如：<br>supportedTopologies:<br>- topology.kubernetes.io/region: asia-east1<br>topology.kubernetes.io/zone: asia-east1-a | |

**Volume** 配置選項

您可以在設定檔的 defaults 區段中控制預設磁碟區配置。

| 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 | NFSv4 為 "true" ， NFSv3 為 "false" |
| snapshotReserve | 為快照保留的磁碟區百分比 | ""（接受預設值 0） |

| 參數 | 說明 | 預設 |
| --- | --- | --- |
| `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，這種配置是理想的，但在實踐中，您很可能需要為配置的磁碟區提供額外的範圍。

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  serviceLevel: premium
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

**SMB 磁碟區配置**

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv1
  namespace: trident
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123456789"
  location: asia-east1
  serviceLevel: flex
  nasType: smb
  apiKey:
    type: service_account
    project_id: cloud-native-data
    client_email: trident-sample@cloud-native-
data.iam.gserviceaccount.com
    client_id: "123456789737813416734"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/trident-
sample%40cloud-native-data.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

使用 **StoragePools** 篩選器進行組態設定

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  serviceLevel: premium
  storagePools:
    - premium-pool1-europe-west6
    - premium-pool2-europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
```

虛擬資源池組態

此後端組態在單一檔案中定義了多個虛擬資源池。虛擬資源池在 `storage`區段中定義。當您有多個支援不同服務層級的儲存資源池、並想在 Kubernetes 中建立代表這些資源池的儲存類別時、虛擬資源池非常實用。虛擬資源池標籤用於區分資源池。例如、在以下範例中、 `performance`標籤和 `serviceLevel`類型用於區分虛擬資源池。

您也可以設定一些適用於所有虛擬池的預設值，並覆寫各個虛擬池的預設值。在下列範例中，`snapshotReserve` 和 `exportRule` 做為所有虛擬池的預設值。

如需更多資訊，請參閱 "虛擬資源池"。

```
---
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-gcnv-secret
type: Opaque
stringData:
  private_key_id: f2cb6ed6d7cc10c453f7d3406fc700c5df0ab9ec
  private_key: |
    -----BEGIN PRIVATE KEY-----
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    znHczZsrrtHisIsAbOguSaPIKeyAZNchRAGzlzZE4jK3bl/qp8B4Kws8zX5ojY9m
    XsYg6gyxy4zq7OlwWgLwGa==
    -----END PRIVATE KEY-----

---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: "123455380079"
  location: europe-west6
  apiKey:
    type: service_account
    project_id: my-gcnv-project
    client_email: myproject-prod@my-gcnv-
project.iam.gserviceaccount.com
    client_id: "103346282737811234567"
    auth_uri: https://accounts.google.com/o/oauth2/auth
    token_uri: https://oauth2.googleapis.com/token
```

```
    auth_provider_x509_cert_url:
https://www.googleapis.com/oauth2/v1/certs
    client_x509_cert_url:
https://www.googleapis.com/robot/v1/metadata/x509/myproject-prod%40my-
gcnv-project.iam.gserviceaccount.com
  credentials:
    name: backend-tbc-gcnv-secret
  defaults:
    snapshotReserve: "10"
    exportRule: 10.0.0.0/24
  storage:
    - labels:
        performance: extreme
      serviceLevel: extreme
      defaults:
        snapshotReserve: "5"
        exportRule: 0.0.0.0/0
    - labels:
        performance: premium
      serviceLevel: premium
    - labels:
        performance: standard
      serviceLevel: standard
```

**GKE 的雲端身分**

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-gcp-gcnv
spec:
  version: 1
  storageDriverName: google-cloud-netapp-volumes
  projectNumber: '012345678901'
  network: gcnv-network
  location: us-west2
  serviceLevel: Premium
  storagePool: pool-premium1
```

支援的拓撲組態

Trident 能夠根據區域和可用區為工作負載配置磁碟區。 `supportedTopologies`此後端配置中的程式碼區塊用於為每個後端提供區域和可用區清單。此處指定的區域和可用區值必須與每個 Kubernetes 叢集節點標籤中的區域和可用區值相符。這些區域和可用區代表儲存類別中可以提供的允許值清單。對於包含後端提供的區域和可用區子集的儲存類， Trident 會在指定的區域和可用區中建立磁碟區。如需詳細資訊，請參閱"使用 CSI 拓撲"。

```
---
version: 1
storageDriverName: google-cloud-netapp-volumes
subscriptionID: 9f87c765-4774-fake-ae98-a721add45451
tenantID: 68e4f836-edc1-fake-bff9-b2d865ee56cf
clientID: dd043f63-bf8e-fake-8076-8de91e5713aa
clientSecret: SECRET
location: asia-east1
serviceLevel: flex
supportedTopologies:
  - topology.kubernetes.io/region: asia-east1
    topology.kubernetes.io/zone: asia-east1-a
  - topology.kubernetes.io/region: asia-east1
    topology.kubernetes.io/zone: asia-east1-b
```

接下來呢？

建立後端組態檔後，執行以下命令：

```
kubectl create -f <backend-file>
```

若要驗證後端是否已成功建立，請執行下列命令：

```
kubectl get tridentbackendconfig

NAME                BACKEND NAME        BACKEND UUID
PHASE   STATUS
backend-tbc-gcnv    backend-tbc-gcnv    b2fd1ff9-b234-477e-88fd-713913294f65
Bound   Success
```

如果後端建立失敗，則表示後端配置存在問題。您可以使用 `kubectl get tridentbackendconfig <backend-name>` 命令描述後端，或執行以下命令查看日誌以確定原因：

```
tridentctl logs
```

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。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: extreme-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=extreme
  backendType: google-cloud-netapp-volumes

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: premium-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=premium
  backendType: google-cloud-netapp-volumes

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: standard-sc
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=standard
  backendType: google-cloud-netapp-volumes
```

如需儲存類別的詳細資訊，請參閱 "建立儲存類別"。

**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。任何具有任意權限或無權限的 Active Directory 使用者名稱/密碼均可用作節點階段金鑰。

預設命名空間上的基本組態

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
```

每個命名空間使用不同的機密

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: "smbcreds"
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

每個磁碟區使用不同的機密

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: gcnv-sc-smb
provisioner: csi.trident.netapp.io
parameters:
  backendType: "google-cloud-netapp-volumes"
  trident.netapp.io/nasType: "smb"
  csi.storage.k8s.io/node-stage-secret-name: ${pvc.name}
  csi.storage.k8s.io/node-stage-secret-namespace: ${pvc.namespace}
```

`nasType: smb` 篩選支援 SMB 磁碟區的儲存池。 `nasType: nfs` 或 `nasType: null` 篩選 NFS 儲存池。

**PVC** 定義範例

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: gcnv-nfs-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 100Gi
  storageClassName: gcnv-nfs-sc
```

若要驗證 PVC 是否已繫結，請執行下列命令：

```
kubectl get pvc gcnv-nfs-pvc

NAME           STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS AGE
gcnv-nfs-pvc   Bound    pvc-b00f2414-e229-40e6-9b16-ee03eb79a213   100Gi
RWX            gcnv-nfs-sc  1m
```

## 設定 NetApp HCI 或 SolidFire 後端

了解如何在 Trident 安裝中建立和使用 Element 後端。

**Element** 驅動程式詳細資料

Trident 提供 `solidfire-san` 儲存驅動程式以與叢集通訊。支援的存取模式有：*ReadWriteOnce* (RWO)、*ReadOnlyMany* (ROX)、*ReadWriteMany* (RWX)、*ReadWriteOncePod* (RWOP)。

`solidfire-san` 儲存驅動程式支援_檔案_和_區塊_磁碟區模式。對於 `Filesystem` volumeMode，Trident 會建立磁碟區並建立檔案系統。檔案系統類型由 StorageClass 指定。

| 驅動程式 | 傳輸協定 | VolumeMode | 支援的存取模式 | 支援的檔案系統 |
|---|---|---|---|---|
| solidfire-san | iSCSI | 區塊 | RWO 、 ROX 、 RWX 、 RWOP | 無檔案系統。原始區塊裝置。 |

| 驅動程式 | 傳輸協定 | VolumeMode | 支援的存取模式 | 支援的檔案系統 |
|---|---|---|---|---|
| solidfire-san | iSCSI | 檔案系統 | RWO、RWOP | xfs, ext3, ext4 |

開始之前

在建立 Element 後端之前、您需要下列項目。

- 執行 Element 軟體的受支援儲存系統。
- NetApp HCI/SolidFire 叢集管理員或租用戶使用者的憑證，可管理磁碟區。
- 所有 Kubernetes 工作節點都應安裝對應的 iSCSI 工具。請參閱"工作節點準備資訊"。

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 |
| storageDriverName | 儲存驅動程式的名稱 | 始終是 "solidfire-san" |
| backendName | 自訂名稱或儲存後端 | "solidfire_" + 儲存設備（iSCSI）IP 位址 |
| Endpoint | 具有租戶認證資料的 SolidFire 叢集的 MVIP | |
| SVIP | 儲存設備（iSCSI）IP 位址和連接埠 | |
| labels | 要套用於磁碟區的任意 JSON 格式標籤集。 | "" |
| TenantName | 要使用的租戶名稱（如果找不到則建立） | |
| InitiatorIFace | 將 iSCSI 流量限制到特定主機介面 | "default" |
| UseCHAP | 使用 CHAP 協定對 iSCSI 進行身份驗證。Trident 就使用 CHAP 協定。 | true |
| AccessGroups | 要使用的存取群組 ID 清單 | 尋找名為「trident」的存取群組 ID |
| Types | QoS 規範 | |
| 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 | 疑難排解時使用的偵錯旗標。例如、{"api":false, "method":true} | null |

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。

範例 **1**： `solidfire-san` 驅動程式具有三種磁碟區類型的後端組態

此範例展示了一個使用 CHAP 驗證的後端文件，並對三種具有特定 QoS 保證的磁碟區類型進行了建模。您很可能需要使用 `IOPS`儲存類別參數來定義儲存類別，以便使用每種儲存類別。

```
---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0
SVIP: <svip>:3260
TenantName: <tenant>
labels:
  k8scluster: dev1
  backend: dev1-element-cluster
UseCHAP: true
Types:
  - Type: Bronze
    Qos:
      minIOPS: 1000
      maxIOPS: 2000
      burstIOPS: 4000
  - Type: Silver
    Qos:
      minIOPS: 4000
      maxIOPS: 6000
      burstIOPS: 8000
  - Type: Gold
    Qos:
      minIOPS: 6000
      maxIOPS: 8000
      burstIOPS: 10000
```

範例 **2**： `solidfire-san` 驅動程式搭配虛擬資源池的後端和儲存類別組態

此範例顯示了配置了虛擬池的後端定義檔以及參考這些虛擬池的 StorageClasses。

Trident 會在配置時將儲存資源池上的標籤複製到後端儲存 LUN。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，這些值將 `type`設定為 Silver。虛擬池在 `storage`區段中定義。在此範例中，部分儲存池設定了自己的類型，而部分儲存池則覆寫了上述預設值。

```
---
version: 1
storageDriverName: solidfire-san
Endpoint: https://<user>:<password>@<mvip>/json-rpc/8.0
```

```
SVIP: <svip>:3260
TenantName: <tenant>
UseCHAP: true
Types:
  - Type: Bronze
    Qos:
      minIOPS: 1000
      maxIOPS: 2000
      burstIOPS: 4000
  - Type: Silver
    Qos:
      minIOPS: 4000
      maxIOPS: 6000
      burstIOPS: 8000
  - Type: Gold
    Qos:
      minIOPS: 6000
      maxIOPS: 8000
      burstIOPS: 10000
type: Silver
labels:
  store: solidfire
  k8scluster: dev-1-cluster
region: us-east-1
storage:
  - labels:
      performance: gold
      cost: "4"
    zone: us-east-1a
    type: Gold
  - labels:
      performance: silver
      cost: "3"
    zone: us-east-1b
    type: Silver
  - labels:
      performance: bronze
      cost: "2"
    zone: us-east-1c
    type: Bronze
  - labels:
      performance: silver
      cost: "1"
    zone: us-east-1d
```

以下 StorageClass 定義均與上述虛擬池相關。使用 `parameters.selector` 欄位，每個 StorageClass 都會指

定可用於託管磁碟區的虛擬池。磁碟區將具有所選虛擬池中定義的屬性。

第一個 StorageClass (`solidfire-gold-four` 將對應到第一個虛擬儲存池。這是唯一提供黃金級效能且 `Volume Type QoS` 為 Gold 的儲存池。最後一個 StorageClass (`solidfire-silver` 會指定任何提供白銀級效能的儲存池。Trident 將決定選擇哪個虛擬儲存池，並確保滿足儲存需求。

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-gold-four
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=gold; cost=4
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-three
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver; cost=3
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-bronze-two
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=bronze; cost=2
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: solidfire-silver-one
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver; cost=1
  fsType: ext4

