使用 Trident

Trident

NetApp
February 02, 2026

This PDF was generated from https://docs.netapp.com/zh-tw/trident/trident-use/worker-node-prep.html on February 02, 2026. Always check docs.netapp.com for the latest.

目錄

使用 Trident . . . 1
準備工作節點 . . . 1
選擇適當的工具 . . . 1
節點服務探索 . . . 1
NFS磁碟區 . . . 2
iSCSI磁碟區 . . . 2
NVMe / TCP 磁碟區 . . . 6
FC 磁碟區上的 SCSI . . . 7
準備配置SMB磁碟區 . . . 9
設定及管理後端 . . . 10
設定後端 . . . 10
Azure NetApp Files . . . 10
Google Cloud NetApp Volumes . . . 29
設定NetApp HCI 一個不只是功能的SolidFire 後端 . . . 44
支援SAN驅動程式ONTAP . . . 49
ASNAS驅動程式ONTAP . . . 76
Amazon FSX for NetApp ONTAP 產品 . . . 110
使用kubecl建立後端 . . . 141
管理後端 . . . 147
建立及管理儲存類別 . . . 157
建立儲存類別 . . . 157
管理儲存類別 . . . 160
資源配置與管理磁碟區 . . . 162
配置 Volume . . . 162
展開Volume . . . 166
匯入磁碟區 . . . 177
自訂磁碟區名稱和標籤 . . . 187
跨命名空間共用NFS磁碟區 . . . 190
跨命名空間複製磁碟區 . . . 194
使用 SnapMirror 複寫磁碟區 . . . 196
使用「csi拓撲」 . . . 202
使用快照 . . . 210
使用磁碟區組快照 . . . 218

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已為Pod配置的磁碟區。若要準備工作節點，您必須根據您選擇的驅動程式來安裝 NFS ， iSCSI ， NVMe / TCP 或 FC 工具。

選擇適當的工具

如果您使用的是驅動程式組合、則應該安裝所有必要的驅動程式工具。Red Hat Enterprise Linux CoreOS （ RHCOS ）的最新版本預設會安裝工具。

NFS工具

"安裝 NFS 工具"如果您正在使用： `ontap-nas` ， `ontap-nas-economy` ， `ontap-nas-flexgroup` ， 或者 `azure-netapp-files`。

iSCSI工具

"安裝iSCSI工具" 如果您使用的是： `ontap-san`、 `ontap-san-economy`、 `solidfire-san`。

NVMe 工具

"安裝 NVMe 工具" 如果您正在使用 `ontap-san` 適用於透過 TCP （ NVMe / TCP ）傳輸協定的非揮發性記憶體高速（ NVMe ）。

> NetApp 建議使用 ONTAP 9.12 或更新版本來處理 NVMe / TCP 。

SCSI over FC 工具

請參閱"設定 FC 擴大機、 FC-NVMe SAN 主機的方法"如需設定 FC 和 FC-NVMe SAN 主機的詳細資訊，。

"安裝 FC 工具"如果您使用 `ontap-san` sanType `fcp` （ SCSI over FC ）。

- 要考慮的要點 * ： * OpenShift 和 KubeVirt 環境支援 SCSI over FC 。* Docker 不支援 SCSI over FC 。* iSCSI 自我修復不適用於 FC 上的 SCSI 。

SMB工具

"準備配置SMB磁碟區" 如果您正在使用： `ontap-nas` 為 SMB 提供卷。

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。

> 節點服務探索可識別探索到的服務、但無法保證服務已正確設定。相反地、沒有探索到的服務並不保證磁碟區掛載會失敗。

檢閱事件

Trident 會為節點建立事件、以識別探索到的服務。若要檢閱這些事件、請執行：

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

檢閱探索到的服務

Trident 會識別 Trident 節點 CR 上每個節點啟用的服務。若要檢視探索到的服務、請執行：

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

NFS磁碟區

使用作業系統的命令來安裝NFS工具。確保NFS服務在開機期間啟動。

RHEL 8以上

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

Ubuntu

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

安裝NFS工具之後、請重新啟動工作節點、以避免將磁碟區附加至容器時發生故障。

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、將其格式化、並將其裝載至 Pod 。

iSCSI自我修復功能

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、以識別所需的維修。Trident 會決定維修優先順序、以及何時優先執行維修。
3. 執行必要的修復、以將目前的iSCSI工作階段狀態恢復至所需的iSCSI工作階段狀態。

自我修復活動記錄位於個別 Dem隨 選裝置上的容器中 `trident-main`。若要檢視記錄、您必須在 Trident 安裝期間將其設 `debug`為「 true 」。

Trident iSCSI 自我修復功能有助於防止：

- 發生網路連線問題後、可能會發生過時或不正常的iSCSI工作階段。如果工作階段過時、 Trident 會在登出前等待七分鐘、以重新建立與入口網站的連線。

> 例如、如果在儲存控制器上旋轉CHAP機密、而網路失去連線、則舊的（*stal_*）CHAP機密可能會持續存在。自我修復可辨識此情況、並自動重新建立工作階段、以套用更新的CHAP機密。

- 遺失iSCSI工作階段
- 遺失LUN
- 升級 Trident 之前應考慮的要點 *
- 如果僅使用每個節點的 igroup （於 23.04+ 推出）、 iSCSI 自我修復將會為 SCSI 匯流排中的所有裝置啟動 SCSI 重新掃描。
- 如果僅使用後端範圍的 igroup （自 2004 年 23 日起已過時）、 iSCSI 自我修復將會針對 SCSI 匯流排中的確切 LUN ID 啟動 SCSI 重新掃描。
- 如果混合使用每個節點的 igroup 和後端範圍的 igroup 、 iSCSI 自我修復將會啟動 SCSI 重新掃描、以取得 SCSI 匯流排中的確切 LUN ID 。

安裝**iSCSI**工具

使用適用於您作業系統的命令來安裝iSCSI工具。

開始之前

- Kubernetes叢集中的每個節點都必須具有唯一的IQN。這是必要的先決條件。
- 若搭配使用RMCOS 4.5或更新版本、或其他與RHEL相容的Linux套裝作業系統 `solidfire-san` 驅動程式和元素OS 12.5或更早版本、請確定CHAP驗證演算法已在中設定為MD5 `/etc/iscsi/iscsid.conf`。元素12.7提供安全的FIPS相容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
  ```

- 當使用搭配 iSCSI PV 執行 RHEL/Red Hat Enterprise Linux CoreOS （ RHCOS ）的工作節點時，請在 StorageClass 中指定 `discard` mountOption 以執行內嵌空間回收。請參閱 "Red Hat 說明文件"。
- 確保您已升級至最新版本 `multipath-tools` 。

RHEL 8以上

1. 安裝下列系統套件：

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

2. 檢查iscsite-initier-utils版本是否為6.6.0.874-2.el7或更新版本：

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

3. 將掃描設為手動：

```
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
```

> 確保 `/etc/multipath.conf`包含 `find_multipaths no`在 `defaults`中。

5. 確保運行的是"iscsid"和"multipathd"：

```
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. 檢查開放式iSCSI版本是否為2.0.874-5ubuntu2・10或更新版本（適用於雙聲網路）或2.0.874-7.1ubuntu6.1或更新版本（適用於焦點）：

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

3. 將掃描設為手動：

```
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
```

> 確保 `/etc/multipath.conf`包含 `find_multipaths no`在 `defaults`中。

5. 確保已啟用並執行「open-iscsi」和「多路徑工具」：

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

> 對於Ubuntu 18.04、您必須先使用「iscsiadmd」探索目標連接埠、然後再啟動「open-iscsi」、iSCSI精靈才能啟動。您也可以修改「iSCSI」服務、以自動啟動「iscsid」。

設定或停用 **iSCSI** 自我修復

您可以設定下列 Trident iSCSI 自我修復設定、以修復過時的工作階段：

- *iSCSI 自我修復時間間隔 * ：決定啟動 iSCSI 自我修復的頻率（預設值： 5 分鐘）。您可以設定較小的數字、或設定較大的數字、將其設定為較常執行。

> 將 iSCSI 自我修復時間間隔設為 0 會完全停止 iSCSI 自我修復。我們不建議停用 iSCSI 自我修復功能；只有在 iSCSI 自我修復功能未如預期運作或無法進行偵錯時、才應停用 iSCSI 自我修復功能。

- *iSCSI 自我修復等待時間 * ：決定 iSCSI 自我修復等待的時間、再登出不正常的工作階段並再次嘗試登入（

預設值： 7 分鐘）。您可以將其設定為較大的數目、以便識別為不正常的工作階段必須等待較長時間才能登出、然後再嘗試重新登入、或是較小的數目來登出和較早登入。

掌舵

若要設定或變更 iSCSI 自我修復設定、請通過 `iscsiSelfHealingInterval` 和 `iscsiSelfHealingWaitTime` 在 helm 安裝或 helm 更新期間的參數。

以下範例將 iSCSI 自我修復間隔設為 3 分鐘、而自我修復等候時間設為 6 分鐘：

```
helm install trident trident-operator-100.2506.0.tgz --set
iscsiSelfHealingInterval=3m0s --set iscsiSelfHealingWaitTime=6m0s -n
trident
```

試用

若要設定或變更 iSCSI 自我修復設定、請通過 `iscsi-self-healing-interval` 和 `iscsi-self-healing-wait-time` 在 Tridentctl 安裝或更新期間的參數。

以下範例將 iSCSI 自我修復間隔設為 3 分鐘、而自我修復等候時間設為 6 分鐘：

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

NVMe / TCP 磁碟區

使用適用於您作業系統的命令來安裝 NVMe 工具。

- NVMe 需要 RHEL 9 或更新版本。
- 如果 Kubernetes 節點的核心版本太舊、或 NVMe 套件無法用於您的核心版本、您可能必須使用 NVMe 套件將節點的核心版本更新為一個。

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

安裝後、請使用命令確認 Kubernetes 叢集中的每個節點都有唯一的 NQN ：

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

Trident 會修改此 `ctrl_device_tmo`值、確保 NVMe 在故障時不會放棄路徑。請勿變更此設定。

FC 磁碟區上的 SCSI

您現在可以搭配 Trident 使用光纖通道（ FC ）傳輸協定，在 ONTAP 系統上配置及管理儲存資源。

先決條件

設定 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 "網路介面顯示" 。
2. 取得啟動器（主機）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。

   如需詳細資訊，請參閱重新輸入交換器廠商文件。

   如需詳細資訊，請參閱下列 ONTAP 文件：

   - "Fibre Channel和FCoE分區總覽"
   - "設定 FC 擴大機、 FC-NVMe SAN 主機的方法"

安裝 **FC** 工具

使用作業系統的命令來安裝FC工具。

- 當使用搭配 FC PV 執行 RHEL/Red Hat Enterprise Linux CoreOS （ RHCOS ）的工作節點時，請在 StorageClass 中指定 `discard` mountOption 以執行內嵌空間回收。請參閱 "Red Hat 說明文件"。

RHEL 8以上

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
```

> 確保 `/etc/multipath.conf`包含 `find_multipaths no`在 `defaults`中。

3. 確定 `multipathd`執行中：

```
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
```

> 確保 `/etc/multipath.conf`包含 `find_multipaths no`在 `defaults`中。

3. 確定 `multipath-tools`已啟用並正在執行：

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

準備配置SMB磁碟區

您可以使用以下方式設定 SMB 卷 `ontap-nas` 司機。

> 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 。若未設定上述任一種通訊協定、將導致 SMB 磁碟區建立失敗。

> `autoExportPolicy`不支援 SMB Volume 。

開始之前

在配置 SMB 磁碟區之前、您必須具備下列項目。

- Kubernetes叢集具備Linux控制器節點、以及至少一個執行Windows Server 2022的Windows工作節點。Trident 僅支援掛載至 Windows 節點上執行的 Pod 的 SMB 磁碟區。
- 至少有一個 Trident 機密包含您的 Active Directory 認證。產生機密 `smbcreds`：

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

- 設定為Windows服務的SCSI Proxy。若要設定 `csi-proxy`、請參閱 "GitHub：csi Proxy" 或 "GitHub：適用於Windows的SCSI Proxy" 適用於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、或 Trident 可以為您建立 SMB 共用。

   > Amazon FSX for ONTAP 需要 SMB 共享。

   您可以使用兩種方式之一來建立SMB管理共用區 "Microsoft管理主控台" 共享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、請建立共用的目錄路徑結構。

      。`vserver cifs share create` 命令會在共用建立期間檢查-path選項中指定的路徑。如果指定的路徑不存在、則命令會失敗。

   b. 建立與指定SVM相關的SMB共用區：

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

   c. 確認共用區已建立：

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

請參閱 "建立SMB共用區" 以取得完整詳細資料。

2. 建立後端時、您必須設定下列項目以指定SMB Volume。如需ONTAP 所有的FSXfor Sendbackend組態選項、請參閱 "FSX提供ONTAP 各種組態選項和範例"。

參數	說明	範例
smbShare	您可以指定下列其中一項：使用 Microsoft 管理主控台或 ONTAP CLI 建立的 SMB 共用名稱；允許 Trident 建立 SMB 共用的名稱；或將參數保留空白以防止共用磁碟區。對於內部部署 ONTAP 、此參數為選用項目。Amazon FSX 需要此參數才能支援 ONTAP 後端、且不可為空白。	smb-share
nasType	*必須設定為 smb.*如果為null、則預設為 nfs。	smb
《生態樣式》	新磁碟區的安全樣式。必須設定為 **ntfs** 或 **mixed** 適用於**SMB**磁碟區。	ntfs 或 mixed 適用於SMB磁碟區
「unixPermissions」	新磁碟區的模式。SMB磁碟區*必須保留為空白。*	"

設定及管理後端

設定後端

後端定義 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如何從該儲存系統配置磁碟區。

Trident 會自動從後端提供符合儲存類別所定義需求的儲存資源池。瞭解如何設定儲存系統的後端。

- "設定Azure NetApp Files 一個靜態後端"
- "設定 Google Cloud NetApp Volumes 後端"
- "設定NetApp HCI 一個不只是功能的SolidFire 後端"
- "使用ONTAP 功能不一的Cloud Volumes ONTAP NAS驅動程式來設定後端"
- "使用ONTAP 不支援的Cloud Volumes ONTAP SAN驅動程式來設定後端"
- "搭配 Amazon FSX for NetApp ONTAP 使用 Trident"

Azure NetApp Files

設定**Azure NetApp Files** 一個靜態後端

您可以將 Azure NetApp Files 設定為 Trident 的後端。您可以使用 Azure NetApp Files 後端連接 NFS 和 SMB 磁碟區。Trident 也支援使用 Azure Kubernetes Services （ aks ） 叢集的託管身分識別來進行認證管理。

Azure NetApp Files 驅動程式詳細資料

Trident 提供下列 Azure NetApp Files 儲存驅動程式、可與叢集進行通訊。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	**Volume**模式	支援的存取模式	支援的檔案系統
《azure-NetApp-fil形》	NFS 中小企業	檔案系統	Rwo 、 ROX 、 rwx 、 RWOP	`nfs`、 `smb`

考量

- Azure NetApp Files 服務不支援小於 50 GiB 的磁碟區。如果要求較小的磁碟區、 Trident 會自動建立 50-GiB 磁碟區。
- Trident 僅支援掛載至 Windows 節點上執行的 Pod 的 SMB 磁碟區。

管理的身分識別

Trident 支援 "託管身分識別"Azure Kubernetes 服務叢集。若要善用託管身分識別所提供的簡化認證管理功能、您必須具備：

- 使用 aks 部署的 Kubernetes 叢集
- 在 aks Kubernetes 叢集上設定的託管身分識別
- 安裝的 Trident ，其中包括 `cloudProvider`要指定 `"Azure"`的。

Trident運算子

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

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

掌舵

以下範例使用環境變數將 Trident Set 安裝 `cloudProvider`至 `Azure` `$CP`：

```
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 並將旗標設定 `cloudProvider`為 `Azure`：

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

雲端身分識別

雲端身分識別可讓 Kubernetes Pod 以工作負載身分驗證來存取 Azure 資源、而非提供明確的 Azure 認證。

若要在 Azure 中使用雲端身分識別、您必須具備：

- 使用 aks 部署的 Kubernetes 叢集
- 在 OKS Kubernetes 叢集上設定的工作負載識別和 oidc-c發行 者
- 安裝的 Trident 、其中包含 `cloudProvider`指定 `"Azure"`及 `cloudIdentity`指定工作負載身分識別的

Trident運算子

若要使用 Trident 運算子安裝 Trident 、請編輯 `tridentorchestrator_cr.yaml`以設定為 `cloudProvider "Azure"`、並設定 `cloudIdentity`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
```

掌舵

使用下列環境變數設定 * 雲端供應商（CP） * 和 * 雲端身分識別（CI） * 旗標的值：

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

下列範例使用環境變數安裝 Trident 並將設定 `cloudProvider`為 `Azure` `$CP` 、並使用環境變數 `$CI`設定 `cloudIdentity`：

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

<code>tridentctl</code>

使用下列環境變數設定 * 雲端供應商 * 和 * 雲端 IDENTITY * 旗標的值：

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

以下範例會安裝 Trident 並將旗標設定 `cloud-provider`為 `$CP`、和 `cloud-identity` `$CI`：

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

準備設定**Azure NetApp Files** 一個功能完善的後端

在您設定Azure NetApp Files 完後端功能之前、您必須確保符合下列要求。

NFS 和 **SMB** 磁碟區的必要條件

如果您是第一次使用 Azure NetApp Files 、或是在新位置使用、則必須先進行一些初始設定、才能設定 Azure NetApp Files 並建立 NFS Volume 。請參閱 "Azure：設定Azure NetApp Files 功能以建立NFS Volume"。

若要設定及使用 "Azure NetApp Files" 後端、您需要下列項目：

> - `subscriptionID`、`tenantID`、`clientID`、`location`和 `clientSecret` 在 AKS 叢集上使用託管身分識別時為選用項目。
> - `tenantID`、`clientID`和 `clientSecret` 在 AKS 叢集上使用雲端身分識別時為選用項目。

- 容量集區。請參閱 "Microsoft ：為 Azure NetApp Files 建立容量集區"。
- 委派給 Azure NetApp Files 的子網路。請參閱 "Microsoft ：將子網路委派給 Azure NetApp Files"。
- Azure訂閱提供的「SubscriptionID」Azure NetApp Files （含功能不支援的功能）。
- `tenantID`、`clientID`和 `clientSecret` 從 "應用程式註冊" 在Azure Active Directory中、具備Azure NetApp Files 充分的權限執行此功能。應用程式登錄應使用下列其中一項：
  - 擁有者或貢獻者角色 "由Azure預先定義"。
  - "自訂貢獻者角色"(`assignableScopes`（在訂閱級別），具有以下權限，僅限於 Trident 所需的權限。建立自訂角色之後"使用Azure入口網站指派角色"，。

自訂參與者角色

```
{
  "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": []
      }
    ]
  }
}
```

- Azure `location` 至少包含一個 "委派的子網路"。從Trident 22.01起 `location` 參數是後端組態檔最上層的必填欄位。會忽略虛擬資源池中指定的位置值。
- 以供使用 `Cloud Identity`請取得 `client ID` 從 "使用者指派的託管身分識別" 並在中指定該 ID `azure.workload.identity/client-id: xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx`。

SMB磁碟區的其他需求

若要建立 SMB Volume 、您必須具備：

- Active Directory 已設定並連線至 Azure NetApp Files 。請參閱 "Microsoft ：建立及管理 Azure NetApp Files 的 Active Directory 連線"。
- Kubernetes叢集具備Linux控制器節點、以及至少一個執行Windows Server 2022的Windows工作節點。Trident 僅支援掛載至 Windows 節點上執行的 Pod 的 SMB 磁碟區。
- 至少有一個 Trident 機密包含您的 Active Directory 認證、以便 Azure NetApp Files 能夠驗證至 Active Directory 。產生機密 `smbcreds`：

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

- 設定為Windows服務的SCSI Proxy。若要設定 `csi-proxy`、請參閱 "GitHub：csi Proxy" 或 "GitHub：適用於Windows的SCSI Proxy" 適用於Windows上執行的Kubernetes節點。

列舉後端組態選項與範例**Azure NetApp Files**

瞭解 Azure NetApp Files 的 NFS 和 SMB 後端組態選項、並檢閱組態範例。

後端組態選項

Trident 使用後端組態（子網路、虛擬網路、服務層級和位置）、在所要求位置的可用容量集區上建立 Azure NetApp Files Volume 、並符合所要求的服務層級和子網路。

Azure NetApp Files 後端提供這些組態選項。

參數	說明	預設
「分度」		永遠為1
「torageDriverName」	儲存驅動程式名稱	「Azure - NetApp-Files」
「後端名稱」	自訂名稱或儲存後端	驅動程式名稱+「_」+隨機字元
《訂閱ID》	Azure訂閱的訂閱ID 在 AKS 叢集上啟用受管理的身分識別時為選用項目。	
「TenantId」	應用程式註冊的租戶ID 在 AKS 叢集上使用託管身分識別或雲端身分識別時、為選用項目。	
"clientID"	應用程式註冊的用戶端ID 在 AKS 叢集上使用託管身分識別或雲端身分識別時、為選用項目。	
「客戶機密」	應用程式註冊的用戶端機密 在 AKS 叢集上使用託管身分識別或雲端身分識別時、為選用項目。	
《服務層級》	其中一種是「標準」、「高級」或「超高」	""（隨機）
位置	要建立新磁碟區的Azure位置名稱 在 AKS 叢集上啟用受管理的身分識別時為選用項目。	
"來源群組"	用於篩選已探索資源的資源群組清單	「[]」（無篩選器）
《netappAccounts》	篩選探索資源的NetApp帳戶清單	「[]」（無篩選器）
《容量Pools》	用於篩選已探索資源的容量集區清單	「[]」（無篩選器、隨機）
「虛擬化網路」	具有委派子網路的虛擬網路名稱	"
《Subnet》	委派給「icrosoft.Netapp/volumes`」的子網路名稱	"

參數	說明	預設
《網路功能》	Volume的vnet功能集可能是 `Basic` 或 `Standard`。並非所有地區都提供網路功能、可能必須在訂閱中啟用。指定 `networkFeatures` 如果未啟用此功能、則會導致磁碟區資源配置失敗。	"
「nfsMountOptions」	精細控制NFS掛載選項。SMB磁碟區已忽略。若要使用NFS 4.1版掛載磁碟區、請包含 `nfsvers=4` 在以逗號分隔的掛載選項清單中、選擇NFS v4.1。儲存類別定義中設定的掛載選項會覆寫在後端組態中設定的掛載選項。	"nfsves=3"
《限制Volume大小》	如果要求的磁碟區大小高於此值、則資源配置失敗	""（預設不強制執行）
「DebugTraceFlags」	疑難排解時要使用的偵錯旗標。範例：「\{"API"：假、「方法」：真、「探索」：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`	設定NFS或SMB磁碟區建立。選項包括 `nfs`、 `smb` 或null。NFS磁碟區的預設值設為null。	`nfs`
`supportedTopologies`	代表此後端所支援的區域和區域清單。如需詳細資訊、請 "使用「csi拓撲」"參閱。	
`qosType`	表示 QoS 類型：自動或手動。	汽車
`maxThroughput`	設定允許的最大吞吐量，單位為 MiB/秒。僅支援手動 QoS 容量池。	`4 MiB/sec`

> 如需網路功能的詳細資訊、請參閱 "設定Azure NetApp Files 適用於某個聲音量的網路功能"。

必要的權限與資源

如果您在建立 PVC 時收到「找不到容量集區」錯誤，您的應用程式註冊可能沒有相關的必要權限和資源（子網路，虛擬網路，容量集區）。如果啟用除錯功能、 Trident 會記錄建立後端時所探索到的 Azure 資源。確認使用的角色是否適當。

的值 `resourceGroups`、 `netappAccounts`、 `capacityPools`、 `virtualNetwork`和 `subnet` 可以使用簡短或完整名稱來指定。在大多數情況下、建議使用完整名稱、因為短名稱可以符合多個名稱相同的資源。

> 如果虛擬網路與Azure NetApp Files(ANF) 儲存帳戶位於不同的資源群組中，則在設定後端資源群組清單時，請為虛擬網路指定資源群組。

。 `resourceGroups`、 `netappAccounts`和 `capacityPools` 值是篩選器、可將探索到的資源集合限制在此儲存後端可用的資源、並可任意組合指定。完整名稱格式如下：

類型	格式
資源群組	<資源群組>
NetApp帳戶	資源群組//<NetApp帳戶>
容量資源池	資源群組//<NetApp帳戶>/<容量資源池>
虛擬網路	資源群組//<虛擬網路>
子網路	資源群組//<虛擬網路>/<子網路>

Volume資源配置

您可以在組態檔的特殊區段中指定下列選項、以控制預設的Volume資源配置。請參閱 [組態範例] 以取得詳細資料。

參數	說明	預設
「匯出規則」	匯出新磁碟區的規則。`exportRule` 必須是以逗號分隔的清單、以CIDR表示法列出所有的IPv4位址或IPv4子網路組合。SMB磁碟區已忽略。	「0.00.0.0/0」
「napshotDir	控制.snapshot目錄的可見度	針對 NFSv3 的 NFSv4 "false" 為 "true"
《大小》	新磁碟區的預設大小	100公克
「unixPermissions」	新磁碟區的UNIX權限（4個八進位數字）。SMB磁碟區已忽略。	""（預覽功能、訂閱時需要白名單）

組態範例

下列範例顯示基本組態、讓大部分參數保留預設值。這是定義後端最簡單的方法。

最小組態

這是絕對最低的後端組態。使用此組態、 Trident 會探索您在設定位置中委派給 Azure NetApp Files 的所有 NetApp 帳戶、容量集區和子網路、並隨機將新磁碟區放在其中一個集區和子網路上。由於省略、因此 `nasType` `nfs` 會套用預設值、而後端會為 NFS 磁碟區進行資源配置。

當您剛開始使用 Azure NetApp Files 並試用時、這項組態是理想的選擇、但實際上您會想要為您所配置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
```