---
apiVersion: storage.k8s.io/v1
```

```
kind: StorageClass
metadata:
  name: solidfire-silver
provisioner: csi.trident.netapp.io
parameters:
  selector: performance=silver
  fsType: ext4
```

尋找更多資訊

- "Volume 存取群組"

## ONTAP SAN 驅動程式

### ONTAP SAN 驅動程式概述

了解如何使用 ONTAP 和 Cloud Volumes ONTAP SAN 驅動程式設定 ONTAP 後端。

#### ONTAP SAN 驅動程式詳細資料

Trident 提供以下 SAN 儲存驅動程式、用於與 ONTAP 叢集通訊。支援的存取模式包括：*ReadWriteOnce*（RWO）、*ReadOnlyMany*（ROX）、*ReadWriteMany*（RWX）、*ReadWriteOncePod*（RWOP）。

| 驅動程式 | 傳輸協定 | **volumeMode** | 支援的存取模式 | 支援的檔案系統 |
|---|---|---|---|---|
| `ontap-san` | iSCSI SCSI over FC | 區塊 | RWO 、 ROX 、 RWX 、 RWOP | 無檔案系統；原始區塊裝置 |
| `ontap-san` | iSCSI SCSI over FC | 檔案系統 | RWO、RWOP<br><br>ROX 和 RWX 在 Filesystem 磁碟區模式下不可用。 | `xfs`, `ext3`, `ext4` |
| `ontap-san` | NVMe/TCP<br><br>請參閱 NVMe/TCP 的其他考量事項。 | 區塊 | RWO 、 ROX 、 RWX 、 RWOP | 無檔案系統；原始區塊裝置 |
| `ontap-san` | NVMe/TCP<br><br>請參閱 NVMe/TCP 的其他考量事項。 | 檔案系統 | RWO、RWOP<br><br>ROX 和 RWX 在 Filesystem 磁碟區模式下不可用。 | `xfs`, `ext3`, `ext4` |

| 驅動程式 | 傳輸協定 | volumeMode | 支援的存取模式 | 支援的檔案系統 |
|---|---|---|---|---|
| ontap-san-economy | iSCSI | 區塊 | RWO 、 ROX 、 RWX 、 RWOP | 無檔案系統；原始區塊裝置 |
| ontap-san-economy | iSCSI | 檔案系統 | RWO、RWOP<br><br>ROX 和 RWX 在 Filesystem 磁碟區模式下不可用。 | xfs, ext3, ext4 |

- 僅當預期持久性磁碟區使用次數高於"支援的 ONTAP Volume 限制"時，才使用 ontap-san-economy。
- 僅當預期持久性磁碟區使用次數高於"支援的 ONTAP Volume 限制" 且無法使用 ontap-san-economy 驅動程式時，才使用 ontap-nas-economy。
- 如果您預計需要資料保護、災難復原或行動性，請勿使用 ontap-nas-economy。
- NetApp 除 ontap-san 外，不建議在所有 ONTAP 驅動程式中使用 Flexvol 自動增長功能。作為變通方案，Trident 支援使用快照預留，並相應地擴展 Flexvol 磁碟區。

使用者權限

Trident 需要以 ONTAP 或 SVM 管理員身分執行，通常使用 `admin`叢集使用者或 `vsadmin`SVM 使用者，或使用具有相同角色但名稱不同的使用者。對於 Amazon FSx for NetApp ONTAP 部署、Trident 需要以 ONTAP 或 SVM 管理員身分執行，使用叢集 `fsxadmin`使用者或 `vsadmin`SVM 使用者，或使用具有相同角色但名稱不同的使用者。 `fsxadmin`使用者是叢集管理使用者的有限替代方案。

如果使用 `limitAggregateUsage`參數，則需要叢集管理員權限。將 Amazon FSx for NetApp ONTAP 與 Trident 搭配使用時， `limitAggregateUsage`參數與 `vsadmin`和 `fsxadmin`使用者帳戶不相容。如果指定此參數，組態作業將失敗。

雖然可以在 ONTAP 中建立更嚴格的角色供 Trident 驅動程式使用，但我們不建議這樣做。大多數新版本的 Trident 都會呼叫額外的 API，這些 API 需要考慮，這會讓升級變得困難且容易出錯。

**NVMe/TCP** 的其他考量事項

Trident 支援非揮發性記憶體高速介面（NVMe）協定，使用 ontap-san 驅動程式，包括：

- IPv6
- NVMe 磁碟區的快照和複本
- 調整 NVMe Volume 的大小
- 導入在 Trident 外部建立的 NVMe 磁碟區，以便 Trident 管理其生命週期
- NVMe 原生多路徑
- K8s 節點的優雅關閉或非優雅關閉（24.06）

Trident 不支援：

- NVMe 原生支援的 DH-HMAC-CHAP
- 裝置對應程式（DM）多重路徑
- LUKS 加密

NVMe 僅支援 ONTAP REST API ，不支援 ONTAPI （ ZAPI ）。

準備使用 **ONTAP SAN** 驅動程式配置後端

# 了解使用 ONTAP SAN 驅動程式配置 ONTAP 後端的要求和驗證選項。

需求

對於所有 ONTAP 後端、 Trident 要求至少將一個 Aggregate 指派給 SVM 。

"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。在 ASA r2 系統中，使用儲存可用區而非 Aggregate。請參閱 "這" 知識庫文章，瞭解如何在 ASA r2 系統中將 Aggregate 指派給 SVM。

請記住，您還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，您可以設定一個 `san-dev`類別，使用 `ontap-san`驅動程式，以及一個 `san-default`類別，使用 `ontap-san-economy`驅動程式。

所有 Kubernetes 工作節點都必須安裝適當的 iSCSI 工具。詳情請參閱 "準備工作節點"。

驗證 **ONTAP** 後端

Trident 提供兩種 ONTAP 後端驗證模式。

- 基於憑證：具有所需權限的 ONTAP 使用者的使用者名稱和密碼。建議使用預先定義的安全登入角色，例如 `admin`或 `vsadmin`，以確保與 ONTAP 版本的最大相容性。
- 基於憑證：Trident 也可以使用安裝在後端的憑證與 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。

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-san
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: password
```

**JSON**

```
{
  "version": 1,
  "backendName": "ExampleBackend",
  "storageDriverName": "ontap-san",
  "managementLIF": "10.0.0.1",
  "svm": "svm_nfs",
  "username": "vsadmin",
  "password": "password"
}
```

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。建立或更新後端是唯一需要瞭解認證資料的步驟。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=admin"
```

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
```

> 執行此命令後，ONTAP 會提示輸入憑證。貼上步驟 1 中產生的 `k8senv.pem`檔案內容，然後輸入 `END`以完成安裝。

4. 確認 ONTAP 安全登入角色支援 `cert` 驗證方法。

```
security login create -user-or-group-name admin -application ontapi
-authentication-method cert
security login create -user-or-group-name admin -application http
-authentication-method cert
```

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。

```
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.json
{
"version": 1,
"storageDriverName": "ontap-san",
"backendName": "SanBackend",
"managementLIF": "1.2.3.4",
"svm": "vserver_test",
"clientCertificate": "Faaaakkkkeeee...Vaaalllluuuueeee",
"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 |
+------------+----------------+--------------------------------------
+--------+---------+
```

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"

使用雙向 **CHAP** 驗證連線

Trident 可以使用雙向 CHAP 對 iSCSI 工作階段進行驗證，適用於 `ontap-san`和 `ontap-san-economy`驅動程式。這需要在後端定義中啟用 `useCHAP`選項。當設定為 `true`時，Trident 會將 SVM 的預設啟動器安全性設定為雙向 CHAP，並從後端檔案設定使用者名稱和密碼。NetApp 建議使用雙向 CHAP 來驗證連線。請參閱以下範

例組態：

```
---
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: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
```

useCHAP 參數為布林值選項，只能配置一次。預設值為 false。將其設為 true 後，無法再將其設為 false。

除了 `useCHAP=true`之外， `chapInitiatorSecret`、 `chapTargetInitiatorSecret`、 `chapTargetUsername`和 `chapUsername`欄位也必須包含在後端定義中。後端建立完成後，可以透過執行 `tridentctl update`來變更金鑰。

運作方式

將 `useCHAP`設為 true 後，儲存管理員會指示 Trident 在儲存後端設定 CHAP。這包括以下內容：

- 在 SVM 上設定 CHAP：
  - 如果 SVM 的預設發起程序安全類型為 none（預設設定）*且*磁碟區中沒有預先存在的 LUN，則 Trident 會將預設安全類型設為 `CHAP`，並繼續設定 CHAP 發起程式和目標使用者名稱及金鑰。
  - 如果 SVM 包含 LUN，Trident 將不會在 SVM 上啟用 CHAP。這確保對 SVM 上已存在的 LUN 的存取不受限制。
- 配置 CHAP 啟動器和目標使用者名稱和密碼；這些選項必須在後端組態中指定（如上所示）。

後端建立完成後， Trident 會建立對應的 `tridentbackend` CRD ，並將 CHAP 金鑰和使用者名稱儲存為 Kubernetes 金鑰。Trident 在此後端建立的所有 PV 都將透過 CHAP 協定進行掛載和連線。

輪換認證資料並更新後端

您可以透過更新 `backend.json` 檔案中的 CHAP 參數來更新 CHAP 憑證。這需要更新 CHAP 金鑰，並使用 `tridentctl update` 命令來反映這些變更。

更新後端的 CHAP 密碼時，您必須使用 `tridentctl`來更新後端。請勿使用 ONTAP CLI 或 ONTAP System Manager 更新儲存叢集上的認證，因為 Trident 將無法識別這些變更。

```
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 |                 UUID                 |
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** 配置選項和範例

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）在儲存層的實作方面有所不同。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"。

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

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。

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 |
| `storageDriverName` | 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` | 叢集或 SVM 管理 LIF 的 IP 位址。<br><br>可以指定完全限定網域名稱（FQDN）。<br><br>如果 Trident 安裝時使用了 IPv6 標誌，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]`。<br><br>如需無縫 MetroCluster 切換、請參閱 MetroCluster 範例。<br><br>ⓘ 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。 | "10.0.0.1", "[2001:1234:abcd::fefe]" |
| `dataLIF` | 協定 LIF 的 IP 位址。如果 Trident 安裝時使用了 IPv6 標誌，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。 | 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。如需詳細資訊、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 | "" |
| `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。如果需要，則為必填項 `useCHAP=true` | "" |
| `chapUsername` | 入站使用者名稱。如果 `useCHAP=true`則為必填項 | "" |
| `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。用於基於憑證的驗證 | "" |
| `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。選用。用於憑證型驗證。 | "" |
| `username` | 與 ONTAP 叢集通訊所需的使用者名稱。用於基於認證的驗證。有關 Active Directory 驗證，請參閱 "使用 Active Directory 憑證對後端 SVM 進行 Trident 驗證"。 | "" |
| `password` | 與 ONTAP 叢集通訊所需的密碼。用於基於認證的驗證。有關 Active Directory 驗證，請參閱 "使用 Active Directory 憑證對後端 SVM 進行 Trident 驗證"。 | "" |
| `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。要更新此參數、您需要建立新的後端。 | `trident` |
| `aggregate` | 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。<br><br>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。<br><br>*請勿指定用於 ASA r2 系統 *。 | "" |

| 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 擷取 Aggregate 使用量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 | 如果要求的磁碟區大小超過此值，則資源配置失敗。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 | 使用 ONTAP REST API 的布林參數。<br>`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 或更新版本開始， `useREST`預設設定為 `true`；若要使用 ONTAPI（ZAPI）呼叫，請將 `useREST`變更為 `false`。<br>useREST 完全符合 NVMe/TCP 標準。<br>NVMe 僅支援 ONTAP REST API ，不支援 ONTAPI （ ZAPI ）。<br>如果指定、則一律設為 **true** （適用於 **ASA r2** 系統）。 | 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 |

| 參數 | 說明 | 預設 |
| --- | --- | --- |
| `formatOptions` | 使用 `formatOptions`為 `mkfs`命令指定命令列引數，這些引數將在每次格式化磁碟區時套用。這樣可讓您根據自己的偏好格式化磁碟區。請確保指定的 formatOptions 與 mkfs 命令選項類似，但不包含裝置路徑。範例："-E nodiscard"<br><br>支援使用 **iSCSI** 傳輸協定的 **`ontap-san`**和 **`ontap-san-economy`**驅動程式。*此外，使用 iSCSI 和 NVMe/TCP 傳輸協定時，也支援 ASA r2 系統。* | |
| `limitVolumePoolSize` | 在 ontap-san-economy 後端使用 LUN 時可請求的最大 FlexVol 大小。 | "（預設不強制執行） |
| `denyNewVolumePools` | 限制 `ontap-san-economy`後端建立新的 FlexVol 磁碟區以包含其 LUN 。僅使用預先存在的 Flexvol 來配置新的 PV 。 | |

使用 **formatOptions** 的建議

Trident 建議採用以下選項來加速格式化程序：

- **-E nodiscard (ext3, ext4):** 在執行 mkfs 時不要嘗試丟棄資料區塊（在固態裝置和稀疏 / 精簡配置儲存上、初始丟棄資料區塊很有用）。此選項取代了已棄用的「 -K 」選項、適用於 ext3 和 ext4 檔案系統。
- **-K (xfs):** 執行 mkfs 時不要嘗試丟棄資料區塊。此選項適用於 xfs 檔案系統。

使用 **Active Directory** 憑證對後端 **SVM** 進行 **Trident** 驗證

您可以設定 Trident 使用 Active Directory （ AD ）憑證對後端 SVM 進行驗證。在 AD 帳戶可以存取 SVM 之前、您必須設定 AD 網域控制站對叢集或 SVM 的存取權限。若要使用 AD 帳戶進行叢集管理、您必須建立網域通道。如需詳細資訊、請參閱 "在 ONTAP 中設定 Active Directory 網域控制器存取"。

步驟

1. 為後端 SVM 設定網域名稱系統（Domain Name System、DNS）設定：

   ```
   vserver services dns create -vserver <svm_name> -dns-servers
   <dns_server_ip1>,<dns_server_ip2>
   ```

2. 執行下列命令、在 Active Directory 中為 SVM 建立電腦帳戶：

   ```
   vserver active-directory create -vserver DataSVM -account-name ADSERVER1
   -domain demo.netapp.com
   ```

3. 使用此命令建立 AD 使用者或群組來管理叢集或 SVM

   ```
   security login create -vserver <svm_name> -user-or-group-name
   <ad_user_or_group> -application <application> -authentication-method domain
   -role vsadmin
   ```

4. 在 Trident 後端組態檔中，將 `username` 和 `password` 參數分別設定為 AD 使用者或群組名稱和密碼。

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` | LUN 的空間分配 | "true" 如果指定、請針對 **ASA r2** 系統設定為 **`true`**。 |
| `spaceReserve` | 空間保留模式；「 none 」（精簡）或「 volume 」（完整）。針對 **ASA r2** 系統設為 `none`。 | "none" |
| `snapshotPolicy` | 要使用的快照原則。針對 **ASA r2** 系統設定為 **`none`**。 | "none" |
| `qosPolicy` | 要為建立的磁碟區指派的 QoS 原則群組。每個儲存資源池/後端可選擇 qosPolicy 或 adaptiveQosPolicy 其中之一。搭配 Trident 使用 QoS 原則群組需要 ONTAP 9.8 或更新版本。您應該使用非共享的 QoS 原則群組，並確保該原則群組個別套用至每個成員。共享的 QoS 原則群組會對所有工作負載的總處理量強制執行上限。 | "" |
| `adaptiveQosPolicy` | 為建立的磁碟區指派的自適應 QoS 原則群組。為每個儲存資源池 / 後端選擇 qosPolicy 或 adaptiveQosPolicy 其中之一 | "" |
| `snapshotReserve` | 為快照預留的磁碟區百分比。請勿為 **ASA r2** 系統指定。 | 若 `snapshotPolicy`為「none」，則為「0」，否則為「」 |
| `splitOnClone` | 建立時將複本從其父項分割 | "false" |
| `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" 如果指定、請針對 **ASA r2** 系統設定為 **`true`**。 |
| `luksEncryption` | 啟用 LUKS 加密。請參閱 "使用 Linux Unified Key Setup （ LUKS ）"。 | "" 設定為 **`false`**適用於 **ASA r2** 系統。 |
| `tieringPolicy` | 分層策略使用「無」 請勿為 **ASA r2** 系統指定。 | |
| `nameTemplate` | 用於建立自訂磁碟區名稱的範本。 | "" |

**Volume** 配置範例

以下是定義預設值的範例：

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: trident_svm
username: admin
password: <password>
labels:
  k8scluster: dev2
  backend: dev2-sanbackend
storagePrefix: alternate-trident
debugTraceFlags:
  api: false
  method: true
defaults:
  spaceReserve: volume
  qosPolicy: standard
  spaceAllocation: 'false'
  snapshotPolicy: default
  snapshotReserve: '10'
```