管理的身分識別

此後端組態已不再如此 `subscriptionID`、 `tenantID`、 `clientID`和 `clientSecret`，使用託管身分識別時為選用功能。

```
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
```

雲端身分識別

此後端組態已不再如此 `tenantID`、 `clientID`和 `clientSecret`（使用雲端身分識別時為選用功能）。

```
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
```

具有容量集區篩選器的特定服務層級組態

此後端組態會將磁碟區放置在 Azure 的位置、並置於 `eastus`容量集區中 `Ultra`。Trident 會自動探索該位置中委派給 Azure NetApp Files 的所有子網路、並隨機在其中一個磁碟區上放置新的磁碟區。

```
---
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
```

具有手動 **QoS** 容量池的後端範例

此後端配置將磁碟區放置在 Azure 中 `eastus`具有手動 QoS 容量池的位置。

```
---
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

此後端組態可進一步將磁碟區放置範圍縮小至單一子網路、並修改部分Volume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"
```

虛擬集區組態

此後端組態可在單一檔案中定義多個儲存集區。當您有多個容量集區支援不同的服務層級、而且想要在Kubernetes中建立代表這些層級的儲存類別時、這很有用。虛擬資源池標籤是用來區分資源池的依據 `performance`。

```
---
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 可根據地區和可用性區域、為工作負載提供更多資源。 `supportedTopologies`此後端組態中的區塊用於提供每個後端的區域和區域清單。此處指定的區域和區域值必須符合每個 Kubernetes 叢集節點上標籤的區域和區域值。這些區域和區域代表可在儲存類別中提供的允許值清單。對於包含後端所提供區域和區域子集的儲存類別、 Trident 會在所述區域和區域中建立磁碟區。如需詳細資訊、請 "使用「csi拓撲」"參閱。

```
---
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

以下內容 `StorageClass` 定義請參閱上述儲存資源池。

使用的範例定義 `parameter.selector` 欄位

使用 `parameter.selector` 您可以為每個項目指定 `StorageClass` 用於裝載磁碟區的虛擬集區。該磁碟區會在所選的資源池中定義各個層面。

```yaml
---
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磁碟區的定義範例

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認證資料。

預設命名空間的基本組態

```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` 支援SMB磁碟區的集區篩選器。 `nasType: nfs` 或 `nasType: null` NFS集區的篩選器。

建立後端

建立後端組態檔之後、請執行下列命令：

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

如果後端建立失敗、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
```

識別並修正組態檔的問題之後、您可以再次執行create命令。

Google Cloud NetApp Volumes

設定 **Google Cloud NetApp Volumes** 後端

您現在可以將 Google Cloud NetApp Volumes 設定為 Trident 的後端。您可以使用 Google Cloud NetApp Volumes 後端來附加 NFS 和 SMB 磁碟區。

Google Cloud NetApp Volumes 驅動程式詳細資料

Trident 提供 `google-cloud-netapp-volumes`與叢集通訊的驅動程式。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	**Volume**模式	支援的存取模式	支援的檔案系統
`google-cloud-netapp-volumes`	NFS 中小企業	檔案系統	Rwo 、 ROX 、 rwx 、 RWOP	`nfs`、 `smb`

GKE 的雲端身分識別

雲端身分識別可讓 Kubernetes Pod 以工作負載身分驗證來存取 Google Cloud 資源、而非提供明確的 Google Cloud 身分證明。

若要在 Google Cloud 中善用雲端身分識別、您必須具備：

- 使用 GKE 部署的 Kubernetes 叢集。
- 在節點集區上設定的 GKE 叢集和 GKE 中繼資料伺服器上設定的工作負載身分識別。
- 具有 Google Cloud NetApp Volumes 管理員（角色 / NetApp 。管理）角色或自訂角色的 GCP 服務帳戶。
- 安裝的 Trident 包含 cloudProvider ，可指定「 GCP 」和 cloudIdentity ，以指定新的 GCP 服務帳戶。以下是一個範例。

Trident運算子

若要使用 Trident 運算子安裝 Trident 、請編輯 `tridentorchestrator_cr.yaml`以設定為 `cloudProvider "GCP"`、並設定 `cloudIdentity`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'
```

掌舵

使用下列環境變數設定 * 雲端供應商（CP） * 和 * 雲端身分識別（CI） * 旗標的值：

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

以下範例使用環境變數安裝 Trident 並設定 `cloudProvider`為 `GCP` `$CP`、並使用環境變數 `$ANNOTATION`設定 `cloudIdentity`：

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

<code>tridentctl</code>

使用下列環境變數設定 * 雲端供應商 * 和 * 雲端 IDENTITY * 旗標的值：

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

以下範例會安裝 Trident 並將旗標設定 `cloud-provider`為 `$CP`、和 `cloud-identity` `$ANNOTATION`：

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

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

在您設定 Google Cloud NetApp Volumes 後端之前、您必須確保符合下列需求。

NFS Volume 的必要條件

如果您是第一次使用 Google Cloud NetApp Volumes 、或是在新位置使用、則需要進行一些初始設定、才能設定 Google Cloud NetApp Volumes 並建立 NFS Volume 。請參閱 "開始之前"。

在設定 Google Cloud NetApp Volumes 後端之前、請先確認您擁有下列項目：

- 使用 Google Cloud NetApp Volumes 服務設定的 Google Cloud 帳戶。請參閱 "Google Cloud NetApp Volumes"。
- Google Cloud 帳戶的專案編號。請參閱 "識別專案"。
- 具有 Volumes Admin （ NetApp Volume 管理）角色的 Google Cloud 服務帳戶 (`roles/netapp.admin`。請參閱 "身分識別與存取管理角色與權限"。
- 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 "儲存資源池總覽"。

如需如何設定 Google Cloud NetApp Volumes 存取權限的詳細資訊、請 "設定 Google Cloud NetApp Volumes 的存取權"參閱。

Google Cloud NetApp Volumes 後端組態選項和範例

瞭解 Google Cloud NetApp Volumes 的後端組態選項，並檢閱組態範例。

後端組態選項

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

參數	說明	預設
「分度」		永遠為1
「torageDriverName」	儲存驅動程式名稱	的值 `storageDriverName` 必須指定為「 googoogle 雲端 -NetApp-Volumes 」。
「後端名稱」	（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`	選用參數、用於指定用於建立磁碟區的儲存資源池。	
「ProjectNumber」	Google Cloud帳戶專案編號。此值可在Google Cloud入口網站首頁找到。	
位置	Trident 建立 GCNV Volume 的 Google Cloud 位置。建立跨區域 Kubernetes 叢集時、在中建立的磁碟區 `location` 可用於跨多個 Google Cloud 區域的節點上排程的工作負載。跨區域流量會產生額外成本。	

參數	說明	預設
「apiKey」	具有此角色的 Google Cloud 服務帳戶的 API 金鑰 `netapp.admin` 。其中包含Google Cloud服務帳戶私密金鑰檔案（逐字複製到後端組態檔）的JSON-格式內容。 `apiKey`必須包含下列金鑰的金鑰值配對：`type` `project_id`、、 `client_email`、 `client_id`、 `auth_uri` `token_uri` `auth_provider_x509_cert_url`、和 `client_x509_cert_url`。	
「nfsMountOptions」	精細控制NFS掛載選項。	"nfsves=3"
《限制Volume大小》	如果要求的磁碟區大小高於此值、則資源配置失敗。	""（預設不強制執行）
《服務層級》	儲存池及其磁碟區的服務層級。這些值包括 `flex`、 `standard`、 `premium`或 `extreme`。	
《標籤》	套用到磁碟區的任意JSON-格式化標籤集	"
網路	用於 GCNV Volume 的 Google Cloud 網路。	
「DebugTraceFlags」	疑難排解時要使用的偵錯旗標。範例：`{"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`	設定NFS或SMB磁碟區建立。選項包括 `nfs`、 `smb` 或null。NFS磁碟區的預設值設為null。	`nfs`
`supportedTopologies`	代表此後端所支援的區域和區域清單。如需詳細資訊、請 "使用「csi拓撲」"參閱。例如：`supportedTopologies: - topology.kubernetes.io/region: asia-east1 topology.kubernetes.io/zone: asia-east1-a`	

Volume資源配置選項

您可以在中控制預設的Volume資源配置 `defaults` 組態檔的一節。

參數	說明	預設
「匯出規則」	新磁碟區的匯出規則。必須是以逗號分隔的任何 IPv4 位址組合清單。	「0.00.0.0/0」
「napshotDir	存取「.snapshot」目錄	針對 NFSv3 的 NFSv4 "false" 為 "true"
「快照保留區」	保留給快照的磁碟區百分比	"（接受預設值 0 ）
「unixPermissions」	新磁碟區的UNIX權限（4個八進位數字）。	"

組態範例

下列範例顯示基本組態、讓大部分參數保留預設值。這是定義後端最簡單的方法。

最小組態

這是絕對最低的後端組態。有了這項組態、 Trident 會探索您在設定位置中委派給 Google Cloud NetApp Volumes 的所有儲存池、並隨機將新磁碟區放在其中一個集區上。由於省略、因此 `nasType` `nfs` 會套用預設值、而後端會為 NFS 磁碟區進行資源配置。

當您剛開始使用 Google Cloud NetApp Volumes 並試用時、這項組態非常理想、但實際上您很可能需要為您所配置的 Volume 提供額外的範圍。

```yaml
---
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 磁碟區的組態

```
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

```yaml
---
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
```

虛擬集區組態

此後端組態會在單一檔案中定義多個虛擬集區。虛擬集區是在一節中定義 `storage` 。當您有多個儲存集區支援不同的服務層級、而且您想要在 Kubernetes 中建立代表這些層級的儲存類別時、這些功能就很有用。虛擬集區標籤用於區分集區。例如、在下面的範例中、 `performance` 標籤和 `serviceLevel` 類型是用來區分虛擬集區。

您也可以將某些預設值設定為適用於所有虛擬集區、並覆寫個別虛擬集區的預設值。在下列範例中 `snapshotReserve` 、並 `exportRule` 做為所有虛擬集區的預設值。

如需詳細資訊、請 "虛擬資源池"參閱。

```
---
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 的雲端身分識別

```
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 可根據地區和可用性區域、為工作負載提供更多資源。 `supportedTopologies`此後端組態中的區塊用於提供每個後端的區域和區域清單。此處指定的區域和區域值必須符合每個 Kubernetes 叢集節點上標籤的區域和區域值。這些區域和區域代表可在儲存類別中提供的允許值清單。對於包含後端所提供區域和區域子集的儲存類別、 Trident 會在所述區域和區域中建立磁碟區。如需詳細資訊、請 "使用「csi拓撲」"參閱。

```
---
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？

建立後端組態檔之後、請執行下列命令：

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

若要確認後端已成功建立、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、表示後端組態有問題。您可以使用命令來描述後端 `kubectl get tridentbackendconfig <backend-name>` 、或是執行下列命令來檢視記錄以判斷原因：

```
tridentctl logs
```

識別並修正組態檔的問題之後、您可以刪除後端、然後再次執行 create 命令。

儲存類別定義

以下是上述後端的基本 `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"
```

- 使用欄位的範例定義 `parameter.selector` ： *

使用、 `parameter.selector` 您可以為用於裝載 Volume 的每個指定 `StorageClass` "虛擬集區" 。該磁碟區會在所選的資源池中定義各個層面。

```yaml
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 "建立儲存類別"參閱。

SMB磁碟區的定義範例

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 認證。任何具有任何 / 無權限的 Active Directory 使用者 / 密碼都可用於節點階段密碼。

預設命名空間的基本組態

```yaml
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

```yaml
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

```yaml
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 支援SMB磁碟區的集區篩選器。 nasType: nfs 或 nasType: null NFS集區的篩選器。

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 是否受限、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** 一個不只是功能的**SolidFire** 後端

瞭解如何在 Trident 安裝中建立和使用元素後端。

元素驅動程式詳細資料

Trident 提供 `solidfire-san`儲存驅動程式以與叢集通訊。支援的存取模式包括： *ReadWriteOnce* （rwo）、 *ReadOnlyMany* （*ROX*）、 *_ReadWriteMany* （*rwx*）、 *_ReadWriteOncePod* （*RWOP*）。

> `solidfire-san`儲存驅動程式支援 _file_ 和 _block_ 磁碟區模式。對於 `Filesystem` volemode 、 Trident 會建立一個 Volume 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	**Volume**模式	支援的存取模式	支援的檔案系統
「olidfire - 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	**Volume**模式	支援的存取模式	支援的檔案系統
「olidfire - san」	iSCSI	檔案系統	RWO 、 RWOP	《xfs》、《ext3》、《ext4》

開始之前

在建立元素後端之前、您需要下列項目。

- 支援的儲存系統、可執行Element軟體。
- 提供給NetApp HCI / SolidFire叢集管理員或租戶使用者的認證、以管理磁碟區。
- 您所有的Kubernetes工作節點都應該安裝適當的iSCSI工具。請參閱 "工作節點準備資訊"。