對於使用 `ontap-san`驅動程式建立的所有磁碟區、Trident 會額外增加 10% 的容量至 FlexVol 以容納 LUN 中繼資料。LUN 將根據使用者在 PVC 中請求的確切大小進行佈建。Trident 會額外增加 10% 的容量至 FlexVol（在 ONTAP 中顯示為可用大小）。使用者現在將獲得他們請求的可用容量。此變更還可防止 LUN 在可用空間未完全使用之前變為唯讀。此變更不適用於 ontap-san-economy。

對於定義了 `snapshotReserve`的後端， Trident 會依照下列方式計算磁碟區的大小：

```
Total volume size = [(PVC requested size) / (1 - (snapshotReserve
percentage) / 100)] * 1.1
```

1.1 是 Trident 為 FlexVol 額外增加的 10% ，以容納 LUN 元資料。對於 `snapshotReserve` = 5% ，且 PVC 請求 = 5 GiB ，總磁碟區大小為 5.79 GiB ，可用大小為 5.5 GiB 。 `volume show`命令應顯示與此範例類似的結果：


目前，調整大小是將新計算方法應用於現有磁碟區的唯一方法。

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。

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

**ONTAP SAN 範例**

這是使用 `ontap-san` 驅動程式的基本配置。

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
labels:
  k8scluster: test-cluster-1
  backend: testcluster1-sanbackend
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回退，請使用 `managementLIF` 指定 SVM 並省略 `svm` 參數。例如：

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

**ONTAP SAN 經濟範例**

```
version: 1
storageDriverName: ontap-san-economy
managementLIF: 10.0.0.1
svm: svm_iscsi_eco
username: vsadmin
password: <password>
```

基於憑證的驗證範例

在這個基本設定範例中 `clientCertificate`、 `clientPrivateKey` 和 `trustedCACertificate`（選用，如果使用受信任的 CA）會填入 `backend.json` 中，並分別採用用戶端憑證、私密金鑰和受信任的 CA 憑證的 base64 編碼值。

```
---
version: 1
storageDriverName: ontap-san
backendName: DefaultSANBackend
managementLIF: 10.0.0.1
svm: svm_iscsi
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
clientCertificate: ZXR0ZXJwYXB...ICMgJ3BhcGVyc2
clientPrivateKey: vciwKIyAgZG...0cnksIGRlc2NyaX
trustedCACertificate: zcyBbaG...b3Igb3duIGNsYXNz
```

### 雙向 CHAP 範例

這些範例建立了一個後端，並將 `useCHAP`設為 `true`。

#### ONTAP SAN CHAP 範例

```
---
version: 1
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_iscsi
labels:
  k8scluster: test-cluster-1
  backend: testcluster1-sanbackend
useCHAP: true
chapInitiatorSecret: cl9qxIm36DKyawxy
chapTargetInitiatorSecret: rqxigXgkesIpwxyz
chapTargetUsername: iJF4heBRT0TCwxyz
chapUsername: uh2aNCLSd6cNwxyz
username: vsadmin
password: <password>
```

#### ONTAP SAN economy CHAP 範例

```
---
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>
```

#### 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
```

使用 **nameTemplate** 的後端組態範例

```
---
version: 1
storageDriverName: ontap-san
backendName: ontap-san-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}}"
```

**formatOptions ontap-san-economy** 驅動程式範例

```
---
version: 1
storageDriverName: ontap-san-economy
managementLIF: ""
svm: svm1
username: ""
password: "!"
storagePrefix: whelk_
debugTraceFlags:
  method: true
  api: true
defaults:
  formatOptions: -E nodiscard
```

具有虛擬資源池的後端範例

在這些範例後端定義檔中，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。備註設置在 FlexVol volume 上。Trident 在配置時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。

## ONTAP SAN 範例

```
---
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"
```

**ONTAP SAN 經濟範例**

```
---
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-san` 後端的第三個虛擬資源池和 `ontap-san-economy` 後端的第四個虛擬資源池。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"
```

- 此 `my-test-app-sc` StorageClass 會對應到 `testAPP` 虛擬資源池，在 `ontap-san` 驅動程式中使用 `sanType: nvme`。這是唯一提供 `testApp` 的資源池。

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

Trident 將決定選擇哪個虛擬資源池，並確保符合儲存需求。

## ONTAP NAS 驅動程式

### ONTAP NAS 驅動程式概述

了解如何使用 ONTAP 和 Cloud Volumes ONTAP NAS 驅動程式設定 ONTAP 後端。

**ONTAP NAS** 驅動程式詳細資料

Trident 提供以下 NAS 儲存驅動程式，用於與 ONTAP 叢集通訊。支援的存取模式有：*ReadWriteOnce* （RWO ）、*ReadOnlyMany* （ROX）、*ReadWriteMany* （RWX）、*ReadWriteOncePod* （RWOP）。

| 驅動程式 | 傳輸協定 | **volumeMode** | 支援的存取模式 | 支援的檔案系統 |
|---|---|---|---|---|
| `ontap-nas` | NFS SMB | 檔案系統 | RWO 、 ROX 、 RWX 、 RWOP | "", `nfs`, `smb` |
| `ontap-nas-economy` | NFS SMB | 檔案系統 | RWO 、 ROX 、 RWX 、 RWOP | "", `nfs`, `smb` |
| `ontap-nas-flexgroup` | NFS SMB | 檔案系統 | RWO 、 ROX 、 RWX 、 RWOP | "", `nfs`, `smb` |

- 僅當預期持久性磁碟區使用次數高於"支援的 ONTAP Volume 限制"時，才使用 `ontap-san-economy`。
- 僅當預期持久性磁碟區使用次數高於"支援的 ONTAP Volume 限制" 且無法使用 `ontap-san-economy` 驅動程式時，才使用 `ontap-nas-economy`。
- 如果您預計需要資料保護、災難復原或行動性，請勿使用 `ontap-nas-economy`。
- NetApp 除 ontap-san 外，不建議在所有 ONTAP 驅動程式中使用 Flexvol 自動增長功能。作為變通方案，Trident 支援使用快照預留，並相應地擴展 Flexvol 磁碟區。

使用者權限

Trident 希望以 ONTAP 或 SVM 管理員身分執行，通常使用 `admin` 叢集使用者或 `vsadmin` SVM 使用者，或使用具有相同角色但名稱不同的使用者。

對於 Amazon FSx for NetApp ONTAP 部署、 Trident 需要以 ONTAP 或 SVM 管理員身分執行，可以使用叢集 `fsxadmin`使用者、 `vsadmin` SVM 使用者、或使用具有相同角色但名稱不同的使用者。 `fsxadmin`使用者是叢集管理員使用者的有限替代方案。

如果使用 `limitAggregateUsage`參數，則需要叢集管理員權限。將 Amazon FSx for NetApp ONTAP 與 Trident 搭配使用時， `limitAggregateUsage`參數與 `vsadmin`和 `fsxadmin`使用者帳戶不相容。如果指定此參數，組態作業將失敗。

雖然可以在 ONTAP 中建立更嚴格的角色供 Trident 驅動程式使用，但我們不建議這樣做。大多數新版本的 Trident 都會呼叫額外的 API，這些 API 需要考慮，這會讓升級變得困難且容易出錯。

準備使用 **ONTAP NAS** 驅動程式設定後端

## 了解使用 ONTAP NAS 驅動程式配置 ONTAP 後端的要求、驗證選項和匯出原則。

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 ONTAP 系統（ASA、AFF 和 FAS）在儲存層的實作方式上有所不同。

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

在 Trident 後端組態中、您無需指定系統為 AFX。當您選擇 `ontap-nas`作為 `storageDriverName`時、Trident 會自動偵測 AFX 系統。

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。如需詳細資訊，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，後端定義必須包含用戶端憑證、金鑰以及受信任 CA 憑證（如果使用，建議使用）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。

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**

```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**

```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> 替換為管理 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...Vaaalllluuuueeee",
"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                 |
STATE   | VOLUMES |
+------------+----------------+--------------------------------------
+--------+---------+
| NasBackend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       9 |
+------------+----------------+--------------------------------------
+--------+---------+
```

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。

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
```

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

- `autoExportPolicy` 設定為 `true`
- 儲存前置詞 `trident`
- PVC UUID a79bcf5f-7b6d-4a40-9876-e2551f159c1c
- 名為 trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c 的 qtree 會為名為 `trident-403b5326-8482-40db96d0-d83fb3f4daec`的 FlexVol 建立匯出原則、為名為 `trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c`的 qtree 建立匯出原則，並在 SVM 上建立名為 `trident_empty`的空白匯出原則。FlexVol 匯出原則的規則將是 qtree 匯出原則中所含任何規則的超集。未附加的磁碟區將重複使用空白匯出原則。

- `autoExportCIDRs` 包含地址塊列表。此字段為可選字段，預設值為 ["0.0.0.0/0", "::/0"]。如果未定義，Trident 會新增在已發佈訊息的工作節點上找到的所有全域作用域單播位址。

在本例中，提供了 `192.168.0.0/24` 位址空間。這表示位於此位址範圍內且具有發佈的 Kubernetes 節點 IP 將新增至 Trident 建立的匯出原則。當 Trident 註冊其執行所在的節點時，會擷取該節點的 IP 位址，並根據 `autoExportCIDRs` 中提供的位址區塊進行檢查。在發佈時，篩選 IP 後，Trident 會為其發佈目標節點的用戶端 IP 建立匯出原則規則。

建立後端後，您可以更新其 `autoExportPolicy`和 `autoExportCIDRs`。您可以為自動管理的後端附加新的 CIDR 或刪除現有的 CIDR。刪除 CIDR 時請務必小心，以確保現有連線不會中斷。您也可以選擇停用後端的 `autoExportPolicy`，並回退到手動建立的匯出原則。這需要在後端組態中設定 `exportPolicy`參數。

Trident 建立或更新後端後、您可以使用 `tridentctl`或對應的 `tridentbackend` CRD 來檢查後端：

```
./tridentctl get backends ontap_nas_auto_export -n trident -o yaml
items:
- backendUUID: 403b5326-8482-40db-96d0-d83fb3f4daec
  config:
    aggregate: ""
    autoExportCIDRs:
    - 192.168.0.0/24
    autoExportPolicy: true
    backendName: ontap_nas_auto_export
    chapInitiatorSecret: ""
    chapTargetInitiatorSecret: ""
    chapTargetUsername: ""
    chapUsername: ""
    dataLIF: 192.168.0.135
    debug: false
    debugTraceFlags: null
    defaults:
      encryption: "false"
      exportPolicy: <automatic>
      fileSystemType: ext4
```

當節點被移除時、 Trident 會檢查所有匯出原則、以移除與該節點對應的存取規則。透過從受管理後端的匯出原則中移除此節點 IP 、 Trident 可防止惡意掛載、除非叢集中的新節點重複使用此 IP 。

對於先前存在的後端，使用 `tridentctl update backend`更新後端可確保 Trident 自動管理匯出原則。這會在需要時建立兩個新的匯出原則，分別以後端的 UUID 和 qtree 名稱命名。後端上的磁碟區在卸載並重新掛載後，將使用新建立的匯出原則。

> 刪除具有自動管理匯出原則的後端將刪除動態建立的匯出原則。如果重新建立該後端，則會將其視為新後端，並建立新的匯出原則。

如果正在執行中的節點的 IP 位址更新，您必須重新啟動該節點上的 Trident pod。Trident 隨後會更新其管理的後端匯出原則，以反映此 IP 位址變更。

#### 準備配置 SMB Volume

稍加準備，即可使用 `ontap-nas` 驅動程式配置 SMB Volume 。

> 您必須在 SVM 上同時設定 NFS 和 SMB/CIFS 通訊協定，才能為 ONTAP 內部部署叢集建立 `ontap-nas-economy` SMB Volume。若未設定其中任一通訊協定，將導致 SMB Volume 建立失敗。

> `autoExportPolicy` 不支援 SMB 磁碟區。

開始之前

在配置 SMB 磁碟區之前、您必須具備以下條件。

- Kubernetes 叢集包含一個 Linux 控制器節點和至少一個執行 Windows Server 2022 的 Windows 工作節點。Trident 僅支援掛載到在 Windows 節點上執行的 pod 的 SMB 磁碟區。
- 至少需要一個包含您的 Active Directory 憑證的 Trident 金鑰。要產生金鑰 `smbcreds`：

```
kubectl create secret generic smbcreds --from-literal username=user
--from-literal password='password'
```

- CSI Proxy 設定為 Windows 服務。若要設定 `csi-proxy`，請參閱"GitHub：CSI Proxy"或"GitHub：適用於 Windows 的 CSI Proxy"以瞭解在 Windows 上執行的 Kubernetes 節點。

步驟

1. 對於內部部署 ONTAP、您可以選擇性地建立 SMB 共用區、或 Trident 可以為您建立一個。

   > Amazon FSx for ONTAP 需要 SMB 共用。

   您可以透過兩種方式建立 SMB 管理共用：使用 "Microsoft Management Console" 共用資料夾嵌入式管理單元或使用 ONTAP CLI。若要使用 ONTAP CLI 建立 SMB 共用：

   a. 如有必要、請建立共用區的目錄路徑結構。

      此 `vserver cifs share create`指令會檢查在建立共用時透過 -path 選項指定的路徑。如果指定的路徑不存在，則命令執行失敗。

   b. 建立與指定 SVM 相關聯的 SMB 共用：

```
vserver cifs share create -vserver vserver_name -share-name
share_name -path path [-share-properties share_properties,...]
[other_attributes] [-comment text]
```

c. 確認共用已建立：