後端組態選項

如需後端組態選項、請參閱下表：

參數	說明	預設
「分度」		永遠為1
「torageDriverName」	儲存驅動程式名稱	永遠為「 SolidFire - SAN 」
「後端名稱」	自訂名稱或儲存後端	「 SolidFire _ 」 + 儲存（ iSCSI ）IP 位址
端點	MVIP、適用於SolidFire 採用租戶認證的不含用戶身分證明的叢集	
《VIP》	儲存設備（iSCSI）IP位址和連接埠	
《標籤》	套用到磁碟區的任意JSON-格式化標籤集。	"
《天王名稱》	要使用的租戶名稱（如果找不到、請建立）	
《初始器IFACE》	將iSCSI流量限制在特定的主機介面	"預設"
《UseCHAP》	使用 CHAP 驗證 iSCSI 。Trident 使用 CHAP 。	是的
《存取群組》	要使用的存取群組ID清單	尋找名為「 Trident 」的存取群組 ID
《類型》	QoS規格	
《限制Volume大小》	如果要求的磁碟區大小高於此值、則資源配置失敗	""（預設不強制執行）
「DebugTraceFlags」	疑難排解時要使用的偵錯旗標。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**：的後端組態 `solidfire-san` 三種磁碟區類型的驅動程式

此範例顯示使用CHAP驗證的後端檔案、並建立具有特定QoS保證的三種Volume類型模型。您很可能會使用「IOPS」儲存類別參數來定義儲存類別、以使用每個類別。

```
---
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**：的後端與儲存類別組態 `solidfire-san` 驅動程式與虛擬資源池

此範例顯示使用虛擬資源池設定的後端定義檔、以及參照這些資源池的StorageClass。

Trident 會在資源配置時、將儲存池上的標籤複製到後端儲存 LUN 。為了方便起見、儲存管理員可以針對每個虛擬資源池定義標籤、並依標籤將磁碟區分組。

在下圖所示的範例後端定義檔中、會針對所有設定的儲存資源池設定特定的預設值 `type` 銀級。虛擬資源池是在中定義的 `storage` 區段。在此範例中、有些儲存資源池會自行設定類型、有些資源池則會覆寫上述預設值。

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

```yaml
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
```

下列StorageClass定義是指上述虛擬資源池。使用 `parameters.selector` 欄位中、每個StorageClass會呼叫

哪些虛擬資源池可用於裝載Volume。磁碟區將會在所選的虛擬資源池中定義各個層面。

第一個 StorageClass (`solidfire-gold-four`）將映射到第一個虛擬池。這是唯一提供黃金級效能的集區 `Volume Type QoS`。Last StorageClass (`solidfire-silver`（最後一個 StorageClass ）調用任何提供銀牌性能的存儲池。Trident 會決定要選取哪個虛擬集區、並確保符合儲存需求。

```
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

---
```

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

如需詳細資訊、請參閱

- "Volume存取群組"

支援SAN驅動程式ONTAP

ONTAP SAN 驅動程式概觀

深入瞭解如何使用ONTAP 支援功能的功能和功能性SAN驅動程式來設定功能性的後端。ONTAP Cloud Volumes ONTAP

ONTAP SAN 驅動程式詳細資料

Trident 提供下列 SAN 儲存驅動程式、可與 ONTAP 叢集進行通訊。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	**Volume**模式	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 檔案系統磁碟區模式中無法使用 Rox 和 rwx 。	《xfs》、《ext3》、《ext4》
「ONTAP-SAN」	NVMe / TCP 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

驅動程式	傳輸協定	**Volume**模式	支援的存取模式	支援的檔案系統
「ONTAP-SAN」	NVMe / TCP 請參閱 NVMe / TCP 的其他考量事項。	檔案系統	RWO 、 RWOP 檔案系統磁碟區模式中無法使用 Rox 和 rwx 。	《xfs》、《ext3》、《ext4》
《ONTAP-san經濟》	iSCSI	區塊	Rwo 、 ROX 、 rwx 、 RWOP	無檔案系統；原始區塊裝置
《ONTAP-san經濟》	iSCSI	檔案系統	RWO 、 RWOP 檔案系統磁碟區模式中無法使用 Rox 和 rwx 。	《xfs》、《ext3》、《ext4》

- 使用 `ontap-san-economy` 只有持續磁碟區使用量計數預期會高於 "支援的 ONTAP Volume 限制"。
- 使用 `ontap-nas-economy` 只有持續磁碟區使用量計數預期會高於 "支援的 ONTAP Volume 限制" 和 `ontap-san-economy` 無法使用驅動程式。
- 請勿使用 `ontap-nas-economy` 如果您預期需要資料保護、災難恢復或行動性、
- NetApp 不建議在所有 ONTAP 驅動程式中使用 FlexVol 自動擴充， ONTAP SAN 除外。作為因應措施， Trident 支援使用快照保留，並據此擴充 FlexVol 磁碟區。

使用者權限

Trident 預期會以 ONTAP 或 SVM 管理員的身分執行、通常使用叢集使用者或 `vsadmin` SVM 使用者、或是使用 `admin`具有相同角色的不同名稱的使用者。對於用於 `NetApp ONTAP` 部署的 `Amazon FSX` 、 `Trident` 預期會以 `ONTAP` 或 `SVM` 管理員的身分、使用叢集使用者或 `vsadmin` SVM 使用者、或是具有相同角色的不同名稱的使用者來執行 `fsxadmin`。 `fsxadmin`使用者只能有限地取代叢集管理使用者。

如果您使用此 `limitAggregateUsage`參數、則需要叢集管理權限。將 Amazon FSX for NetApp ONTAP 搭配 Trident 使用時、此 `limitAggregateUsage`參數將無法與和 `fsxadmin`使用者帳戶搭配 `vsadmin`使用。如果您指定此參數、組態作業將會失敗。

雖然可以在 ONTAP 中建立更具限制性的角色、讓 Trident 驅動程式可以使用、但我們不建議這樣做。Trident的大多數新版本都會呼叫額外的API、而這些API必須納入考量、使升級變得困難且容易出錯。

NVMe / TCP 的其他考量事項

Trident 支援使用驅動程式的非揮發性記憶體高速（ NVMe ）傳輸協定 `ontap-san`、包括：

- IPv6
- NVMe 磁碟區的快照和複本
- 調整 NVMe 磁碟區大小

- 匯入在 Trident 之外建立的 NVMe Volume 、以便 Trident 管理其生命週期
- NVMe 原生多重路徑
- K8s 節點正常或不正常關機（ 24.06 ）

Trident 不支援：

- NVMe 原生支援的 DH-HMAC-CHAP
- 裝置對應工具（ 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** 支援的**SAN**驅動程式來設定後端

瞭解使用 ONTAP SAN 驅動程式設定 ONTAP 後端的需求和驗證選項。

需求

對於所有 ONTAP 後端，Trident 要求至少將一個聚合分配給 SVM。

> "ASA r2 系統"與其他ONTAP系統（ASA、 AFF和FAS）在儲存層的實作上有所不同。在ASA r2 系統中，使用儲存可用區而不是聚合。請參閱"這"知識庫文章，介紹如何在ASA r2 系統中將聚合指派給 SVM。

請記住、您也可以執行多個驅動程式、並建立指向一個或多個驅動程式的儲存類別。例如、您可以設定使用「ONTAP-SAN」驅動程式的「SAN開發」類別、以及使用「ONTAP-SAN經濟」類別的「SAN預設」類別。

您所有的Kubernetes工作節點都必須安裝適當的iSCSI工具。請參閱 "準備工作節點" 以取得詳細資料。

驗證 **ONTAP** 後端

Trident 提供兩種驗證 ONTAP 後端的模式。

- 認證型：ONTAP 對具備所需權限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 各種版本的最大相容性。
- 憑證型： Trident 也可以使用安裝在後端的憑證與 ONTAP 叢集通訊。在此處、後端定義必須包含用戶端憑證、金鑰及信任的CA憑證（建議使用）的Base64編碼值。

您可以更新現有的後端、以便在認證型和憑證型方法之間移動。不過、一次只支援一種驗證方法。若要切換至不同的驗證方法、您必須從後端組態中移除現有方法。

> 如果您嘗試同時提供*認證與認證*、後端建立將會失敗、並在組態檔中提供多種驗證方法。

啟用認證型驗證

Trident 需要 SVM 範圍 / 叢集範圍管理員的認證、才能與 ONTAP 後端通訊。建議您使用標準的預先定義角色、例如 `admin`或 `vsadmin`。如此可確保與未來 ONTAP 版本的前移相容性、這些版本可能會公開未來 Trident 版本所使用的功能 API 。自訂安全登入角色可建立並搭配 Trident 使用、但不建議使用。

後端定義範例如下所示：

YAML

```yaml
---
version: 1
backendName: ExampleBackend
storageDriverName: ontap-san
managementLIF: 10.0.0.1
svm: svm_nfs
username: vsadmin
password: password
```

JSON

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

請記住、後端定義是唯一以純文字儲存認證的位置。建立後端之後、使用者名稱/密碼會以Base64編碼、並儲存為Kubernetes機密。建立或更新後端是唯一需要具備認證知識的步驟。因此、這是一項純管理員操作、由Kubernetes /儲存管理員執行。

啟用基於憑證的身份驗證

新的和現有的後端可以使用憑證、並與ONTAP 該後端通訊。後端定義需要三個參數。

- 用戶端憑證：用戶端憑證的Base64編碼值。
- 用戶端私密金鑰：關聯私密金鑰的Base64編碼值。
- 信任的CACertifate：受信任CA憑證的Base64編碼值。如果使用信任的CA、則必須提供此參數。如果未使用信任的CA、則可忽略此問題。

典型的工作流程包括下列步驟。

步驟

1. 產生用戶端憑證和金鑰。產生時、請將Common Name（CN）（一般名稱（CN））設定為ONTAP 驗證身分。

```
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ONTAP 至整個叢集。這可能已由儲存管理員處理。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 支援叢集上安裝用戶端憑證和金鑰（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 支援「cert」驗證方法的支援功能。

```
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。以ONTAP Management LIF IP和SVM名稱取代<SfManagement LIF>和<vserver name>。

```
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編碼憑證、金鑰和信任的CA憑證。

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

7. 使用從上一步取得的值建立後端。

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

更新驗證方法或旋轉認證資料

您可以更新現有的後端、以使用不同的驗證方法或旋轉其認證資料。這兩種方法都可行：使用使用者名稱/密碼的後端可更新以使用憑證；使用憑證的後端可更新為使用者名稱/密碼。若要這麼做、您必須移除現有的驗證方法、然後新增驗證方法。然後使用更新的backend.json檔案、其中包含執行「tridentctl後端更新」所需的參數。

```
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 |
+------------+----------------+--------------------------------------
+--------+---------+
```

> 當您旋轉密碼時、儲存管理員必須先更新ONTAP 使用者的密碼（位於BIOS）。接著是後端更新。在循環憑證時、可將多個憑證新增至使用者。然後更新後端以使用新的憑證、之後可從ONTAP 該叢集刪除舊的憑證。

更新後端不會中斷對已建立之磁碟區的存取、也不會影響之後建立的磁碟區連線。成功的後端更新表示 Trident 可以與 ONTAP 後端通訊、並處理未來的 Volume 作業。

為 **Trident** 建立自訂 **ONTAP** 角色

您可以使用最低 Privileges 來建立 ONTAP 叢集角色、這樣就不需要使用 ONTAP 管理員角色來執行 Trident 中的作業。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 叢集角色來執行作業。

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. 使用下列命令建立新角色：

```
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 系統管理員中執行下列步驟：

1. * 建立自訂角色 * ：
   a. 若要在叢集層級建立自訂角色、請選取 * 叢集 > 設定 * 。

      （或）若要在 SVM 層級建立自訂角色、請選取 * 儲存設備 > 儲存 VM > > `required SVM` 設定 > 使用者與角色 * 。

   b. 選取 * 使用者和角色 * 旁的箭頭圖示（ * → * ）。
   c. 在 * 角色 * 下選擇 **+Add** 。
   d. 定義角色的規則、然後按一下 * 儲存 * 。
2. * 將角色對應至 Trident 使用者 * ： + 在「 * 使用者與角色 * 」頁面上執行下列步驟：
   a. 在 * 使用者 * 下選取新增圖示 **+** 。
   b. 選取所需的使用者名稱、然後在 * 角色 * 的下拉式功能表中選取角色。
   c. 按一下「 * 儲存 * 」。

如需詳細資訊、請參閱下列頁面：

- "用於管理 ONTAP 的自訂角色"或"定義自訂角色"
- "與角色和使用者合作"

使用雙向 **CHAP** 驗證連接

Trident 可以使用和 `ontap-san-economy`驅動程式的雙向 `CHAP` 驗證 `iSCSI` 工作階段 `ontap-san`。這需要在後端定義中啟用 `useCHAP`選項。設為 `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。請參閱下列組態範例：