```
vserver cifs share show -share-name share_name
```

詳情請參閱 "建立 SMB 共用區"。

2. 建立後端時，必須配置以下內容以指定 SMB 磁碟區。有關所有 FSx for ONTAP 後端設定選項，請參閱 "FSx for ONTAP 設定選項和範例"。

| 參數 | 說明 | 範例 |
|---|---|---|
| `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` |
| `nasType` | * 必須設為 `smb`。*如果為 null ，則預設為 `nfs`。 | `smb` |
| `securityStyle` | 新磁碟區的安全樣式。對於 **SMB** 磁碟區，必須設定為 **`ntfs`或 `mixed`**。 | `ntfs` 或 `mixed` 適用於 SMB 磁碟區 |
| `unixPermissions` | 新磁碟區的模式。**SMB** 磁碟區必須保留空白。 | "" |

### 啟用安全的 **SMB**

從 25.06 版本開始，NetApp Trident 支援使用 `ontap-nas`和 `ontap-nas-economy`後端建立的 SMB 磁碟區的安全性資源配置。啟用安全 SMB 後，您可以使用存取控制清單（ACL）為 Active Directory（AD）使用者和使用者群組提供對 SMB 共用的受控存取權限。

要記住的要點

- 不支援匯入 `ontap-nas-economy`磁碟區。
- `ontap-nas-economy` 磁碟區僅支援唯讀複本。
- 如果啟用了安全 SMB，Trident 將忽略後端提到的 SMB 共用。
- 更新 PVC 註解、儲存類別註解和後端欄位不會更新 SMB 共用 ACL。
- 複製 PVC 註釋中指定的 SMB 共用 ACL 將優先於來源 PVC 中的 ACL 。
- 啟用安全 SMB 時，請確保提供有效的 AD 使用者。無效使用者將不會被加入到 ACL 中。
- 如果在後端、儲存類別和 PVC 中為同一個 AD 使用者提供不同的權限，則權限優先順序為：PVC、儲存類別，然後是後端。

- 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。

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。

在 Trident 後端設定中，您無需指定係統為 NetApp AFX 儲存系統。當您選擇 `ontap-nas`作為 `storageDriverName`時，Trident 會自動偵測 AFX 儲存系統。如下表所示，某些後端組態參數不適用於 AFX 儲存系統。

### 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 |

| 參數 | 說明 | 預設 |
| --- | --- | --- |
| `storageDriverName` | 儲存驅動程式的名稱<br>對於 NetApp AFX 系統，僅支援 `ontap-nas`。 | `ontap-nas`、 `ontap-nas-economy` 或 `ontap-nas-flexgroup` |
| `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。可以指定完全限定域名（FQDN）。如果 Trident 安裝時使用了 IPv6 標誌，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]`。有關無縫 MetroCluster 切換，請參閱MetroCluster 範例。 | "10.0.0.1", "[2001:1234:abcd::fefe]" |
| `dataLIF` | 協定 LIF 的 IP 位址。NetApp 建議指定 `dataLIF`。如果未提供、Trident 會從 SVM 擷取 dataLIF。您可以指定完整網域名稱（FQDN）以用於 NFS 掛載作業、讓您建立循環配置資源 DNS、以便在多個 dataLIF 之間進行負載平衡。可在初始設定後變更。請參閱。如果 Trident 是使用 IPv6 旗標安裝、則可設定為使用 IPv6 位址。IPv6 位址必須以方括弧定義、例如 `[28e8:d9fb:a825:b7bf:69a8:d02f:9e7b:3555]`。* 省略 MetroCluster 。*請參閱MetroCluster 範例。 | 指定的位址或從 SVM 衍生（如果未指定）（不建議） |
| `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 |
| `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。 | 錯誤 |
| `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"]` |
| `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。用於基於憑證的驗證 | "" |
| `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。選用。用於憑證型驗證 | "" |
| `username` | 用於連接叢集 / SVM 的使用者名稱。用於基於憑證的身份驗證。有關 Active Directory 身份驗證，請參閱 "使用 Active Directory 憑證對後端 SVM 進行 Trident 驗證"。 | |

| 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` | 在 SVM 中配置新磁碟區時所使用的前置字元。設定後無法更新<br><br>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。 | "Trident" |
| `aggregate` | 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。<br><br>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。<br><br>請勿指定用於 **AFX** 儲存系統。 | "" |
| `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 | 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。<br>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。 | "" |
| `limitVolumeSize` | 如果要求的磁碟區大小超過此值，則資源配置會失敗。 | "（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` 時，預設使用 `NFS` 磁碟區。如果指定，則對於 **`AFX`** 儲存系統始終設定為 **`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 | 用於使用 ONTAP REST API 的布林參數。 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 及更新版本。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 或更新版本開始， `useREST`預設設定為 `true；將 useREST`變更為 `false`以使用 ONTAPI（ZAPI）呼叫。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 | 為建立的磁碟區指派的 QoS 策略群組。為每個儲存池 / 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 | NFSv4 為 "true" ， NFSv3 為 "false" |
| `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。 | "" |

使用 QoS 原則群組搭配 Trident 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 原則群組會強制限制所有工作負載的總處理量上限。

**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/.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` 命令應顯示類似於此範例的結果：


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。

> 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。虛擬池在儲存部分中定義。

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。

**ONTAP NAS 範例**

```
---
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: mysqldb
      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"
```

**ONTAP NAS 經濟範例**

```
---
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 `parameters.selector` 欄位，每個 StorageClass 都會指定哪些虛擬資源池可用於託管磁碟區。磁碟區將具有所選虛擬資源池中定義的各個層面。

- `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 將對應至 `ontap-nas-flexgroup` 後端的第三個虛擬資源池。這是唯一提供銀級保護和 20000 信用點的資源池。

```
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>
```

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
```

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，可讓客戶啟動並執行由 NetApp ONTAP 儲存作業系統提供支援的檔案系統。FSx for ONTAP 讓您能夠利用熟悉的 NetApp 特性、效能和管理功能，同時享受在 AWS 上儲存資料的簡易性、敏捷性、安全性和可擴充性。FSx for ONTAP 支援 ONTAP 檔案系統特性和管理 API。

您可以將 Amazon FSx for NetApp ONTAP 檔案系統與 Trident 整合，以確保在 Amazon Elastic Kubernetes Service （EKS） 中執行的 Kubernetes 叢集可以配置由 ONTAP 支援的區塊和檔案持續磁碟區。

檔案系統是 Amazon FSx 中的主要資源，類似於內部部署的 ONTAP 叢集。在每個 SVM 中，您可以建立一個或多個磁碟區，這些磁碟區是用於儲存檔案系統中的檔案和資料夾的資料容器。Amazon FSx for NetApp ONTAP 將作為雲端託管檔案系統提供。這種新的檔案系統類型稱為 **NetApp ONTAP**。

將 Trident 與 Amazon FSx for NetApp ONTAP 搭配使用，可確保在 Amazon Elastic Kubernetes Service（EKS ）中執行的 Kubernetes 叢集能夠配置由 ONTAP 支援的區塊和檔案持續磁碟區。

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 叢集或自管理 Kubernetes 叢集。
- 現有的 Amazon FSx for NetApp ONTAP 檔案系統和儲存虛擬機器（SVM），可從叢集的工作節點存取。
- 已準備好 "NFS 或 iSCSI" 的工作節點。

> 請確保根據您的 EKS AMI 類型，按照 Amazon Linux 和 Ubuntu "Amazon Machine Images"（AMI）所需的節點準備步驟進行操作。

考量事項

- SMB 磁碟區：
  - 僅透過 `ontap-nas`驅動程式支援 SMB 磁碟區。
  - Trident EKS 外掛程式不支援 SMB 磁碟區。
  - Trident 僅支援掛載到在 Windows 節點上執行的 Pod 的 SMB 磁碟區。如需詳細資訊，請參閱 "準備配置 SMB Volume"。
- 在 Trident 24.02 之前，在啟用自動備份的 Amazon FSx 檔案系統上建立的磁碟區無法由 Trident 刪除。為了避免在 Trident 24.02 或更高版本中出現此問題，請在 AWS FSx for ONTAP 的後端組態檔中指定 `fsxFilesystemID`、AWS `apiRegion`、AWS `apikey`和 `AWS secretKey`。

> 如果您要為 Trident 指定 IAM 角色，則可以省略明確指定 `apiRegion`、`apiKey` 和 `secretKey` 欄位給 Trident。如需更多資訊，請參閱"FSx for ONTAP 設定選項和範例"。

同時使用 **Trident SAN/iSCSI** 和 **EBS-CSI** 驅動程式

如果您打算將 ontap-san 驅動程式（例如 iSCSI）與 AWS（EKS、ROSA、EC2 或任何其他執行個體）搭配使用，則節點上所需的多路徑配置可能會與 Amazon Elastic Block Store（EBS）CSI 驅動程式衝突。為確保多路徑功能正常運作而不干擾同一節點上的 EBS 磁碟，您需要在多路徑設定中排除 EBS。以下範例展示了一個 `multipath.conf`檔案，其中包含所需的 Trident 設定，同時將 EBS 磁碟從多路徑中排除：

```
defaults {
    find_multipaths no
}
blacklist {
    device {
        vendor "NVME"
        product "Amazon Elastic Block Store"
    }
}
```

驗證

Trident 提供兩種驗證模式。

- 基於憑證（建議）：將憑證安全地儲存在 AWS Secrets Manager 中。您可以使用 `fsxadmin`使用者作為您的檔案系統，或使用為您的 SVM 設定的 `vsadmin`使用者。

> Trident 需要以 `vsadmin` SVM 使用者身分執行，或以具有相同角色但名稱不同的使用者身分執行。Amazon FSx for NetApp ONTAP 提供了一個 `fsxadmin`使用者，可作為 ONTAP `admin`叢集使用者的有限替代方案。我們強烈建議將 `vsadmin`與 Trident 搭配使用。

- 基於憑證：Trident 將使用安裝在 SVM 上的憑證與 FSx 檔案系統上的 SVM 進行通訊。

有關啟用身分驗證的詳細資訊，請參閱您的驅動程式類型的身分驗證說明：

- "ONTAP NAS 認證"
- "ONTAP SAN 驗證"

已測試的 **Amazon Machine Images**（**AMI**）

EKS 叢集支援多種作業系統，但 AWS 已針對容器和 EKS 優化了某些 Amazon Machine Images（AMIs）。以下 AMIs 已使用 NetApp Trident 25.02 進行測試。

| AMI | NAS | NAS 經濟 | iSCSI | iSCSI 經濟性 |
|---|---|---|---|---|
| AL2023_x86_64_STANDARD | 是的 | 是的 | 是的 | 是的 |
| AL2_x86_64 | 是的 | 是的 | 是的* | 是的* |
| BOTTLEROCKET_x86_64 | 是的** | 是的 | N/A | N/A |
| AL2023_ARM_64_STANDARD | 是的 | 是的 | 是的 | 是的 |
| AL2_ARM_64 | 是的 | 是的 | 是的* | 是的* |
| BOTTLEROCKET_ARM_64 | 是的** | 是的 | N/A | N/A |

- * 無法在不重新啟動節點的情況下刪除 PV
- ** 無法與 Trident 版本 25.02 的 NFSv3 搭配使用。

> 如果您所需的 AMI 未在此列出，並不意味著它不受支援；這僅僅意味著它尚未經過測試。此列表僅供參考，列出了已知可正常運作的 AMI。

使用以下工具進行測試：

- EKS 版本：1.32
- 安裝方法：Helm 25.06 和做為 AWS 附加元件 25.06
- 對於 NAS ， NFSv3 和 NFSv4.1 都進行了測試。

- 對於 SAN，僅測試了 iSCSI，未測試 NVMe-oF。

已執行測試：

- 建立：Storage Class、PVC、Pod
- 刪除：pod、pvc（常規、qtree/lun – 經濟型、有 AWS 備份的 NAS）

#### 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。

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"
        }
      }
    }
  ]
}
```

請更新 `trust-relationship.json` 文件中的以下值：

- **<account_id>** - 您的 AWS 帳號 ID
- **<oidc_provider>** - 您的 EKS 叢集的 OIDC。您可以透過執行以下命令來取得 oidc_provider：

```
aws eks describe-cluster --name my-cluster --query
"cluster.identity.oidc.issuer"\
  --output text | sed -e "s/^https:\/\///"
```

將 **IAM** 角色與 **IAM** 原則關聯：

建立角色後，使用此命令將原則（在上述步驟中建立）附加至角色：

```
aws iam attach-role-policy --role-name my-role --policy-arn <IAM policy
ARN>
```

驗證 **OICD** 提供者是否已關聯：

請確認您的 OIDC 提供者已關聯到您的叢集。您可以使用以下命令進行驗證：

```
aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
```

如果輸出為空、請使用以下命令將 IAM OIDC 關聯到您的叢集：

```
eksctl utils associate-iam-oidc-provider --cluster $cluster_name
--approve
```

如果您使用的是 **eksctl**，請使用以下範例在 EKS 中為服務帳戶建立 IAM 角色：

```
eksctl create iamserviceaccount --name trident-controller --namespace
trident \
  --cluster <my-cluster> --role-name AmazonEKS_FSxN_CSI_DriverRole
--role-only \
  --attach-policy-arn <IAM-Policy ARN> --approve
```

#### 安裝 **Trident**

## Trident 簡化了 Kubernetes 中 Amazon FSx for NetApp ONTAP 儲存管理，讓您的開發人員和管理員能夠專注於應用程式部署。

您可以使用下列方法之一安裝 Trident：

- Helm
- 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。如需詳細資訊，請參閱 "為 CSI 磁碟區啟用快照功能"。

#### 透過 **Helm** 安裝 **Trident**

**Pod Identity**

1. 新增 Trident Helm 儲存庫：

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
```

2. 請依照下列範例安裝 Trident：

```
helm install trident-operator netapp-trident/trident-operator --version 100.2502.1 --namespace trident --create-namespace
```

您可以使用 `helm list` 命令來檢視安裝詳細資訊，例如名稱、命名空間、圖表、狀態、應用程式版本和修訂號。

```
helm list -n trident
```

```
NAME                NAMESPACE   REVISION    UPDATED
STATUS       CHART                          APP VERSION

trident-operator    trident     1           2024-10-14
14:31:22.463122 +0300 IDT    deployed     trident-operator-
100.2502.0    25.02.0
```

服務帳戶關聯（**IRSA**）

1. 新增 Trident Helm 儲存庫：

```
helm repo add netapp-trident https://netapp.github.io/trident-helm-chart
```

2. 設定 **cloud provider** 和 **cloud identity** 的值：

```
helm install trident-operator netapp-trident/trident-operator
--version 100.2502.1  \
--set cloudProvider="AWS" \
--set cloudIdentity="'eks.amazonaws.com/role-arn:
arn:aws:iam::<accountID>:role/<AmazonEKS_FSxN_CSI_DriverRole>'" \
--namespace trident \
--create-namespace
```

您可以使用 `helm list` 命令來檢視安裝詳細資訊，例如名稱、命名空間、圖表、狀態、應用程式版本和修訂號。

```
helm list -n trident
```

```
NAME                NAMESPACE   REVISION    UPDATED
STATUS       CHART                          APP VERSION

trident-operator    trident     1           2024-10-14
14:31:22.463122 +0300 IDT    deployed     trident-operator-
100.2510.0    25.10.0
```

如果您打算使用 iSCSI ，請確保用戶端電腦上已啟用 iSCSI 。如果您使用的是 AL2023 Worker node OS ，可以透過在 helm 安裝過程中新增節點準備參數來自動安裝 iSCSI 用戶端：

```
helm install trident-operator netapp-trident/trident-operator
--version 100.2502.1 --namespace trident --create-namespace --
set nodePrep={iscsi}
```

#### 透過 **EKS** 外掛程式安裝 **Trident**

Trident EKS 附加元件包含最新的安全性修補程式、錯誤修正，並經 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：

- 具有附加訂閱的 Amazon EKS 叢集帳戶
- 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

啟用適用於 **AWS** 的 **Trident** 附加元件

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：

   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：

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"
  }
}
```

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。

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
```

#### ONTAP SAN 和 NAS 驅動程式整合

若要建立儲存後端，您需要建立 JSON 或 YAML 格式的組態檔。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 的儲存設備，以及如何使用 AWS 密碼來儲存要使用的 SVM 認證資料：

**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:xxxxxxxx: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:xxxxxxxx:secret:secret-
name",
      "type": "awsarn"
    }
  }
}
```

執行下列命令以建立和驗證 Trident Backend Configuration（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
PHASE    STATUS

backend-tbc-ontap-nas          tbc-ontap-nas         933e0071-66ce-4324-
b9ff-f96d916ac5e9    Bound    Success
```

#### 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：

- `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 。推薦用於區塊儲存。
- `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" 。

設定檔建立完成後，執行以下命令將其建立在 EKS 中：

```
kubectl create -f configuration_file
```

若要驗證狀態、請執行此命令：

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

```
NAME                        BACKEND NAME            BACKEND UUID
PHASE   STATUS
backend-fsx-ontap-nas   backend-fsx-ontap-nas   7a551921-997c-4c37-a1d1-
f2f4c87fa629   Bound   Success
```

後端進階組態和範例

請參閱下表以了解後端組態選項：

| 參數 | 說明 | 範例 |
| --- | --- | --- |
| `version` | | 始終為 1 |
| `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，可以設定為使用 IPv6 位址。IPv6 位址必須用中括號定義，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]" |

| 參數 | 說明 | 範例 |
| --- | --- | --- |
| `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 掛載操作，從而建立輪詢 DNS 以在多個 dataLIF 之間進行負載平衡。初始設定後可以更改。請參閱。**ONTAP SAN** 驅動程式：iSCSI 無需指定。Trident 使用 ONTAP 選擇性 LUN 對應來發現建立多路徑會話所需的 iSCSI LIF。如果明確定義了 dataLIF，則會產生警告。如果 Trident 是使用 IPv6 標誌安裝的，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]。 | |
| `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"]" |
| `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。用於基於憑證的驗證 | "" |
| `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。選用。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` |

| 參數 | 說明 | 範例 |
| --- | --- | --- |
| `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 |
| `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` |

| 參數 | 說明 | 範例 |
| --- | --- | --- |
| 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。 | ""<br>""<br>"" |
| 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。如需詳細資訊，請參閱"建立 AWS Secrets Manager 密碼"。 | |

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 | LUN 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。每個儲存池或後端選擇 qosPolicy 或 adaptiveQosPolicy 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。 | "" |

| 參數 | 說明 | 預設 |
| --- | --- | --- |
| adaptiveQosPolicy | 為建立的磁碟區指派的自適應 QoS 原則群組。每個儲存資源池或後端可選擇 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 "0" | 如果 snapshotPolicy`是 `none，`else`"" |
| splitOnClone | 建立時將複本從其父項分割 | false |
| 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 |
| luksEncryption | 啟用 LUKS 加密。請參閱 "使用 Linux Unified Key Setup （ LUKS ）"。僅限 SAN。 | "" |
| tieringPolicy | 要使用的分層原則 none | |
| unixPermissions | 新磁碟區的模式。**SMB** 磁碟區請保留空白。 | "" |
| 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。 |

#### 配置 **SMB Volume**

您可以使用 ontap-nas 驅動程式來配置 SMB 磁碟區。在完成 ONTAP SAN 和 NAS 驅動程式整合 之前，請先完成以下步驟："準備配置 SMB Volume"

### 配置儲存等級和 **PVC**

配置 Kubernetes StorageClass 物件並建立儲存類別，以指示 Trident 如何配置磁碟區。建立一個 PersistentVolumeClaim（PVC）來使用已配置的 Kubernetes StorageClass 請求存取 PV。然後，您可以將 PV 掛載到 Pod。

#### 建立儲存類別

#### 配置 **Kubernetes StorageClass** 物件

該 "Kubernetes StorageClass 對象"物件將 Trident 識別為該類別使用的儲存配置器，並指示 Trident 如何配置磁碟區。使用此範例為使用 NFS 的磁碟區設定 Storageclass（有關完整的屬性列表，請參閱下方的 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"
```

請使用以下範例為使用 iSCSI 的磁碟區設定 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"
```

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
```

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

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 storage-class-ontapnas.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
```

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 PVC 後，您可以在 pod 中掛載 Volume 。

範例資訊清單

**PersistentVolumeClaim** 樣本清單

這些範例展示了 PVC 的基本配置選項。

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。

#### Trident 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 |

| 屬性 | 類型 | 價值觀 | 優惠 | 要求 | 支援者 |
| --- | --- | --- | --- | --- | --- |
| 加密 | 布林值 | true、false | 儲存池支援加密磁碟區 | 已啟用加密的磁碟區 | ontap-nas、ontap-nas-economy、ontap-nas-flexgroups、ontap-san |
| IOPS | int | 正整數 | Pool 能夠保證此範圍內的 IOPS | Volume 保證了這些 IOPS | solidfire-san |

[1]：ONTAP Select 系統不支援

部署範例應用程式

## 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。

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
```

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。此 EKS 外掛程式可協助您持續確保 Amazon EKS 叢集的安全性和穩定性，並減少安裝、設定和更新外掛程式所需的工作量。

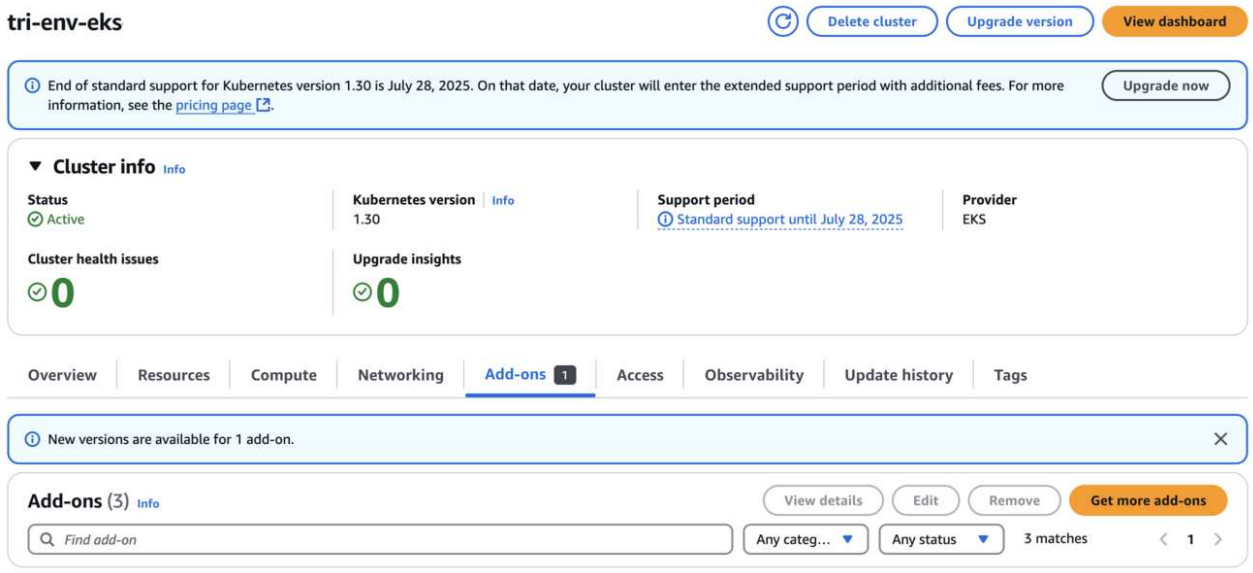
先決條件

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

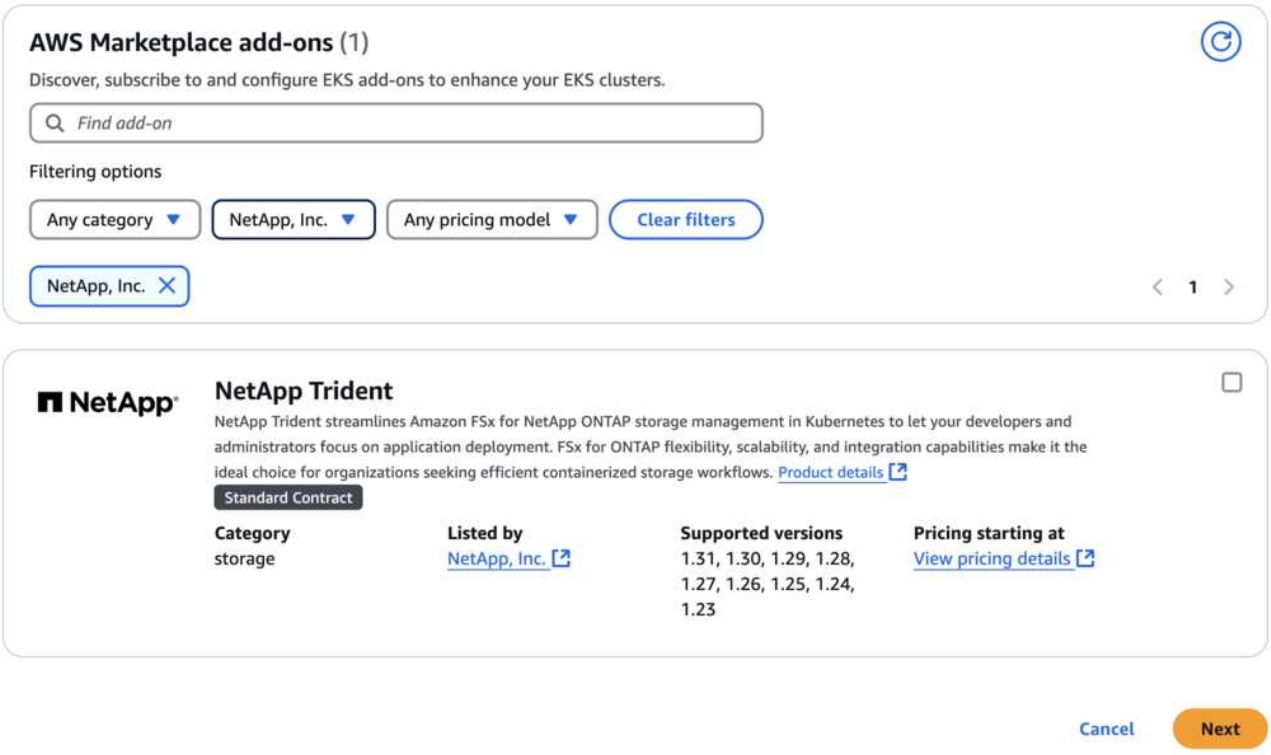
- 具有使用附加元件權限的 Amazon EKS 叢集帳戶。請參閱 "Amazon EKS 附加元件"。
- AWS 對 AWS Marketplace 的權限：
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- AMI 類型：Amazon Linux 2（AL2_x86_64）或 Amazon Linux 2 Arm（AL2_ARM_64）
- 節點類型：AMD 或 ARM
- 現有的 Amazon FSx for NetApp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

步驟

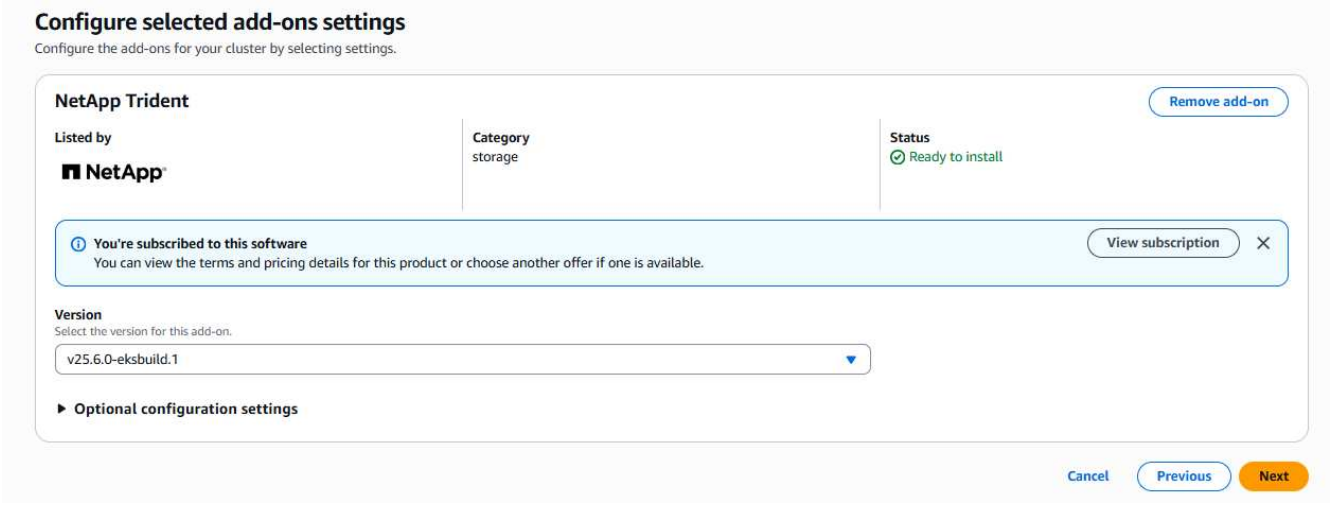
1. 請務必建立 IAM 角色和 AWS 密鑰，以使 EKS Pod 能夠存取 AWS 資源。有關說明，請參閱 "建立 IAM 角色和 AWS Secret"。
2. 在 EKS Kubernetes 叢集上，導覽至 **Add-ons** 標籤。