```yaml
---
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」參數是布林選項、只能設定一次。預設值設為假。將其設為true之後、您就無法將其設為假。

除了"useCHAP=true"之外、"chapInitiator Secret（chapInitiator機密）、"chaptarketatorSecret（chaptarketusernameｊ）、"chaptarketusernameｊ（chaptargetuseamuseｊ）和"chapusernameｊ（chamus在建立後端後端之後、可以執行「tridentctl update」來變更機密。

工作原理

儲存管理員會將設定 `useCHAP`為 true 、指示 Trident 在儲存後端上設定 CHAP 。這包括下列項目：

- 在SVM上設定CHAP：
  - 如果 SVM 的預設啟動器安全性類型為無（預設為「無」） * 且 * 磁碟區中沒有預先存在的 LUN 、則 Trident 會將預設安全性類型設為 `CHAP`、並繼續設定 CHAP 啟動器和目標使用者名稱和機密。
  - 如果 SVM 包含 LUN 、 Trident 將不會在 SVM 上啟用 CHAP 。這可確保不限制對 SVM 上已存在的 LUN 的存取。
- 設定CHAP啟動器和目標使用者名稱和機密；這些選項必須在後端組態中指定（如上所示）。

建立後端之後、 Trident 會建立對應的 `tridentbackend` CRD 、並將 CHAP 機密和使用者名稱儲存為 Kubernetes 機密。Trident 在此後端建立的所有 PV 都會透過 CHAP 掛載及附加。

輪換憑證並更新後端

您可以更新「backend.json」檔案中的CHAP參數、以更新CHAP認證。這需要更新CHAP機密、並使用「tridentctl update」命令來反映這些變更。

更新後端的 CHAP 機密時、您必須使用 `tridentctl`來更新後端。請勿使用 ONTAP CLI 或 ONTAP 系統管理員更新儲存叢集上的認證，因為 Trident 將無法取得這些變更。

```
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 |
+----------------+----------------+--------------------------------------
+--------+---------+
```

現有連線不會受到影響；如果 Trident 在 SVM 上更新認證、則這些連線將繼續保持作用中狀態。新的連線使用更新的認證資料、而現有的連線會繼續保持作用中。中斷舊PV的連線並重新連線、將會使用更新的認證資料。

SAN組態選項與範例ONTAP

瞭解如何在 Trident 安裝中建立及使用 ONTAP SAN 驅動程式。本節提供後端組態範例及將後端對應至 StorageClasses 的詳細資料。

"ASA r2 系統"與其他ONTAP系統（ASA、 AFF和FAS）在儲存層的實作上有所不同。這些變化會影響某些參數的使用，如註釋中所述。"詳細了解 ASA r2 系統與其他 ONTAP 系統之間的差異"。

> 只有 `ontap-san`ASA r2 系統支援驅動程式（支援 iSCSI、NVMe/TCP 和 FC 協定）。

在Trident後端設定中，無需指定您的系統是ASA r2。當您選擇 `ontap-san`作為 `storageDriverName`Trident可自動偵測ASA r2 或其他ONTAP系統。如下表所示，某些後端設定參數不適用於ASA r2 系統。

如需後端組態選項、請參閱下表：

參數	說明	預設
「分度」		永遠為1
「torageDriverName」	儲存驅動程式名稱	`ontap-san`或 `ontap-san-economy`
「後端名稱」	自訂名稱或儲存後端	驅動程式名稱 + "_" + dataLIF
《馬納格門達利》	叢集或 SVM 管理 LIF 的 IP 位址。 您可以指定完整網域名稱（FQDN）。 如果使用 IPv6 旗標安裝 Trident 、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]`。 如需無縫 MetroCluster 之間的互通性MetroCluster 範例、請參閱。 ⓘ 如果您使用的是「 vsadmin 」認證，則必須是 SVM 的認 `managementLIF`證；如果使用的是「 `admin` 」認證，則必須是叢集的認證 `managementLIF`。	"10.0.0.1 " ，"[2001:1234:abcd::fefe]
「DataLIF」	傳輸協定LIF的IP位址。如果使用 IPv6 旗標安裝 Trident 、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]`。* 請勿指定 iSCSI 。 **Trident** 使用**"可選擇的LUN對應ONTAP"**來探索建立多重路徑工作階段所需的 **iSCSI** 生命。如果明確定義、就會產生警告 **`dataLIF`**。MetroCluster 省略。 *請參閱MetroCluster 範例。	源自SVM
《虛擬機器》	要使用的儲存虛擬機器 * MetroCluster 請省略。 * 請參閱 MetroCluster 範例。	如果指定SVM "managementLIF"則衍生
《使用CHAP》	使用CHAP驗證iSCSI以供ONTAP 支援不支援的SAN驅動程式使用[布林值]。設為 `true`、讓 Trident 設定並使用雙向 CHAP 做為後端所指定 SVM 的預設驗證。如 "準備使用ONTAP 支援的SAN驅動程式來設定後端" 需詳細資訊、請參閱。*不支援 FCP 或 NVMe/TCP。 *	「假」
《chapInitiator機密》	CHAP啟動器密碼。如果是"useCHAP=true"、則為必要項目	"
《標籤》	套用到磁碟區的任意JSON-格式化標籤集	"

參數	說明	預設
《chapTargetInitiator機密》	CHAP目標啟動器機密。如果是"useCHAP=true"、則為必要項目	"
「chapUsername」	傳入使用者名稱。如果是"useCHAP=true"、則為必要項目	"
《chapTargetUsername》	目標使用者名稱。如果是"useCHAP=true"、則為必要項目	"
「用戶端憑證」	用戶端憑證的Base64編碼值。用於憑證型驗證	"
「clientPrivateKey」	用戶端私密金鑰的Base64編碼值。用於憑證型驗證	"
「可信賴的CACertifcate」	受信任CA憑證的Base64編碼值。選用。用於憑證型驗證。	"
《使用者名稱》	與ONTAP叢集通訊所需的使用者名稱。用於基於憑證的身份驗證。有關 Active Directory 驗證，請參閱 "使用 Active Directory 憑證向後端 SVM 驗證Trident 的身份"。	"
密碼	與ONTAP集群通訊所需的密碼。用於基於憑證的身份驗證。有關 Active Directory 驗證，請參閱 "使用 Active Directory 憑證向後端 SVM 驗證Trident 的身份"。	"
《虛擬機器》	要使用的儲存虛擬機器	如果指定SVM "managementLIF"則衍生
「toragePrefix」	在SVM中配置新磁碟區時所使用的前置碼。稍後無法修改。若要更新此參數、您需要建立新的後端。	`trident`
《Aggregate》	用於資源配置的Aggregate（選用；如果已設定、則必須指派給SVM）。對於 `ontap-nas-flexgroup`驅動程式、此選項會被忽略。如果未指派、 任何可用的集合體都可用於佈建 FlexGroup Volume 。 ⓘ 在 SVM 中更新 Aggregate 時、它會透過輪詢 SVM 而無需重新啟動 Trident 控制器、在 Trident 中自動更新。當您在 Trident 中設定特定的 Aggregate 以配置 Volume 時、如果將 Aggregate 重新命名或移出 SVM 、則在輪詢 SVM Aggregate 時、後端將會移至 Trident 中的失敗狀態。您必須將 Aggregate 變更為 SVM 上的 Aggregate 、或是將其全部移除、才能使後端重新上線。 不要指定 **ASA r2** 系統。	"

參數	說明	預設
「限制Aggregateusage」	如果使用率高於此百分比、則無法進行資源配置。如果您使用 Amazon FSX for NetApp ONTAP 後端、請勿指定 `limitAggregateUsage`。提供的 `fsxadmin`和 `vsadmin`不包含使用 Trident 擷取彙總使用量並加以限制所需的權限。不要指定 **ASA r2** 系統。	""（預設不強制執行）
《限制Volume大小》	如果要求的磁碟區大小高於此值、則資源配置失敗。也會限制其管理 LUN 的最大磁碟區大小。	"" （預設不會強制執行）
《lunsPerFlexvol》	每FlexVol 個LUN的最大LUN數量、範圍必須在[50、200]	`100`
「DebugTraceFlags」	疑難排解時要使用的偵錯旗標。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。	`null`
《useREST》	使用ONTAP REST API 的布林參數。 `useREST`當設定為 `true`，Trident使用ONTAP REST API 與後端通訊；當設定為 `false`，Trident使用 ONTAPI（ZAPI）呼叫與後端進行通訊。此功能需要ONTAP 9.11.1 及更高版本。此外，使用的ONTAP登入角色必須具有訪問 `ontapi`應用。這是透過預定義的 `vsadmin`和 `cluster-admin` 角色。從Trident 24.06 版本和ONTAP 9.15.1 或更高版本開始，`useREST`設定為 `true`預設；改變 `useREST`到 `false`使用 ONTAPI (ZAPI) 呼叫。 `useREST`完全符合 NVMe/TCP 的要求。 NVMe 僅支援ONTAP REST API，不支援 ONTAPI (ZAPI)。 如果指定，則始終設定為 **`true`**適用於**ASA r2** 系統。	`true` 對於 ONTAP 9.15.1 或更高版本，否則 `false`。
`sanType`	用於選擇 `iscsi` iSCSI 、 `nvme` NVMe / TCP 或 `fcp` SCSI over Fibre Channel （ FC ）。	`iscsi` 如果空白

參數	說明	預設
formatOptions	用於 formatOptions`指定命令的命令列引數、每當格式化磁碟區時都會套用這些引數 `mkfs。這可讓您根據偏好設定來格式化 Volume 。請務必指定與 mkfs 命令選項類似的格式選項、但不包括裝置路徑。範例：「 -E nobard 」 *支援 `ontap-san`和 `ontap-san-economy`帶有 iSCSI 協定的驅動程式。 **此外，在使用 iSCSI 和 NVMe/TCP 協定時，支援 ASA r2 系統。 *	
limitVolumePoolSize	在 ONTAP SAN 經濟型後端中使用 LUN 時、可要求的最大 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 。只有預先存在的 FlexVols 可用於佈建新的 PV 。	

使用 **formatOptions** 的建議

Trident建議採用以下選項來加速格式化流程：

- **-E nodiscard (ext3, ext4):** 不要嘗試在 mkfs 時丟棄區塊（最初丟棄區塊對固態設備和稀疏/精簡配置儲存很有用）。這將取代已棄用的選項“-K”，並且適用於 ext3、ext4 檔案系統。
- **-K (xfs):** 不要在執行 mkfs 時嘗試丟棄區塊。此選項適用於 xfs 檔案系統。

使用 **Active Directory** 憑證向後端 **SVM** 驗證**Trident** 的身份

您可以設定Trident以使用 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。對於使用 AD 帳戶進行叢集管理，您必須建立網域隧道。參考 "在ONTAP中設定 Active Directory 網域控制站存取" 了解詳情。

步驟

1. 為後端 SVM 配置網域名稱系統 (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後端設定檔中，設定 username 和 password 參數分別為 AD 使用者或群組名稱和密碼。

用於資源配置磁碟區的後端組態選項

您可以使用中的這些選項來控制預設資源配置 `defaults` 組態區段。如需範例、請參閱下列組態範例。

參數	說明	預設
"paceAllocate（配置）"	LUN的空間分配	"true" 如果指定，則設定為 `**true**`適用於 **ASA r2** 系統。
《保護區》	空間保留模式；「無」（精簡）或「 Volume （大量）」（粗）。設定為 `**none**`適用於 **ASA r2** 系統。	" 無 "
「快照原則」	要使用的 Snapshot 原則。設定為 `**none**`適用於 **ASA r2** 系統。	" 無 "
「qosPolicy」	要指派給所建立磁碟區的QoS原則群組。選擇每個儲存集區/後端的其中一個qosPolicy或adaptiveQosPolicy。搭配 Trident 使用 QoS 原則群組需要 ONTAP 9 8 或更新版本。您應該使用非共用的 QoS 原則群組、並確保個別將原則群組套用至每個成員。共享 QoS 原則群組會強制執行所有工作負載總處理量的上限。	"
《adaptiveQosPolicy》	要指派給所建立磁碟區的調適性QoS原則群組。選擇每個儲存集區/後端的其中一個qosPolicy或adaptiveQosPolicy	"
「快照保留區」	保留給快照的磁碟區百分比。不要為 **ASA r2** 系統指定。	「 0 」如果 `snapshotPolicy` 為「無」、否則為「」
「PlitOnClone」	建立複本時、從其父複本分割複本	"假"
加密	在新磁碟區上啟用 NetApp Volume Encryption （ NVE ）；預設為 `false`。必須在叢集上授權並啟用NVE、才能使用此選項。如果在後端啟用 NAE 、則 Trident 中配置的任何 Volume 都將啟用 NAE 。如需更多資訊、請參閱"Trident 如何與 NVE 和 NAE 搭配運作"：。	“false” 如果指定，則設定為 `**true**`適用於 **ASA r2** 系統。
`luksEncryption`	啟用LUKS加密。請參閱 "使用Linux統一金鑰設定（LUKS）"。	"" 設定為 `**false**`適用於 **ASA r2** 系統。
「分層政策」	分層策略使用「無」 不要為 **ASA r2** 系統指定。	
`nameTemplate`	建立自訂磁碟區名稱的範本。	"