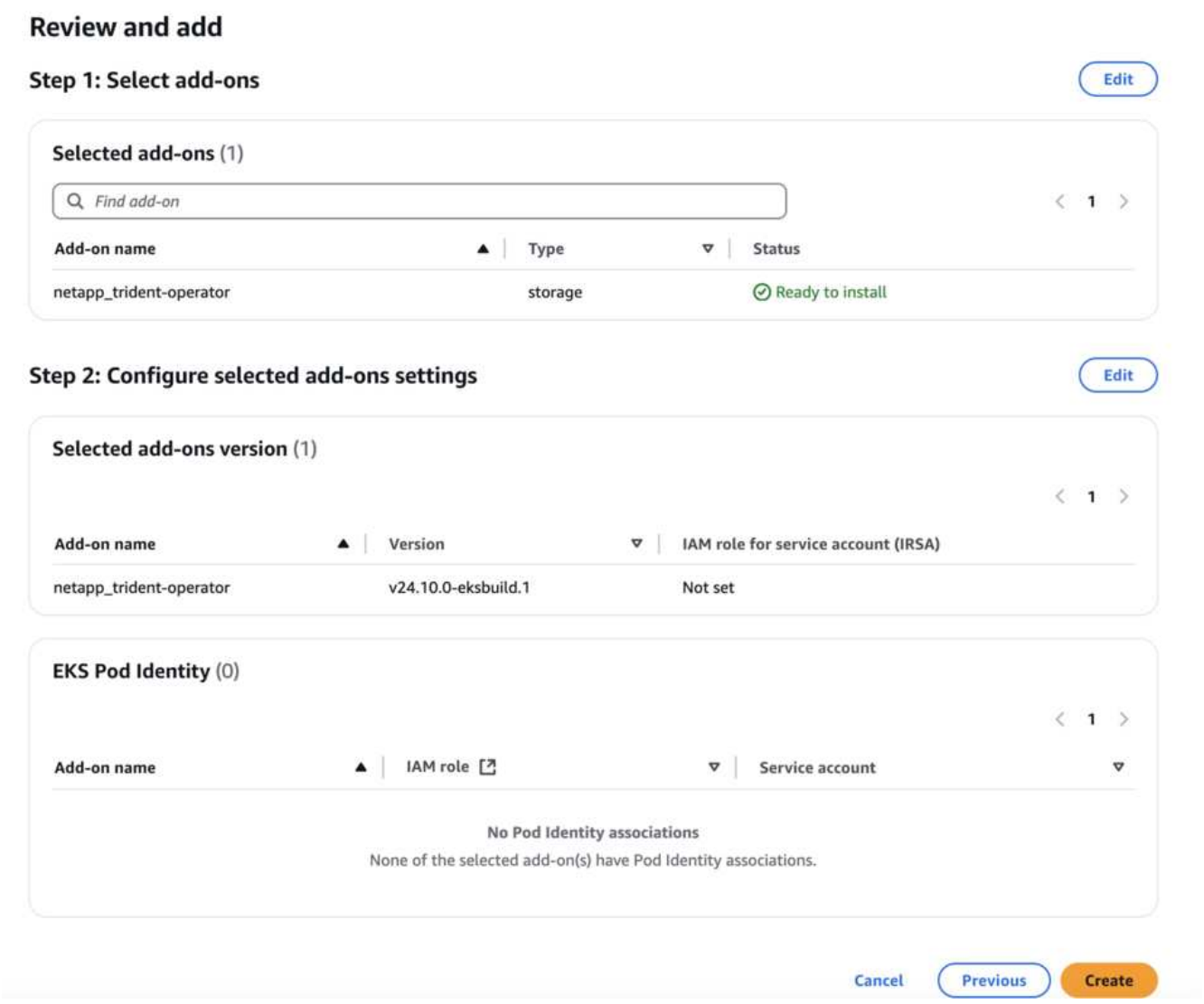

3. 前往 **AWS Marketplace** 外掛程式，然後選擇 *storage* 類別。




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。

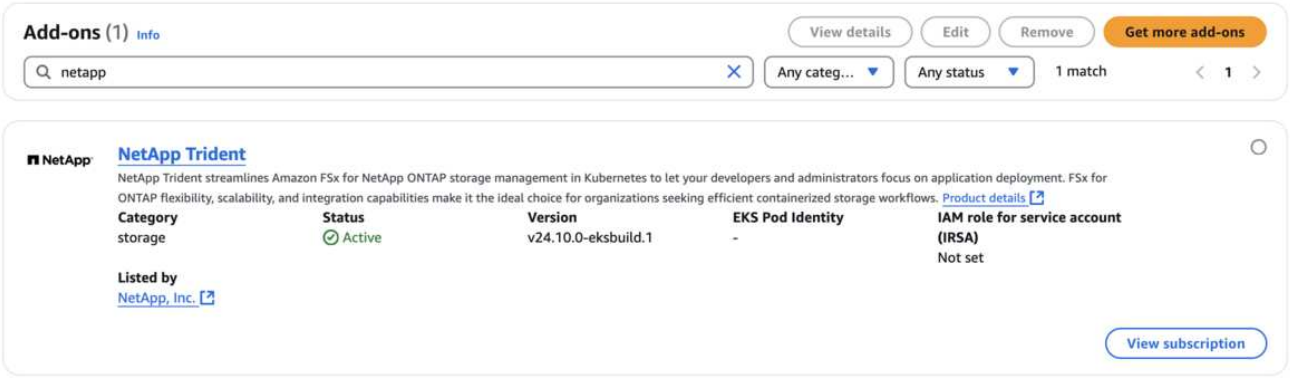



6. 配置所需的附加元件設定。




7. 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"。
8. 選擇 **Create** 。

9. 確認附加元件的狀態為 *Active* 。




10. 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` (`tbe 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。

> `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。

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。

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。其中包含有關儲存叢集 / 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。用於基於認證的身份驗證 |
| ONTAP | 密碼 | 連接叢集 /SVM 的密碼。用於基於憑證的身份驗證 |
| ONTAP | clientPrivateKey | 客戶端私密金鑰的 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 發起方金鑰。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                       BACKEND NAME          BACKEND UUID
PHASE   STATUS
backend-tbc-ontap-san   ontap-san-backend     8d24fce7-6f60-4d4a-8ef6-
bab2699e6ab8   Bound   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。所有新建立的 `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。

> 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
    ```

- 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來檢視記錄以判斷原因：

```
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
   ```

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: mysqldb
      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。

```
kubectl 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
```

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.templ` 檔案、並將 *`BACKEND_TYPE`* 替換為儲存驅動程式名稱。

```
./tridentctl -n trident get backend
+-------------+---------------+--------------------------------------
+--------+---------+
|    NAME     | STORAGE DRIVER |                 UUID                 |
STATE  | VOLUMES |
+-------------+---------------+--------------------------------------
+--------+---------+
| nas-backend | ontap-nas      | 98e19b74-aec7-4a3d-8dcf-128e5033b214 |
online |       0 |
+-------------+---------------+--------------------------------------
+--------+---------+

cp sample-input/storage-class-csi.yaml.templ sample-input/storage-class-
basic-csi.yaml

# Modify __BACKEND_TYPE__ with the storage driver field above (e.g.,
ontap-nas)
vi sample-input/storage-class-basic-csi.yaml
```

## 管理儲存類別

您可以檢視現有儲存類別、設定預設儲存類別、識別儲存類別後端，以及刪除儲存類別。

### 檢視現有的儲存類別

- 若要檢視現有的 Kubernetes 儲存類別，請執行下列命令：

```
kubectl get storageclass
```

- 若要檢視 Kubernetes 儲存類別詳細資訊，請執行下列命令：

```
kubectl get storageclass <storage-class> -o json
```

- 若要檢視 Trident 的同步儲存類別，請執行下列命令：

```
tridentctl get storageclass
```

- 若要檢視 Trident 的同步儲存類別詳細資料、請執行下列命令：

```
tridentctl get storageclass <storage-class> -o json
```

設定預設儲存類別

Kubernetes 1.6 新增了設定預設儲存類別的功能。如果使用者未在持久性磁碟區宣告（PVC）中指定儲存類別，則將使用此儲存類別來配置持久性磁碟區。

- 透過在儲存類別定義中將註解 `storageclass.kubernetes.io/is-default-class` 設為 true 來定義預設儲存類別。根據規範，任何其他值或註解的缺失都將被解釋為 false。
- 您可以使用下列命令將現有儲存類別設定為預設儲存類別：

```
kubectl patch storageclass <storage-class-name> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
```

- 同樣地，您可以使用下列命令移除預設儲存類別註釋：

```
kubectl patch storageclass <storage-class-name> -p '{"metadata":
{"annotations":{"storageclass.kubernetes.io/is-default-class":"false"}}}'
```

Trident 安裝程式套件中也有包含此註解的範例。

叢集中一次只能存在一個預設儲存類別。Kubernetes 技術上並不會禁止存在多個預設儲存類別，但它的行為會如同根本沒有預設儲存類別一樣。

識別儲存類別的後端

這是一個可以使用 `tridentctl` 為 Trident 後端物件輸出的 JSON 資料來回答的問題範例。此範例使用了 `jq` 實用程式，您可能需要先安裝該程式。

```
tridentctl get storageclass -o json | jq  '[.items[] | {storageClass:
.Config.name, backends: [.storage]|unique}]'
```

刪除儲存類別

若要從 Kubernetes 中刪除儲存類別，請執行以下命令：

```
kubectl delete storageclass <storage-class>
```

`<storage-class>` 應該替換成您的儲存類別。

透過此儲存類別建立的任何持久性磁碟區都將保持不變，Trident 將繼續管理它們。

Trident 會為其建立的磁碟區強制執行空白 `fsType`。對於 iSCSI 後端、建議在 StorageClass 中強制執行 `parameters.fsType`。您應該刪除現有的 StorageClasses 並使用指定的 `parameters.fsType`重新建立它們。

# 配置和管理磁碟區

## 配置磁碟區

建立一個 PersistentVolumeClaim（PVC），使用已配置的 Kubernetes StorageClass 來請求存取 PV。然後，您可以將 PV 掛載到 Pod 中。

概況

`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 PVC 之後，您可以在 Pod 中掛載磁碟區。

### 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    1Gi        RWO                   5m
```

1. 在 pod 中掛載 volume 。

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

> 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 task-pv-pod -- df -h /my/mount/path
```

3. 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
```

範例資訊清單

### PersistentVolumeClaim 樣本清單

這些範例展示了 PVC 的基本配置選項。

#### 具有 RWO 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 RWO 存取權限的基本 PVC。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-storage
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: basic-csi
```

#### PVC 與 NVMe/TCP

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 NVMe/TCP 基本 PVC。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: pvc-san-nvme
spec:
accessModes:
  - ReadWriteOnce
resources:
  requests:
    storage: 300Mi
storageClassName: protection-gold
```

**Pod 清單範例**

這些範例展示了將 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: storage
      persistentVolumeClaim:
       claimName: pvc-storage
  containers:
    - name: pv-container
      image: nginx
      ports:
        - containerPort: 80
          name: "http-server"
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: storage
```

基本 **NVMe/TCP** 組態

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-nginx
spec:
  volumes:
    - name: basic-pvc
      persistentVolumeClaim:
        claimName: pvc-san-nvme
  containers:
    - name: task-pv-container
      image: nginx
      volumeMounts:
        - mountPath: "/my/mount/path"
          name: basic-pvc
```

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。

# 擴充磁碟區

Trident 為 Kubernetes 使用者提供了在建立磁碟區後擴充磁碟區的功能。尋找有關擴充 iSCSI、NFS、SMB、NVMe/TCP 和 FC 磁碟區所需的組態資訊。

## 擴充 iSCSI Volume

您可以使用 CSI 配置程式擴充 iSCSI 持續性磁碟區（PV）。

> ⓘ iSCSI 磁碟區擴充受 `ontap-san`、 `ontap-san-economy`、 `solidfire-san` 驅動程式支援，並且需要 Kubernetes 1.16 及更高版本。

### 步驟 1：設定 StorageClass 以支援磁碟區擴充

編輯 StorageClass 定義，將 `allowVolumeExpansion` 欄位設為 `true`。

```
cat storageclass-ontapsan.yaml
```

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-san
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
allowVolumeExpansion: True
```

對於已存在的 StorageClass，對其進行編輯以包含 `allowVolumeExpansion` 參數。

### 步驟 2：使用您建立的 StorageClass 建立 PVC

編輯 PVC 定義並更新 `spec.resources.requests.storage` 以反映新的所需尺寸，該尺寸必須大於原始尺寸。

```
cat pvc-ontapsan.yaml
```

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

Trident 會建立持續磁碟區（PV），並將其與此持續磁碟區宣告（PVC）建立關聯。

```
kubectl get pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi
RWO            ontap-san      8s

kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM              STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi        RWO
Delete           Bound    default/san-pvc    ontap-san               10s
```

#### 步驟 3：定義連接 PVC 的 pod

將 PV 連接到 Pod 以調整其大小。調整 iSCSI PV 大小時有兩種情況：

- 如果 PV 連接到 pod，Trident 會擴展儲存後端上的 Volume、重新掃描裝置並調整檔案系統大小。
- 嘗試調整未掛載的 PV 的大小時，Trident 會在儲存後端擴充該磁碟區。PVC 綁定到 Pod 後，Trident 會重新掃描裝置並調整檔案系統的大小。擴充操作成功完成後，Kubernetes 會更新 PVC 的大小。

在此範例中、會建立使用 `san-pvc` 的 Pod 。

```
 kubectl get pod
NAME         READY   STATUS    RESTARTS   AGE
ubuntu-pod   1/1     Running   0          65s

 kubectl describe pvc san-pvc
Name:          san-pvc
Namespace:     default
StorageClass:  ontap-san
Status:        Bound
Volume:        pvc-8a814d62-bd58-4253-b0d1-82f2885db671
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
               volume.beta.kubernetes.io/storage-provisioner:
csi.trident.netapp.io
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      1Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Mounted By:    ubuntu-pod
```

#### 步驟 4：展開 PV

若要將已建立的 PV 的大小從 1Gi 調整為 2Gi，請編輯 PVC 定義並將 `spec.resources.requests.storage` 更新為 2Gi。

```
kubectl edit pvc san-pvc
```

```
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: "2019-10-10T17:32:29Z"
  finalizers:
  - kubernetes.io/pvc-protection
  name: san-pvc
  namespace: default
  resourceVersion: "16609"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/san-pvc
  uid: 8a814d62-bd58-4253-b0d1-82f2885db671
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
 # ...
```

步驟 **5**：驗證擴充

您可以檢查 PVC、PV 和 Trident Volume 的大小，以驗證擴充是否正常運作：

```
kubectl get pvc san-pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi
RWO            ontap-san      11m
kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM             STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi        RWO
Delete           Bound    default/san-pvc   ontap-san               12m
tridentctl get volumes -n trident
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
| pvc-8a814d62-bd58-4253-b0d1-82f2885db671 | 2.0 GiB | ontap-san     |
block    | a9b7bfff-0505-4e31-b6c5-59f492e02d33 | online | true    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

### 擴充 FC Volume

您可以使用 CSI 設定程式擴充 FC Persistent Volume（PV）。

FC Volume 擴充受 `ontap-san` 驅動程式支援，需要 Kubernetes 1.16 及更高版本。

#### 步驟 1：設定 StorageClass 以支援磁碟區擴充

編輯 StorageClass 定義，將 `allowVolumeExpansion` 欄位設為 `true`。

```
cat storageclass-ontapsan.yaml
```

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontap-san
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
allowVolumeExpansion: True
```

對於已存在的 StorageClass，對其進行編輯以包含 `allowVolumeExpansion` 參數。

#### 步驟 2：使用您建立的 StorageClass 建立 PVC

編輯 PVC 定義並更新 `spec.resources.requests.storage` 以反映新的所需尺寸，該尺寸必須大於原始尺寸。

```
cat pvc-ontapsan.yaml
```

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

Trident 會建立持續磁碟區（PV），並將其與此持續磁碟區宣告（PVC）建立關聯。

```
kubectl get pvc
NAME       STATUS    VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc    Bound     pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi
RWO            ontap-san      8s

kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM              STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   1Gi        RWO
Delete           Bound    default/san-pvc    ontap-san               10s
```

#### 步驟 3：定義連接 PVC 的 pod

將 PV 附加至 Pod 以調整其大小。調整 FC PV 大小時有兩種情況：

- 如果 PV 連接到 pod，Trident 會擴展儲存後端上的 Volume、重新掃描裝置並調整檔案系統大小。
- 嘗試調整未掛載的 PV 的大小時，Trident 會在儲存後端擴充該磁碟區。PVC 綁定到 Pod 後，Trident 會重新掃描裝置並調整檔案系統的大小。擴充操作成功完成後，Kubernetes 會更新 PVC 的大小。

在此範例中、會建立使用 `san-pvc` 的 Pod 。

```
 kubectl get pod
NAME         READY   STATUS    RESTARTS   AGE
ubuntu-pod   1/1     Running   0          65s

 kubectl describe pvc san-pvc
Name:          san-pvc
Namespace:     default
StorageClass:  ontap-san
Status:        Bound
Volume:        pvc-8a814d62-bd58-4253-b0d1-82f2885db671
Labels:        <none>
Annotations:   pv.kubernetes.io/bind-completed: yes
               pv.kubernetes.io/bound-by-controller: yes
               volume.beta.kubernetes.io/storage-provisioner:
csi.trident.netapp.io
Finalizers:    [kubernetes.io/pvc-protection]
Capacity:      1Gi
Access Modes:  RWO
VolumeMode:    Filesystem
Mounted By:    ubuntu-pod
```

步驟 **4**：展開 **PV**

若要將已建立的 PV 的大小從 1Gi 調整為 2Gi，請編輯 PVC 定義並將 `spec.resources.requests.storage` 更新為 2Gi。

```
kubectl edit pvc san-pvc
```

```
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: "2019-10-10T17:32:29Z"
  finalizers:
  - kubernetes.io/pvc-protection
  name: san-pvc
  namespace: default
  resourceVersion: "16609"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/san-pvc
  uid: 8a814d62-bd58-4253-b0d1-82f2885db671
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
 # ...
```

步驟 **5**：驗證擴充

您可以檢查 PVC、PV 和 Trident Volume 的大小，以驗證擴充是否正常運作：

```
kubectl get pvc san-pvc
NAME      STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
san-pvc   Bound    pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi
RWO            ontap-san      11m
kubectl get pv
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM             STORAGECLASS   REASON   AGE
pvc-8a814d62-bd58-4253-b0d1-82f2885db671   2Gi        RWO
Delete           Bound    default/san-pvc   ontap-san               12m
tridentctl get volumes -n trident
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
| pvc-8a814d62-bd58-4253-b0d1-82f2885db671 | 2.0 GiB | ontap-san     |
block    | a9b7bfff-0505-4e31-b6c5-59f492e02d33 | online | true    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

### 擴充 NFS Volume

Trident 支援在 `ontap-nas`、 `ontap-nas-economy`、 `ontap-nas-flexgroup`和 `azure-netapp-files`後端上配置的 NFS PV 的磁碟區擴充。

#### 步驟 1：設定 StorageClass 以支援磁碟區擴充

要調整 NFS PV 的大小，管理員首先需要配置儲存類別以允許磁碟區擴展，方法是將 `allowVolumeExpansion`欄位設為 `true`：

```
cat storageclass-ontapnas.yaml
```

```yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: ontapnas
provisioner: csi.trident.netapp.io
parameters:
  backendType: ontap-nas
allowVolumeExpansion: true
```

如果您已經建立了一個沒有此選項的儲存類別，您可以簡單地使用 `kubectl edit storageclass` 編輯現有儲存類別以允許磁碟區擴充。

步驟 2：使用您建立的 StorageClass 建立 PVC

```
cat pvc-ontapnas.yaml
```

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: ontapnas20mb
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 20Mi
  storageClassName: ontapnas
```

Trident 應該為此 PVC 建立 20 MiB NFS PV：

```
kubectl get pvc
NAME           STATUS    VOLUME
CAPACITY     ACCESS MODES    STORAGECLASS    AGE
ontapnas20mb   Bound     pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   20Mi
RWO            ontapnas        9s

kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM                  STORAGECLASS    REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   20Mi       RWO
Delete           Bound    default/ontapnas20mb   ontapnas
2m42s
```

步驟 3：展開 PV

若要將新建的 20 MiB PV 調整為 1 GiB，請編輯 PVC 並將 `spec.resources.requests.storage` 設為 1 GiB：

```
kubectl edit pvc ontapnas20mb
```

```
# Please edit the object below. Lines beginning with a '#' will be
ignored,
# and an empty file will abort the edit. If an error occurs while saving
this file will be
# reopened with the relevant failures.
#
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
    pv.kubernetes.io/bind-completed: "yes"
    pv.kubernetes.io/bound-by-controller: "yes"
    volume.beta.kubernetes.io/storage-provisioner: csi.trident.netapp.io
  creationTimestamp: 2018-08-21T18:26:44Z
  finalizers:
  - kubernetes.io/pvc-protection
  name: ontapnas20mb
  namespace: default
  resourceVersion: "1958015"
  selfLink: /api/v1/namespaces/default/persistentvolumeclaims/ontapnas20mb
  uid: c1bd7fa5-a56f-11e8-b8d7-fa163e59eaab
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
# ...
```

步驟 **4**：驗證擴充

您可以檢查 PVC、PV 和 Trident Volume 的大小來驗證調整大小是否正確：

```
kubectl get pvc ontapnas20mb
NAME           STATUS   VOLUME
CAPACITY   ACCESS MODES   STORAGECLASS    AGE
ontapnas20mb   Bound    pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   1Gi
RWO            ontapnas        4m44s

kubectl get pv pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7
NAME                                       CAPACITY   ACCESS MODES
RECLAIM POLICY   STATUS   CLAIM                  STORAGECLASS    REASON
AGE
pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7   1Gi        RWO
Delete           Bound    default/ontapnas20mb   ontapnas
5m35s

tridentctl get volume pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 -n trident
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
| pvc-08f3d561-b199-11e9-8d9f-5254004dfdb7 | 1.0 GiB | ontapnas      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | true    |
+-----------------------------------------+---------+---------------
+----------+-----------------------------------------+--------+---------+
```

## 匯入磁碟區

## 您可以使用 `tridentctl import`或透過使用 Trident 匯入註解建立持久性磁碟區宣告（PVC），將現有儲存磁碟區匯入為 Kubernetes PV。

### 概述和注意事項

您可以將磁碟區匯入 Trident 以：

- 將應用程式容器化並重複使用其現有資料集
- 為臨時應用程式使用資料集的複本
- 重建失敗的 Kubernetes 叢集
- 在災難復原期間遷移應用程式資料

### 考量事項

在匯入磁碟區之前，請檢閱下列考量事項。

- Trident 只能匯入 RW（讀寫）類型的 ONTAP 磁碟區。DP（資料保護）類型的磁碟區是 SnapMirror 目標磁碟區。在將磁碟區匯入 Trident 之前，您應該中斷鏡射關係。

- 我們建議匯入沒有作用中連線的磁碟區。若要匯入正在使用的磁碟區，請複製該磁碟區，然後執行匯入。

  這一點對於區塊磁碟區尤其重要，因為 Kubernetes 無法感知先前的連線，很容易將作用中磁碟區附加到 Pod 上。這可能導致資料毀損。

- 雖然 `StorageClass`必須在 PVC 上指定，但 Trident 在匯入期間不會使用此參數。儲存類別用於在建立磁碟區時根據儲存特性從可用的儲存池中進行選擇。由於磁碟區已存在，因此在匯入期間不需要選擇儲存池。因此，即使磁碟區存在於與 PVC 中指定的儲存類別不符的後端或儲存池上，匯入也不會失敗。
- 現有磁碟區的大小在 PVC 中決定和設定。儲存驅動程式匯入磁碟區後，會建立 PV 並將其 ClaimRef 與 PVC 關聯起來。
  - 回收原則最初在 PV 中設定為 `retain`。在 Kubernetes 成功繫結 PVC 和 PV 後，回收原則會更新為與 Storage Class 的回收原則相符。
  - 如果 Storage Class 的回收策略為 `delete`，則當 PV 被刪除時，儲存磁碟區也會被刪除。
- 預設情況下， Trident 會管理 PVC ，並在後端重新命名 FlexVol Volume 和 LUN 。您可以傳遞 `--no-manage`旗標來匯入非託管 Volume ，以及 `--no-rename`旗標來保留 Volume 名稱。
  - `--no-manage`* - 如果使用 `--no-manage`標誌，Trident 在物件生命週期內不會對 PVC 或 PV 執行任何其他操作。刪除 PV 時，儲存磁碟區不會被刪除，其他操作（例如磁碟區複製和磁碟區調整大小）也會被忽略。
  - `--no-rename`* - 如果使用 `--no-rename` 標誌、Trident 會在匯入磁碟區時保留現有磁碟區名稱、並管理磁碟區的生命週期。此選項僅支援 `ontap-nas`、 `ontap-san`（包括 ASA r2 系統）和 `ontap-san-economy` 驅動程式。

    如果您想使用 Kubernetes 進行容器化工作負載，但又想在 Kubernetes 之外管理儲存磁碟區的生命週期，那麼這些選項非常有用。

- PVC 和 PV 會新增一個註釋，其作用有兩個：一是指示該磁碟區已匯入，二是指示 PVC 和 PV 是否受管理。此註釋不應修改或刪除。

匯入磁碟區

您可以使用 `tridentctl import` 或透過建立具有 Trident 匯入註解的 PVC 來匯入磁碟區。

如果使用 PVC 註釋，則無需下載或使用 `tridentctl`匯入磁碟區。

使用 **tridentctl**

步驟

1. 建立 PVC 檔案（例如 `pvc.yaml`），用於建立 PVC。PVC 檔案應包含 `name`、 `namespace`、 `accessModes`和 `storageClassName`。您也可以在 PVC 定義中指定 `unixPermissions`。

   以下是最低規格範例：

   ```
   kind: PersistentVolumeClaim
   apiVersion: v1
   metadata:
     name: my_claim
     namespace: my_namespace
   spec:
     accessModes:
       - ReadWriteOnce
     storageClassName: my_storage_class
   ```

   僅包含必需參數。其他參數（例如 PV 名稱或磁碟區大小）可能會導致匯入命令失敗。

2. 使用 `tridentctl import` 指令指定包含磁碟區的 Trident 後端名稱以及儲存上唯一標識該磁碟區的名稱（例如：ONTAP FlexVol、Element Volume）。 `-f` 參數是必需的，用於指定 PVC 檔案的路徑。

   ```
   tridentctl import volume <backendName> <volumeName> -f <path-to-pvc-file>
   ```

使用 **PVC** 註釋

步驟

1. 建立一個 PVC YAML 檔案（例如， `pvc.yaml`），其中包含所需的 Trident 匯入註解。PVC 檔案應包含：

   - `name` 和 `namespace` 在中繼資料中
   - `accessModes`、 `resources.requests.storage` 和 `storageClassName` 在規格中
   - 註釋：
     - `trident.netapp.io/importOriginalName`：後端的磁碟區名稱
     - `trident.netapp.io/importBackendUUID`：磁碟區所在的後端 UUID
     - `trident.netapp.io/notManaged`（可選）：設定為 `"true"`表示非託管磁碟區。預設值為 `"false"`。

       以下是匯入託管磁碟區的範例規格：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: <pvc-name>
  namespace: <namespace>
  annotations:
    trident.netapp.io/importOriginalName: "<volume-name>"
    trident.netapp.io/importBackendUUID: "<backend-uuid>"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: <size>
  storageClassName: <storage-class-name>
```

2. 將 PVC YAML 檔案套用到您的 Kubernetes 叢集：

```
kubectl apply -f <pvc-file>.yaml
```

Trident 會自動匯入磁碟區並將其繫結至 PVC。

範例

請查看以下磁碟區匯入範例，以了解支援的驅動程式。

#### ONTAP NAS 和 ONTAP NAS FlexGroup

Trident 支援使用 `ontap-nas` 和 `ontap-nas-flexgroup` 驅動程式進行磁碟區匯入。

- Trident 不支援使用 `ontap-nas-economy` 驅動程式進行磁碟區匯入。
- `ontap-nas` 和 `ontap-nas-flexgroup` 驅動程式不允許重複的磁碟區名稱。

使用 `ontap-nas`驅動程式建立的每個磁碟區都是 `ONTAP` 叢集上的 `FlexVol` 磁碟區。使用 `ontap-nas`驅動程式匯入 `FlexVol` 磁碟區的操作方式相同。`ONTAP` 叢集上已存在的 `FlexVol` 磁碟區可以作為 `ontap-nas` PVC 匯入。同樣、FlexGroup 磁碟區也可以作為 `ontap-nas-flexgroup` PVC 匯入。

使用 **tridentctl** 的 **ONTAP NAS** 範例

以下範例展示如何使用 `tridentctl`匯入託管磁碟區和非託管磁碟區。

託管磁碟區

以下範例匯入名為 `managed_volume`的 Volume，該 Volume 位於名為 `ontap_nas`的後端：

```
tridentctl import volume ontap_nas managed_volume -f <path-to-pvc-file>

+----------------------------------------+---------+---------------
+----------+----------------------------------------+---------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------+---------------
+----------+----------------------------------------+---------+---------+
| pvc-bf5ad463-afbb-11e9-8d9f-5254004dfdb7 | 1.0 GiB | standard      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | true    |
+----------------------------------------+---------+---------------
+----------+----------------------------------------+---------+---------+
```

未託管磁碟區

使用 `--no-manage` 參數時、Trident 不會重新命名磁碟區。

以下範例會在 `ontap_nas` 後端匯入 `unmanaged_volume`：

```
tridentctl import volume nas_blog unmanaged_volume -f <path-to-pvc-
file> --no-manage

+----------------------------------------+---------+---------------
+----------+----------------------------------------+---------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------+---------------
+----------+----------------------------------------+---------+---------+
| pvc-df07d542-afbc-11e9-8d9f-5254004dfdb7 | 1.0 GiB | standard      |
file     | c5a6f6a4-b052-423b-80d4-8fb491a14a22 | online | false   |
+----------------------------------------+---------+---------------
+----------+----------------------------------------+---------+---------+
```

使用 **PVC** 註解的 **ONTAP NAS** 範例

以下範例展示如何使用 PVC 註解匯入託管和非託管磁碟區。

託管磁碟區

以下範例從後端 `81abcb27-ea63-49bb-b606-0a5315ac5f21`匯入一個名為 `ontap_volume1`的 1GiB `ontap-nas` 磁碟區，並使用 PVC 註解設定了 RWO 存取模式：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: <managed-imported-volume>
  namespace: <namespace>
  annotations:
    trident.netapp.io/importOriginalName: "ontap_volume1"
    trident.netapp.io/importBackendUUID: "81abcb27-ea63-49bb-b606-
0a5315ac5f21"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: <storage-class-name>
```