Volume資源配置範例

以下是定義預設值的範例：

```yaml
---
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'
```

> 對於使用驅動程式建立的所有磁碟區 `ontap-san`、 Trident 會為 FlexVol 額外增加 10% 的容量、以容納 LUN 中繼資料。LUN的配置大小與使用者在PVc中要求的大小完全相同。Trident 將 10% 新增至 FlexVol （在 ONTAP 中顯示為可用大小）。使用者現在可以取得所要求的可用容量。此變更也可防止LUN成為唯讀、除非可用空間已充分利用。這不適用於ONTAP-san經濟型。

對於定義的後端 `snapshotReserve`， Trident 將按以下方式計算卷的大小：

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

1.1 是Trident為容納 LUN 元資料而額外添加到FlexVol 的10%。對於 `snapshotReserve` = 5%，PVC 請求 = 5 GiB，則總磁碟區大小為 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：

```
Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- -----
          _pvc_89f1c156_3801_4de4_9f9d_034d54c395f4
                                  online     RW         10GB     5.00GB    0%
          _pvc_e42ec6fe_3baa_4af6_996d_134adbbb8e6d
                                  online     RW       5.79GB     5.50GB    0%
          _pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba
                                  online     RW          1GB    511.8MB    0%
3 entries were displayed.
```

目前、只有調整大小、才能將新計算用於現有的Volume。

最低組態範例

下列範例顯示基本組態、讓大部分參數保留預設值。這是定義後端最簡單的方法。

> 如果您在 NetApp ONTAP 上搭配 Trident 使用 Amazon FSX ， NetApp 建議您指定生命體的 DNS 名稱，而非 IP 位址。

ONTAP SAN 範例

這是使用的基本組態 `ontap-san` 驅動程式：

```
---
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、避免在切換和切換期間手動更新後端定義 "SVM 複寫與還原"。

若要無縫切換和切換，請使用並省略 `svm`參數來指定 `SVM` `managementLIF` 。例如：

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

ONTAP SAN 經濟效益範例

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

憑證型驗證範例

在此基本組態範例中 `clientCertificate`、 `clientPrivateKey`和 `trustedCACertificate` （選用、如果使用信任的CA）會填入 `backend.json` 並分別取得用戶端憑證、私密金鑰及信任CA憑證的基礎64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

這些範例使用建立後端 `useCHAP` 設定為 `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 經濟 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

您必須在 ONTAP 後端上設定 NVMe 的 SVM 。這是適用於 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

您必須在 ONTAP 後端設定具有 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
```

名稱範本的後端組態範例

```
---
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 經濟型驅動程式範例

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

虛擬集區的後端範例

在這些後端定義檔案範例中、會針對所有儲存池設定特定的預設值、例如 `spaceReserve` 無、`spaceAllocation` 假、和 `encryption` 錯。虛擬資源池是在儲存區段中定義的。

Trident 會在「意見」欄位中設定資源配置標籤。在 FlexVol volume Trident 上設定的註解會將虛擬集區上的所有標籤複製到資源配置時的儲存磁碟區。為了方便起見、儲存管理員可以針對每個虛擬資源池定義標籤、並依標籤將磁碟區分組。

在這些範例中、有些儲存池是自行設定的 `spaceReserve`、 `spaceAllocation`和 `encryption` 值、而某些資源池會覆寫預設值。

ONTAP SAN 範例

```yaml
---
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 經濟效益範例

```yaml
---
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**StorageClass**

下列 StorageClass 定義請參閱 [虛擬集區的後端範例]。使用 `parameters.selector` 欄位中、每個 StorageClass 都會呼叫哪些虛擬集區可用於主控磁碟區。磁碟區將會在所選的虛擬資源池中定義各個層面。

- 。`protection-gold` StorageClass 會對應至中的第一個虛擬集區 `ontap-san` 後端：這是唯一提供金級保護的集區。

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

- 。 `protection-not-gold` StorageClass 會對應至中的第二個和第三個虛擬集區 `ontap-san` 後端：這是唯一提供金級以外保護層級的集區。

```yaml
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 會對應至中的第三個虛擬集區 `ontap-san-economy` 後端：這是唯一為 mysqldb 類型應用程式提供儲存池組態的集區。

```yaml
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 會對應至中的第二個虛擬集區 `ontap-san` 後端：這是唯一提供銀級保護和 20000 個信用點數的資源池。

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

- 。 `creditpoints-5k` StorageClass 會對應至中的第三個虛擬集區 `ontap-san` 中的後端和第四個虛擬集區 `ontap-san-economy` 後端：這是唯一擁有 5000 個信用點數的集區方案。

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

- 。 `my-test-app-sc` StorageClass 會對應至 `testAPP` 中的虛擬集區 `ontap-san` 驅動程式搭配 `sanType: nvme`。這是唯一的集區服務項目 `testApp`。

```
---
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 會決定要選取哪個虛擬集區、並確保符合儲存需求。

ASNAS驅動程式ONTAP

ONTAP NAS 驅動程式概述

深入瞭解如何使用ONTAP 功能性和功能性NAS驅動程式來設定功能性的後端。ONTAP Cloud Volumes ONTAP

ONTAP NAS 驅動程式詳細資料

Trident 提供下列 NAS 儲存驅動程式、可與 ONTAP 叢集進行通訊。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	**Volume**模式	支援的存取模式	支援的檔案系統
「ONTAP-NAS」	NFS 中小企業	檔案系統	Rwo 、 ROX 、 rwx 、 RWOP	" 、 `nfs`、 `smb`
《ONTAP-NANAS經濟》	NFS 中小企業	檔案系統	Rwo 、 ROX 、 rwx 、 RWOP	" 、 `nfs`、 `smb`
「ONTAP-NAA-flexgroup」	NFS 中小企業	檔案系統	Rwo 、 ROX 、 rwx 、 RWOP	" 、 `nfs`、 `smb`

- 使用 `ontap-san-economy` 只有持續磁碟區使用量計數預期會高於 "支援的 ONTAP Volume 限制"。
- 使用 `ontap-nas-economy` 只有持續磁碟區使用量計數預期會高於 "支援的 ONTAP Volume 限制" 和 `ontap-san-economy` 無法使用驅動程式。
- 請勿使用 `ontap-nas-economy` 如果您預期需要資料保護、災難恢復或行動性、
- NetApp 不建議在所有 ONTAP 驅動程式中使用 FlexVol 自動擴充， ONTAP SAN 除外。作為因應措施， Trident 支援使用快照保留，並據此擴充 FlexVol 磁碟區。

使用者權限

Trident 預期會以 ONTAP 或 SVM 管理員的身分執行、通常使用叢集使用者或 `vsadmin` SVM 使用者、或是使用 `admin`具有相同角色的不同名稱的使用者。

對於用於 NetApp ONTAP 部署的 Amazon FSX 、 Trident 預期會以 ONTAP 或 SVM 管理員的身分、使用叢集使用者或 `vsadmin` SVM 使用者、或是具有相同角色的不同名稱的使用者來執行 `fsxadmin`。 `fsxadmin`使用者只能有限地取代叢集管理使用者。

如果您使用此 `limitAggregateUsage`參數、則需要叢集管理權限。將 Amazon FSX for NetApp ONTAP 搭配 Trident 使用時、此 `limitAggregateUsage`參數將無法與和 `fsxadmin`使用者帳戶搭配 `vsadmin`使用。如果您指定此參數、組態作業將會失敗。

雖然可以在 ONTAP 中建立更具限制性的角色、讓 Trident 驅動程式可以使用、但我們不建議這樣做。Trident的大多數新版本都會呼叫額外的API、而這些API必須納入考量、使升級變得困難且容易出錯。

準備使用**ONTAP** 不含**NAS**的驅動程式來設定後端

瞭解使用 ONTAP NAS 驅動程式設定 ONTAP 後端的需求、驗證選項和匯出原則。

從 25.10 版本開始， NetApp Trident支持"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。

只有 `ontap-nas`AFX 系統支援 NFS 協定驅動程式；不支援 SMB 協定。

在Trident後端設定中，您無需指定您的系統是 AFX。當您選擇 `ontap-nas`作為 `storageDriverName`Trident可自動偵測 AFX 系統。

需求

- 對於所有 ONTAP 後端，Trident 要求至少將一個聚合分配給 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的儲存類別。例如、您可以設定使用的Gold類別 `ontap-nas` 驅動程式和銅級、使用 `ontap-nas-economy` 一、
- 您所有的Kubernetes工作節點都必須安裝適當的NFS工具。請參閱 "請按這裡" 以取得更多詳細資料。
- Trident 僅支援掛載至 Windows 節點上執行的 Pod 的 SMB 磁碟區。如 準備配置SMB磁碟區 需詳細資訊、請參閱。

驗證 **ONTAP** 後端

Trident 提供兩種驗證 ONTAP 後端的模式。

- 認證型：此模式需要對 ONTAP 後端擁有足夠的權限。建議您使用與預先定義的安全登入角色相關聯的帳戶、例如 `admin` 或 `vsadmin` 以確保與ONTAP 更新版本的最大相容性。
- 憑證型：此模式需要在後端安裝憑證、 Trident 才能與 ONTAP 叢集通訊。在此處、後端定義必須包含用戶端憑證、金鑰及信任的CA憑證（建議使用）的Base64編碼值。

您可以更新現有的後端、以便在認證型和憑證型方法之間移動。不過、一次只支援一種驗證方法。若要切換至不同的驗證方法、您必須從後端組態中移除現有方法。

> 如果您嘗試同時提供*認證與認證*、後端建立將會失敗、並在組態檔中提供多種驗證方法。

啟用認證型驗證

Trident 需要 SVM 範圍 / 叢集範圍管理員的認證、才能與 ONTAP 後端通訊。建議您使用標準的預先定義角色、例如 `admin``或 `vsadmin`。如此可確保與未來 ONTAP 版本的前移相容性、這些版本可能會公開未來 Trident 版本所使用的功能 API 。自訂安全登入角色可建立並搭配 Trident 使用、但不建議使用。

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"
    }
}
```

請記住、後端定義是唯一以純文字儲存認證的位置。建立後端之後、使用者名稱/密碼會以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、這是一項純管理員操作、由Kubernetes /儲存管理員執行。

啟用憑證型驗證

新的和現有的後端可以使用憑證、並與ONTAP 該後端通訊。後端定義需要三個參數。

- 用戶端憑證：用戶端憑證的Base64編碼值。
- 用戶端私密金鑰：關聯私密金鑰的Base64編碼值。
- 信任的CACertifate：受信任CA憑證的Base64編碼值。如果使用信任的CA、則必須提供此參數。如果未使用信任的CA、則可忽略此問題。

典型的工作流程包括下列步驟。

步驟

1. 產生用戶端憑證和金鑰。產生時、請將Common Name（CN）（一般名稱（CN））設定為ONTAP 驗證身分。

```
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ONTAP 至整個叢集。這可能已由儲存管理員處理。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 支援叢集上安裝用戶端憑證和金鑰（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 支援「cert」驗證方法的支援功能。

```
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。以ONTAP Management LIF IP和SVM名稱取代<SfManagement LIF>和<vserver name>。您必須確保LIF的服務原則設定為「預設資料管理」。

```
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編碼憑證、金鑰和信任的CA憑證。

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

7. 使用從上一步取得的值建立後端。

```
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...Vaaallllluuuueeee",
"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 |
+------------+----------------+--------------------------------------
+--------+---------+
```

更新驗證方法或旋轉認證資料

您可以更新現有的後端、以使用不同的驗證方法或旋轉其認證資料。這兩種方法都可行：使用使用者名稱/密碼的後端可更新以使用憑證；使用憑證的後端可更新為使用者名稱/密碼。若要這麼做、您必須移除現有的驗證方法、然後新增驗證方法。然後使用更新的backend.json檔案、其中包含要執行的必要參數 `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 |
+------------+----------------+--------------------------------------
+--------+---------+
```

> 當您旋轉密碼時、儲存管理員必須先更新ONTAP 使用者的密碼（位於BIOS）。接著是後端更新。在循環憑證時、可將多個憑證新增至使用者。然後更新後端以使用新的憑證、之後可從ONTAP 該叢集刪除舊的憑證。

更新後端不會中斷對已建立之磁碟區的存取、也不會影響之後建立的磁碟區連線。成功的後端更新表示 Trident 可以與 ONTAP 後端通訊、並處理未來的 Volume 作業。

為 **Trident** 建立自訂 **ONTAP** 角色

您可以使用最低 Privileges 來建立 ONTAP 叢集角色、這樣就不需要使用 ONTAP 管理員角色來執行 Trident 中的作業。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 叢集角色來執行作業。

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. 使用下列命令建立新角色：

```
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 系統管理員中執行下列步驟：

1. * 建立自訂角色 * ：
   a. 若要在叢集層級建立自訂角色、請選取 * 叢集 > 設定 * 。

      （或）若要在 SVM 層級建立自訂角色、請選取 * 儲存設備 > 儲存 VM > > `required SVM` 設定 > 使用者與角色 * 。

   b. 選取 * 使用者和角色 * 旁的箭頭圖示（ * → * ）。
   c. 在 * 角色 * 下選擇 **+Add** 。
   d. 定義角色的規則、然後按一下 * 儲存 * 。
2. * 將角色對應至 Trident 使用者 * ： + 在「 * 使用者與角色 * 」頁面上執行下列步驟：
   a. 在 * 使用者 * 下選取新增圖示 **+** 。
   b. 選取所需的使用者名稱、然後在 * 角色 * 的下拉式功能表中選取角色。
   c. 按一下「 * 儲存 * 」。

如需詳細資訊、請參閱下列頁面：

- "用於管理 ONTAP 的自訂角色"或"定義自訂角色"
- "與角色和使用者合作"

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 在使用匯出原則時提供兩個選項：

- Trident 可以動態管理匯出原則本身；在此作業模式中、儲存管理員會指定代表可接受 IP 位址的 CIDR 區塊清單。Trident 會在發佈時自動將屬於這些範圍的適用節點 IP 新增至匯出原則。或者、如果未指定 CIDR 、則在要發佈的磁碟區所在節點上找到的所有全域範圍單點傳播 IP 都會新增至匯出原則。
- 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 會使用預設匯出原則。

動態管理匯出原則

Trident 提供動態管理 ONTAP 後端匯出原則的功能。這可讓儲存管理員為工作節點IP指定允許的位址空間、而非手動定義明確的規則。它可大幅簡化匯出原則管理；修改匯出原則不再需要在儲存叢集上進行手動介入。此外、這有助於將儲存叢集的存取限制在裝載磁碟區且指定範圍內有 IP 的工作節點、以支援精細且自動化的管理。

> ⓘ 使用動態匯出原則時、請勿使用網路位址轉譯（ NAT ）。使用 NAT 時、儲存控制器會看到前端 NAT 位址、而非實際 IP 主機位址、因此在匯出規則中找不到相符項目時、就會拒絕存取。

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、您必須確保SVM中的根連接點具有先前建立的匯出原則、並具有允許節點CIDR區塊（例如預設匯出原則）的匯出規則。請務必遵循 NetApp 建議的最佳實務做法、將 SVM 專用於 Trident 。

以下是使用上述範例說明此功能的運作方式：

- `autoExportPolicy`**設定為** `true`。這表示 Trident 會為使用此後端為 SVM 佈建的每個 Volume 建立匯出原則 `svm1`、並使用位址區塊來處理規則的新增和刪除 `autoexportCIDRs`。在磁碟區附加至節點之前、該磁碟區會使用沒有規則的空匯出原則、以防止不必要的存取該磁碟區。當磁碟區發佈至節點 Trident 時、會建立一個匯出原則、其名稱與包含指定 CIDR 區塊內節點 IP 的基礎 qtree 相同。這些 IP 也會新增至父 FlexVol volume 所使用的匯出原則
  - 例如：
    - 後端 UUID 403b5326-8482-40der-96d0-d83fb3f4daec
    - `autoExportPolicy`**設定為** `true`

    - 儲存字首 `trident`
    - PVC UUID a79bcf5f-7b6d-4a40-9876-e2551f159c1c
    - qtree 名稱為 Trident _PVC_a79bcf5f_7b6d_4a40_9876_e2551f159c1c FlexVol 、會為命名的 qtree 建立匯出原則、為命名的 qtree 建立匯 `trident-403b5326-8482-40db96d0-d83fb3f4daec` 出原則、 `trident_pvc_a79bcf5f_7b6d_4a40_9876_e2551f159c1c`以及在 `SVM` 上命名的空白匯出原則 `trident_empty`。FlexVol 匯出原則的規則將是 qtree 匯出原則所包含的任何規則的超集。未附加的任何磁碟區都會重複使用空的匯出原則。
- `autoExportCIDRs`包含位址區塊清單。此欄位為選用欄位、預設為「0.00.0.0/0」、「：/0」。如果未定義、 Trident 會新增所有在工作節點上找到的全域範圍單點傳播位址、並提供出版物。

在此範例中 `192.168.0.0/24`、會提供位址空間。這表示位於此位址範圍內的 Kubernetes 節點 IP 與出版物將會新增至 Trident 所建立的匯出原則。當 Trident 登錄其執行的節點時，它會擷取節點的 IP 位址，並對照中提供的位址區塊進行檢查 `autoExportCIDRs`。在發佈時，在篩選 IP 之後， Trident 會為其所發佈節點的用戶端 IP 建立匯出原則規則。

您可以在建立後端後、更新「AutoExpportPolicy」和「AutoExpportCTR」。您可以為自動管理或刪除現有CIDR的後端附加新的CIDR。刪除CIDR時請務必謹慎、以確保不會中斷現有的連線。您也可以選擇停用後端的「autodportPolicy」、然後回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 建立或更新後端之後、您可以使用或對應的 `tridentbackend` CRD 來檢查後端 `tridentctl`：

```
./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
```

移除節點時、 Trident 會檢查所有匯出原則、以移除對應於節點的存取規則。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、使用更新後端 `tridentctl update backend`可確保 Trident 自動管理匯出原則。這會在需要

時建立兩個以後端 UUID 和 qtree 名稱命名的新匯出原則。後端上的磁碟區會在新建立的匯出原則卸載並重新掛載之後、使用這些原則。

> 刪除具有自動管理匯出原則的後端、將會刪除動態建立的匯出原則。如果重新建立後端、則會將其視為新的後端、並導致建立新的匯出原則。

如果即時節點的 IP 位址已更新、您必須在節點上重新啟動 Trident Pod 。然後 Trident 會更新匯出原則、以反映其所管理的 IP 變更。

準備配置SMB磁碟區

只需稍加準備、您就可以使用來配置 SMB 磁碟區 `ontap-nas` 驅動程式：

> 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 。若未設定上述任一種通訊協定、將導致 SMB 磁碟區建立失敗。

> `autoExportPolicy`不支援 SMB Volume 。

開始之前

在配置 SMB 磁碟區之前、您必須具備下列項目。

- Kubernetes叢集具備Linux控制器節點、以及至少一個執行Windows Server 2022的Windows工作節點。Trident 僅支援掛載至 Windows 節點上執行的 Pod 的 SMB 磁碟區。
- 至少有一個 Trident 機密包含您的 Active Directory 認證。產生機密 `smbcreds`：

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

- 設定為Windows服務的SCSI Proxy。若要設定 `csi-proxy`、請參閱 "GitHub：csi Proxy" 或 "GitHub：適用於Windows的SCSI Proxy" 適用於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、或 Trident 可以為您建立 SMB 共用。

   > Amazon FSX for ONTAP 需要 SMB 共享。

   您可以使用兩種方式之一來建立SMB管理共用區 "Microsoft管理主控台" 共享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、請建立共用的目錄路徑結構。

      。`vserver cifs share create` 命令會在共用建立期間檢查-path選項中指定的路徑。如果指定的路徑不存在、則命令會失敗。

   b. 建立與指定SVM相關的SMB共用區：

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

c. 確認共用區已建立：

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

請參閱 "建立SMB共用區" 以取得完整詳細資料。

2. 建立後端時、您必須設定下列項目以指定SMB Volume。如需ONTAP 所有的FSXfor Sendbackend組態選項、請參閱 "FSX提供ONTAP 各種組態選項和範例"。

參數	說明	範例
smbShare	您可以指定下列其中一項：使用 Microsoft 管理主控台或 ONTAP CLI 建立的 SMB 共用名稱；允許 Trident 建立 SMB 共用的名稱；或將參數保留空白以防止共用磁碟區。對於內部部署 ONTAP 、此參數為選用項目。Amazon FSX 需要此參數才能支援 ONTAP 後端、且不可為空白。	smb-share
nasType	*必須設定為 smb.*如果為null、則預設為 nfs。	smb
《生態樣式》	新磁碟區的安全樣式。必須設定為 **ntfs** 或 **mixed** 適用於**SMB**磁碟區。	ntfs 或 mixed 適用於SMB磁碟區
「unixPermissions」	新磁碟區的模式。SMB磁碟區*必須保留為空白。*	"

啟用安全 **SMB**

從 25.06 版本開始，NetApp Trident 支援使用以下方式建立的 SMB 磁碟區的安全性配置 `ontap-nas`和 `ontap-nas-economy`後端。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。

值得記住的重點

- 輸入 `ontap-nas-economy`不支援卷。
- 僅支援唯讀克隆 `ontap-nas-economy`卷。
- 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。
- 克隆 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。
- 如果您在後端、儲存類別和 PVC 中為同一個 AD 使用者提供不同的權限，則權限優先權為：PVC、儲存類別、後端。
- 安全 SMB 支持 `ontap-nas`託管磁碟區匯入，不適用於非託管磁碟區匯入。