未託管磁碟區

以下範例從後端 `34abcb27-ea63-49bb-b606-0a5315ac5f34`匯入名為 `ontap-volume2`的 1Gi `ontap-nas` 磁碟區，並使用 PVC 註解設定 RWO 存取模式：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: <umanaged-imported-volume>
  namespace: <namespace>
  annotations:
    trident.netapp.io/importOriginalName: "ontap-volume2"
    trident.netapp.io/importBackendUUID: "34abcb27-ea63-49bb-b606-
0a5315ac5f34"
    trident.netapp.io/notManaged: "true"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: <storage-class-name>
```

**ONTAP SAN**

Trident 支援使用 `ontap-san`（iSCSI、NVMe/TCP 和 FC）及 `ontap-san-economy` 驅動程式進行磁碟區匯入。

Trident 可以匯入包含單一 LUN 的 ONTAP SAN FlexVol Volume 。這與 `ontap-san`驅動程式一致、此驅動程式會為每個 PVC 建立一個 FlexVol Volume 、並在 FlexVol Volume 內建立一個 LUN 。Trident 會匯入 FlexVol Volume 並將其與 PVC 定義建立關聯。Trident 可以匯入 `ontap-san-economy`包含多個 LUN 的 Volume 。

以下範例展示如何匯入託管和非託管磁碟區：

託管磁碟區

對於託管磁碟區、 Trident 會將 FlexVol 磁碟區重新命名為 `pvc-<uuid>`格式、並將 `FlexVol` 磁碟區內的 `LUN` 重新命名為 `lun0。

以下範例匯入 `ontap-san-managed` FlexVol volume，該磁碟區存在於 `ontap_san_default`後端：

```
tridentctl import volume ontapsan_san_default ontap-san-managed -f pvc-
basic-import.yaml -n trident -d

+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
| pvc-d6ee4f54-4e40-4454-92fd-d00fc228d74a | 20 MiB | basic         |
block    | cd394786-ddd5-4470-adc3-10c5ce4ca757 | online | true    |
+-----------------------------------------+--------+--------------
+----------+-----------------------------------------+--------+---------+
```

未託管磁碟區

以下範例會在 `unmanaged_example_volume` `ontap_san` 後端匯入：

```
tridentctl import volume -n trident san_blog unmanaged_example_volume
-f pvc-import.yaml --no-manage
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
|                   NAME                  |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
| pvc-1fc999c9-ce8c-459c-82e4-ed4380a4b228 | 1.0 GiB | san-blog      |
block    | e3275890-7d80-4af6-90cc-c7a0759f555a | online | false   |
+-----------------------------------------+---------+--------------
+----------+-----------------------------------------+--------+---------+
```

如果您將 LUN 對應到與 Kubernetes 節點 IQN 共用相同 IQN 的 igroup，如下例所示，您將收到以下錯誤：`LUN already mapped to initiator(s) in this group`。您需要移除啟動器或取消映射 LUN 才能匯入磁碟區。

```
Vserver   Igroup       Protocol OS Type  Initiators
--------- ------------ -------- -------- ------------------------------------
svm0      k8s-nodename.example.com-fe5d36f2-cded-4f38-9eb0-c7719fc2f9f3
                       iscsi    linux    iqn.1994-05.com.redhat:4c2e1cf35e0
svm0      unmanaged-example-igroup
                       mixed    linux    iqn.1994-05.com.redhat:4c2e1cf35e0
```

元素

Trident 支援使用 `solidfire-san`驅動程式匯入 NetApp Element 軟體和 NetApp HCI 磁碟區。

Element 驅動程式支援重複的磁碟區名稱。但是，如果存在重複的磁碟區名稱，Trident 會傳回錯誤。因應措施是複製磁碟區、提供唯一的磁碟區名稱，然後匯入複製的磁碟區。

以下範例會在後端 `element_default` 匯入一個 `element-managed` volume。

```
tridentctl import volume element_default element-managed -f pvc-basic-
import.yaml -n trident -d

+----------------------------------------+--------+---------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE  | STORAGE CLASS |
PROTOCOL |             BACKEND UUID              | STATE  | MANAGED |
+----------------------------------------+--------+---------------
+----------+----------------------------------------+--------+---------+
| pvc-970ce1ca-2096-4ecd-8545-ac7edc24a8fe | 10 GiB | basic-element |
block    | d3ba047a-ea0b-43f9-9c42-e38e58301c49 | online | true    |
+----------------------------------------+--------+---------------
+----------+----------------------------------------+--------+---------+
```

**Azure NetApp Files**

Trident 支援使用 `azure-netapp-files` 驅動程式導入磁碟區。

若要匯入 Azure NetApp Files 磁碟區，請透過磁碟區路徑識別該磁碟區。磁碟區路徑是磁碟區匯出路徑中 `:/`後的部分。例如，如果掛載路徑為 `10.0.0.2:/importvol1`，則磁碟區路徑為 `importvol1`。

以下範例會在後端 `azurenetappfiles_40517` 匯入一個 `azure-netapp-files` volume，並使用 volume 路徑 `importvol1`。

```
tridentctl import volume azurenetappfiles_40517 importvol1 -f <path-to-
pvc-file> -n trident

+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS |
PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
| pvc-0ee95d60-fd5c-448d-b505-b72901b3a4ab | 100 GiB | anf-storage   |
file     | 1c01274f-d94b-44a3-98a3-04c953c9a51e | online | true    |
+----------------------------------------+---------+--------------
+----------+----------------------------------------+--------+---------+
```

#### Google Cloud NetApp Volumes

Trident 支援使用 `google-cloud-netapp-volumes` 驅動程式導入磁碟區。

以下範例使用磁碟區 `testvoleasiaeast1` 從後端 `backend-tbc-gcnv1` 匯入磁碟區。

```
tridentctl import volume backend-tbc-gcnv1 "testvoleasiaeast1" -f < path-
to-pvc> -n trident

+----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
|                   NAME                 |  SIZE   | STORAGE CLASS
| PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
| pvc-a69cda19-218c-4ca9-a941-aea05dd13dc0 |  10 GiB | gcnv-nfs-sc-
identity | file     | 8c18cdf1-0770-4bc0-bcc5-c6295fe6d837 | online | true
|
+----------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

以下範例示範如何在同一區域存在兩個磁碟區時匯入 `google-cloud-netapp-volumes`磁碟區：

```
tridentctl import volume backend-tbc-gcnv1
"projects/123456789100/locations/asia-east1-a/volumes/testvoleasiaeast1"
-f <path-to-pvc> -n trident

+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
|                   NAME                   |  SIZE   | STORAGE CLASS
| PROTOCOL |             BACKEND UUID             | STATE  | MANAGED |
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
| pvc-a69cda19-218c-4ca9-a941-aea05dd13dc0 |  10 GiB | gcnv-nfs-sc-
identity | file     | 8c18cdf1-0770-4bc0-bcc5-c6295fe6d837 | online | true
|
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

## 自訂磁碟區名稱和標籤

使用 Trident ，您可以為建立的磁碟區指派有意義的名稱和標籤。這有助於您識別磁碟區並將其輕鬆對應到相應的 Kubernetes 資源（ PVC ）。您也可以在後端定義模板，用於建立自訂磁碟區名稱和自訂標籤；您建立、匯入或複製的任何磁碟區都會遵循這些模板。

### 開始之前

支援自訂磁碟區名稱和標籤：

- Volume 建立、匯入和複製作業。
- 對於 `ontap-nas-economy` 驅動程式而言，只有 Qtree 磁碟區的名稱符合名稱範本。
- 對於 `ontap-san-economy` 驅動程式而言，只有 LUN 名稱符合名稱範本。

### 限制

- 自訂磁碟區名稱僅與 ONTAP 內部部署驅動程式相容。
- 僅 `ontap-san`、 `ontap-nas` 和 `ontap-nas-flexgroup` 驅動程式支援自訂標籤。
- 自訂磁碟區名稱不適用於現有磁碟區。

### 可自訂磁碟區名稱的關鍵行為

- 如果由於名稱範本中的語法無效而導致故障，則後端建立將失敗。但是，如果範本應用程式失敗，則磁碟區將根據現有的命名慣例命名。
- 當使用後端組態中的名稱範本命名磁碟區時，儲存前置字元不適用。任何所需的前置字元值都可以直接新增

至範本。

後端組態範例、名稱範本和標籤

可以在根層級和 / 或池層級定義自訂名稱範本。

根層級範例

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

資源池層級範例

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "backendName": "ontap-nfs-backend",
  "managementLIF": "<ip address>",
  "svm": "svm0",
  "username": "<admin>",
  "password": "<password>",
  "useREST": true,
  "storage": [
    {
      "labels": {
        "labelname": "label1",
        "name": "{{ .volume.Name }}"
      },
      "defaults": {
        "nameTemplate": "pool01_{{ .volume.Name }}_{{ .labels.cluster
}}_{{ .volume.Namespace }}_{{ .volume.RequestName }}"
      }
    },
    {
      "labels": {
        "cluster": "label2",
        "name": "{{ .volume.Name }}"
      },
      "defaults": {
        "nameTemplate": "pool02_{{ .volume.Name }}_{{ .labels.cluster
}}_{{ .volume.Namespace }}_{{ .volume.RequestName }}"
      }
    }
  ]
}
```

名稱範本範例

範例 **1**：

```
"nameTemplate": "{{ .config.StoragePrefix }}_{{ .volume.Name }}_{{
.config.BackendName }}"
```

範例 **2**：

```
"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**

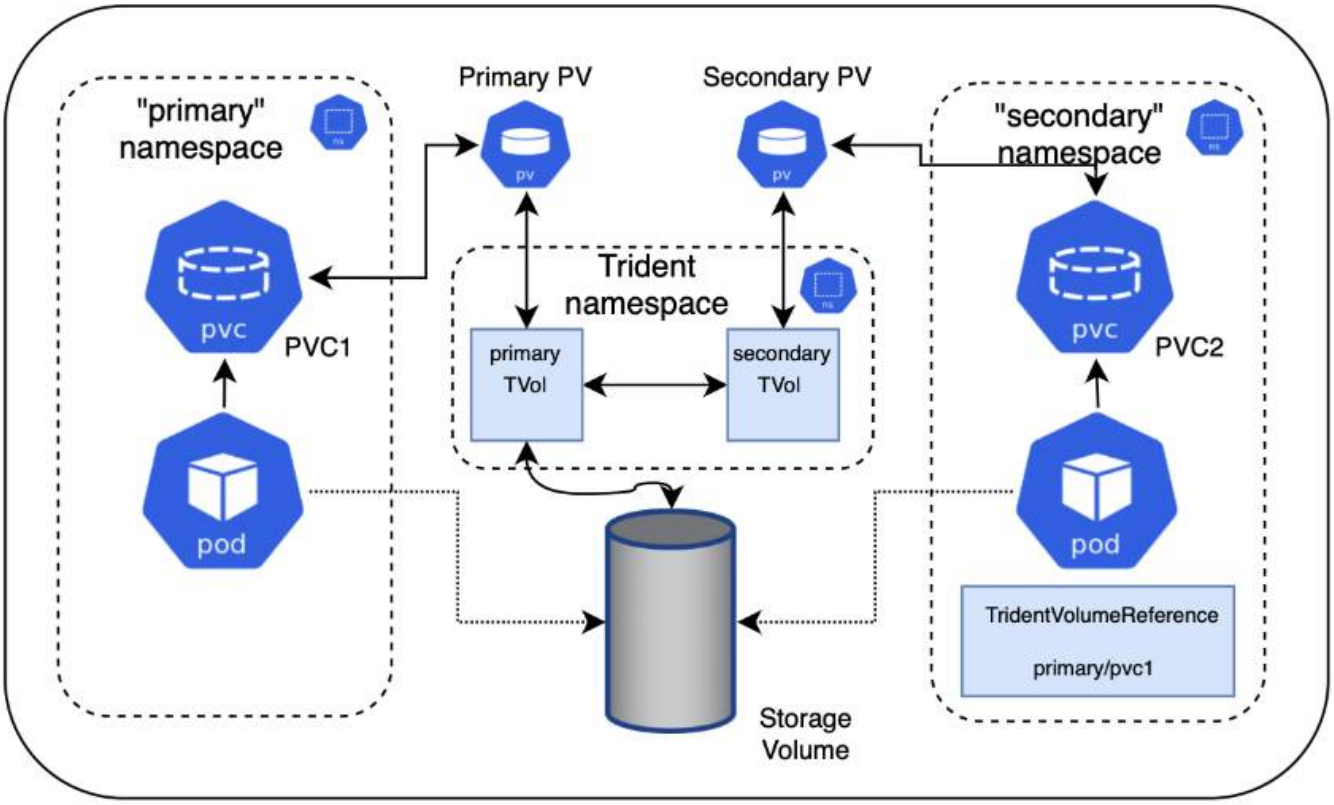
使用 Trident、您可以在主命名空間中建立磁碟區、並在一個或多個次要命名空間中共用該磁碟區。

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。

- 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
```

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、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：

- 訪問 "在命名空間之間共享磁碟區：迎接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。

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。

- 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。如需詳細資訊，請參閱 "ONTAP One 隨附的授權"。

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"。

  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）

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。

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。

> ⓘ `WaitForFirstConsumer` 綁定模式不需要拓樸標籤。它可以獨立於 CSI 拓樸功能使用。

您需要準備的項目

若要使用 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:"1e11e4a2108024935ecfcb2912226cedeafd99df",
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:"1e11e4a2108024935ecfcb2912226cedeafd99df",
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。

```
kubectl 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","kube
rnetes.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","kube
rnetes.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","kube
rnetes.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**

```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**

```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"
    }
  ]
}
```

`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-central1
managementLIF: 172.16.238.5
svm: nfs_svm
username: admin
password: password
supportedTopologies:
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-a
  - topology.kubernetes.io/region: us-central1
    topology.kubernetes.io/zone: us-central1-b
storage:
  - labels:
      workload: production
    supportedTopologies:
      - topology.kubernetes.io/region: us-central1
        topology.kubernetes.io/zone: us-central1-a
  - labels:
      workload: dev
    supportedTopologies:
      - topology.kubernetes.io/region: us-central1
        topology.kubernetes.io/zone: us-central1-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樸標籤、StorageClasses 可定義為包含拓樸資訊。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。

請參閱下列範例：

```
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：

```
kubectl create -f pvc.yaml
persistentvolumeclaim/pvc-san created
kubectl get pvc
NAME      STATUS    VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS
AGE
pvc-san   Pending                                      netapp-san-us-east1
2s
kubectl 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                         Message
  ----    ------                ----  ----                         -------
  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: vol1
    persistentVolumeClaim:
      claimName: pvc-san
  containers:
  - name: sec-ctx-demo
    image: busybox
    command: [ "sh", "-c", "sleep 1h" ]
    volumeMounts:
    - name: vol1
      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：

```
kubectl 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>
kubectl 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。

> 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 `<pvc-name>`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。

> 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: pvc1-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

1. 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。

     <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
```

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。

   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: pvc1
    volumeSnapshotName: pvc1-snapshot
  status:
    startTime: "2023-04-14T00:20:34Z"
    completionTime: "2023-04-14T00:20:37Z"
    state: Succeeded
kind: List
metadata:
  resourceVersion: ""
```

- 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.yam
l
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
```

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。

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`

> 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 。

  > 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: pvc1-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
```

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_volumegroupsnapshotcl
asses.yaml
kubectl apply -f https://raw.githubusercontent.com/kubernetes-
csi/external-snapshotter/release-
8.2/client/config/crd/groupsnapshot.storage.k8s.io_volumegroupsnapshotco
ntents.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
```

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"