步驟

1. 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. 在儲存類別中加入註解。

   添加 `trident.netapp.io/smbShareAdUser`註解到儲存類，以啟用安全 SMB 而不會失敗。為註釋指定的使用者值 `trident.netapp.io/smbShareAdUser`應該與 `smbcreds`秘密。您可以選擇以下其中之一 `smbShareAdUserPermission ： full_control ， change ， 或者 read 。預設權限是 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
```

列舉**NAS**組態選項與範例**ONTAP**

瞭解如何在 Trident 安裝中建立及使用 ONTAP NAS 驅動程式。本節提供後端組態範例及將後端對應至 StorageClasses 的詳細資料。

從 25.10 版本開始， NetApp Trident支持"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。

> 只有 `ontap-nas`NetApp AFX 系統支援 NFS 協定驅動程式；不支援 SMB 協定。

在Trident後端設定中，無需指定您的系統是NetApp AFX 儲存系統。當您選擇 `ontap-nas`作為 `storageDriverName`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

如需後端組態選項、請參閱下表：

參數	說明	預設
「分度」		永遠為1
「torageDriverName」	儲存驅動程式名稱 ⓘ 僅適用於NetApp AFX 系統 `ontap-nas`已支援。	`ontap-nas`、 `ontap-nas-economy`或 `ontap-nas-flexgroup`

參數	說明	預設
「後端名稱」	自訂名稱或儲存後端	驅動程式名稱 + "_" + dataLIF
《馬納格門達利》	叢集或 SVM 管理 LIF 的 IP 位址可以指定完整網域名稱（ FQDN ）。如果使用 IPv6 旗標安裝 Trident 、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]`。如需無縫 MetroCluster 之間的互通性MetroCluster 範例、請參閱。	"10.0.0.1 "，"[2001:1234:abcd::fefe]
「DataLIF」	傳輸協定LIF的IP位址。NetApp 建議指定 `dataLIF`。如果未提供， Trident 會從 SVM 擷取 dataLIFs 。您可以指定完整網域名稱（ FQDN ），以用於 NFS 裝載作業，讓您建立循環 DNS ，以便在多個 dataLIFs 之間進行負載平衡。可在初始設定之後變更。請參閱 。如果使用 IPv6 旗標安裝 Trident 、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]`。* MetroCluster 省略。 *請參閱MetroCluster 範例。	指定位址或從SVM衍生（若未指定）（不建議使用）
《虛擬機器》	要使用的儲存虛擬機器 * MetroCluster 請省略。 * 請參閱 MetroCluster 範例。	如果指定SVM "managementLIF"則衍生
「AutoExpportPolicy」	啟用自動匯出原則建立及更新[布 林值]。使用 `autoExportPolicy`和 `autoExportCIDRs`選項、 Trident 可以自動管理匯出原則。	錯
《AutoExpportCIDR》（自動匯出CTR）	將 Kubernetes 節點 IP 篩選在啟用時的 CIDR 清單 `autoExportPolicy`。使用 `autoExportPolicy`和 `autoExportCIDRs`選項、 Trident 可以自動管理匯出原則。	["0.0.0/0" 、 ":/0"]`
《標籤》	套用到磁碟區的任意JSON-格式化標籤集	"
「用戶端憑證」	用戶端憑證的Base64編碼值。用於憑證型驗證	"
「clientPrivateKey」	用戶端私密金鑰的Base64編碼值。用於憑證型驗證	"
「可信賴的CACertifcate」	受信任CA憑證的Base64編碼值。選用。用於憑證型驗證	"
《使用者名稱》	用於連接到叢集/SVM 的使用者名稱。用於基於憑證的身份驗證。有關 Active Directory 驗證，請參閱 "使用 Active Directory 憑證向後端 SVM 驗證Trident 的身份"。	
密碼	連接到叢集/SVM 的密碼。用於基於憑證的身份驗證。有關 Active Directory 驗證，請參閱 "使用 Active Directory 憑證向後端 SVM 驗證Trident 的身份"。	

參數	說明	預設
「toragePrefix」	在SVM中配置新磁碟區時所使用的前置碼。設定後無法更新 ⓘ 使用 ONTAP NAS 經濟型和 24 個以上字元的 storagePrefix 時， qtree 將不會內嵌儲存前置字元，不過它會位於磁碟區名稱中。	" Trident "
《Aggregate》	用於資源配置的Aggregate（選用；如果已設定、則必須指派給SVM）。對於 `ontap-nas-flexgroup`驅動程式、此選項會被忽略。如果未指派、 任何可用的集合體都可用於佈建 FlexGroup Volume 。 ⓘ 在 SVM 中更新 Aggregate 時、它會透過輪詢 SVM 而無需重新啟動 Trident 控制器、在 Trident 中自動更新。當您在 Trident 中設定特定的 Aggregate 以配置 Volume 時、如果將 Aggregate 重新命名或移出 SVM 、則在輪詢 SVM Aggregate 時、後端將會移至 Trident 中的失敗狀態。您必須將 Aggregate 變更為 SVM 上的 Aggregate 、或是將其全部移除、才能使後端重新上線。 *請勿指定用於 AFX 儲存系統。 *	"
「限制Aggregateusage」	如果使用率超過此百分比，則配置失敗。不適用於**Amazon FSx for ONTAP**。*請勿指定用於 AFX 儲存系統。 *	""（預設不強制執行）
FlexgroupAggregateList	用於資源配置的集合體清單（選用；如果已設定、則必須指派給 SVM ）。指派給 SVM 的所有集合體都會用於佈建 FlexGroup Volume 。支援 * ONTAP NAS FlexGroup * 儲存驅動程式。 ⓘ 在 SVM 中更新 Aggregate 清單時、會透過輪詢 SVM 而無需重新啟動 Trident 控制器、自動在 Trident 中更新清單。當您在 Trident 中設定特定的 Aggregate 清單來配置 Volume 時、如果將 Aggregate 清單重新命名或移出 SVM 、則在輪詢 SVM Aggregate 時、後端將會移至 Trident 中的失敗狀態。您必須將 Aggregate 清單變更為 SVM 上的集合清單、或是將其全部移除以使後端重新上線。	"
《限制Volume大小》	如果請求的磁碟區大小大於此值，則配置失敗。	"" （預設不會強制執行）

參數	說明	預設
「DebugTraceFlags」	疑難排解時要使用的偵錯旗標。例如、 { "api" ： false 、 "method" ： true} 請勿使用 `debugTraceFlags` 除非您正在疑難排解並需要詳細的記錄傾印。	null
`nasType`	配置 NFS 或 SMB 磁碟區的建立。選項有 `nfs`，`smb`或空值。設定為 null 則預設使用 NFS 磁碟區。如果指定，則始終設定為 `**nfs**`適用於 **AFX** 儲存系統。	`nfs`
「nfsMountOptions」	以逗號分隔的NFS掛載選項清單。Kubernetes-Persistent Volume 的掛載選項通常是在儲存類別中指定、但如果儲存類別中未指定掛載選項、則 Trident 會回復為使用儲存後端組態檔案中指定的掛載選項。如果儲存類別或組態檔案中未指定任何掛載選項、 Trident 將不會在關聯的持續磁碟區上設定任何掛載選項。	"
"qtreesPerFlexvol"	每FlexVol 個邊的最大qtree數、必須在範圍內[50、300]	"200"
`smbShare`	您可以指定下列其中一項：使用 Microsoft 管理主控台或 ONTAP CLI 建立的 SMB 共用名稱；允許 Trident 建立 SMB 共用的名稱；或將參數保留空白以防止共用磁碟區。對於內部部署 ONTAP 、此參數為選用項目。Amazon FSX 需要此參數才能支援 ONTAP 後端、且不可為空白。	`smb-share`
《useREST》	使用ONTAP REST API 的布林參數。 `useREST`設定為 `true`Trident使用ONTAP REST API 與後端通訊；當設定為 `false`Trident使用 ONTAPI (ZAPI) 呼叫與後端通訊。此功能需要ONTAP 9.11.1 及更高版本。此外，所使用的ONTAP登入角色必須具有存取權限。 `ontapi`應用。預定義項滿足了這一點。 `vsadmin`和 `cluster-admin`角色。從Trident 24.06 版本和ONTAP 9.15.1 或更高版本開始， `useREST`設定為 `true`預設；更改 `useREST`到 `false`使用 ONTAPI (ZAPI) 呼叫。如果指定，則始終設定為 `**true**`適用於 **AFX** 儲存系統。	`true` 對於 ONTAP 9.15.1 或更高版本，否則 `false`。
`limitVolumePoolSize`	在 ONTAP NAS 經濟型後端使用 qtree 時、可要求的 FlexVol 大小上限。	""（預設不強制執行）
`denyNewVolumePools`	限制 `ontap-nas-economy`後端建立新的 FlexVol 磁碟區以包含其 qtree 。只有預先存在的 FlexVols 可用於佈建新的 PV 。	
`adAdminUser`	具有 SMB 共用完全存取權限的 Active Directory 管理員使用者或使用者群組。使用此參數可為 SMB 共用提供具有完全控制權的管理員權限。	

用於資源配置磁碟區的後端組態選項

您可以使用中的這些選項來控制預設資源配置 `defaults` 組態區段。如需範例、請參閱下列組態範例。

參數	說明	預設
"paceAllocate（配置）"	qtree 的空間分配	" 對 "
《保護區》	空間保留模式；「無」（精簡）或「 Volume 」（粗）	" 無 "
「快照原則」	要使用的Snapshot原則	" 無 "
「qosPolicy」	要指派給所建立磁碟區的QoS原則群組。選擇每個儲存集區/後端的其中一個qosPolicy或adaptiveQosPolicy	"
《adaptiveQosPolicy》	要指派給所建立磁碟區的調適性QoS原則群組。選擇每個儲存集區/後端的其中一個qosPolicy或adaptiveQosPolicy。不受ONTAP-NAS-經濟支援。	"
「快照保留區」	保留給快照的磁碟區百分比	「 0 」如果 `snapshotPolicy` 為「無」、否則為「」
「PlitOnClone」	建立複本時、從其父複本分割複本	"假"
加密	在新磁碟區上啟用 NetApp Volume Encryption （ NVE ）；預設為 `false`。必須在叢集上授權並啟用NVE、才能使用此選項。如果在後端啟用 NAE 、則 Trident 中配置的任何 Volume 都將啟用 NAE 。如需更多資訊、請參閱"Trident 如何與 NVE 和 NAE 搭配運作"：。	"假"
「分層政策」	分層原則以使用「無」	
「unixPermissions」	新磁碟區的模式	"777" 表示 NFS 磁碟區； SMB 磁碟區為空的（不適用）
「napshotDir	控制對的存取 `.snapshot` 目錄	針對 NFSv3 的 NFSv4 "false" 為 "true"
「匯出政策」	要使用的匯出原則	"預設"
《生態樣式》	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預設為 `unix`。SMB預設為 `ntfs`。
`nameTemplate`	建立自訂磁碟區名稱的範本。	"

搭配 Trident 使用 QoS 原則群組需要 ONTAP 9 8 或更新版本。您應該使用非共用的 QoS 原則群組、並確保個別將原則群組套用至每個成員。共享 QoS 原則群組會強制執行所有工作負載總處理量的上限。

Volume資源配置範例

以下是定義預設值的範例：

```yaml
---
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"
```

為了 `ontap-nas`和 `ontap-nas-flexgroups`Trident現在使用新的計算方法，以確保FlexVol 的尺寸與 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具有更多空間的原始FlexVol 。此計算可確保使用者在 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，當使用者要求 PVC（例如 5 GiB）時，如果快照預留百分比為 50%，則只能獲得 2.5 GiB 的可寫入空間。這是因為使用者要求的是整個磁碟區。 `snapshotReserve`是其中的百分比。在Trident 21.07 中，使用者要求的是可寫入空間，而Trident定義了該空間。`snapshotReserve`將數值表示為佔總體積的百分比。這不適用於 `ontap-nas-economy。請參閱以下範例以了解其工作原理：

計算方式如下：

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

對於快照預留 = 50% 且 PVC 請求 = 5 GiB 的情況，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 請求中請求的大小。 `volume show`命令應顯示與此範例類似的結果：

```
Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- -----
                  _pvc_89f1c156_3801_4de4_9f9d_034d54c395f4
                                   online     RW         10GB     5.00GB    0%
                  _pvc_e8372153_9ad9_474a_951a_08ae15e1c0ba
                                   online     RW          1GB    511.8MB    0%
2 entries were displayed.
```

升級Trident時，先前安裝的現有後端將以上述方式設定磁碟區。對於升級前建立的捲，您應該調整其大小以觀察到變更。例如，一個 2 GiB 的 PVC 包含 `snapshotReserve=50`先前的結果是卷提供了 1 GiB 的可寫空間。例如，將磁碟區大小調整為 3 GiB，將在 6 GiB 的磁碟區上為應用程式提供 3 GiB 的可寫入空間。

最低組態範例

下列範例顯示基本組態、讓大部分參數保留預設值。這是定義後端最簡單的方法。

> ⓘ 如果您在NetApp ONTAP 支援Trident的NetApp支援上使用Amazon FSX、建議您指定lifs的DNS名稱、而非IP位址。

ONTAP NAS 經濟效益範例

```yaml
---
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

```yaml
---
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、避免在切換和切換期間手動更新後端定義 "SVM 複寫與還原"。

若要無縫切換和切換、請使用指定 SVM `managementLIF` 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 Volume 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
```

憑證型驗證範例

這是最小的後端組態範例。 `clientCertificate`、 `clientPrivateKey`和 `trustedCACertificate` （選用、如果使用信任的CA）會填入 `backend.json` 並分別取得用戶端憑證、私密金鑰及信任CA憑證的基礎64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

本範例說明如何指示 Trident 使用動態匯出原則來自動建立及管理匯出原則。和 `ontap-nas-flexgroup`驅動程式的運作方式相同 `ontap-nas-economy`。

```
---
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

此範例顯示 `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
```

Amazon FSX for ONTAP 使用 SMB Volume 範例

- `smbShare` 使用 SMB 磁碟區的 ONTAP 需要 FSX 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_
```

名稱範本的後端組態範例

```
---
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}}"
```

虛擬集區的後端範例

在下面顯示的後端定義檔案範例中、會針對所有儲存池設定特定的預設值、例如 `spaceReserve` 無、`spaceAllocation` 假、和 `encryption` 錯。虛擬資源池是在儲存區段中定義的。

Trident 會在「意見」欄位中設定資源配置標籤。註解是在 FlexVol for 或 FlexGroup for `ontap-nas-flexgroup` 上設定 `ontap-nas`。Trident 會在資源配置時、將虛擬集區上的所有標籤複製到儲存磁碟區。為了方便起見、儲存管理員可以針對每個虛擬資源池定義標籤、並依標籤將磁碟區分組。

在這些範例中、有些儲存池是自行設定的 `spaceReserve`、 `spaceAllocation`和 `encryption` 值、而某些資源池會覆寫預設值。

ONTAP NAS 範例

```yaml
---
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

```yaml
---
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 經濟效益範例

```yaml
---
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StorageClass

請參閱下列 StorageClass 定義 [虛擬集區的後端範例]。使用 `parameters.selector` 欄位中、每個 StorageClass 都會呼叫哪些虛擬集區可用於主控磁碟區。磁碟區將會在所選的虛擬資源池中定義各個層面。

- 。 `protection-gold` StorageClass 會對應至中的第一個和第二個虛擬集區 `ontap-nas-flexgroup` 後端：這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"
```

- 。 `protection-not-gold` StorageClass 會對應至中的第三和第四個虛擬集區 `ontap-nas-flexgroup` 後端：這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 會對應至中的第四個虛擬集區 `ontap-nas` 後端：這是唯一為 mysqldb 類型應用程式提供儲存池組態的集區。

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

- t `protection-silver-creditpoints-20k` StorageClass 會對應至中的第三個虛擬集區 `ontap-nas-flexgroup` 後端：這是唯一提供銀級保護和 20000 個信用點數的資源池。

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

- 。 `creditpoints-5k` StorageClass 會對應至中的第三個虛擬集區 `ontap-nas` 後端和中的第二個虛擬集區 `ontap-nas-economy` 後端：這是唯一擁有 5000 個信用點數的集區方案。

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

Trident 會決定要選取哪個虛擬集區、並確保符合儲存需求。

更新 `dataLIF` 初始組態之後

您可以在初始設定後變更 dataLIF ，方法是執行下列命令，以更新的 dataLIF 提供新的後端 JSON 檔案。

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

如果 PVCS 連接到一個或多個 Pod ，您必須關閉所有對應的 Pod ，然後重新啟動，新的 dataLIF 才會生效。

安全 **SMB** 範例

使用 **ontap-nas** 驅動程式的後端配置

```
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** 驅動程式的後端配置

```
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
```

具有儲存池的後端配置

```yaml
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
```

採用 **ontap-nas** 驅動程式的儲存類別範例

```yaml
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
```

確保添加 `annotations`啟用安全 SMB。如果沒有註釋，安全 SMB 就無法運作，無論後端或 PVC 中設定了什麼配置。

採用 **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
```

具有單一 **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
```

具有多個 **AD** 使用者的 **PVC** 範例

```yaml
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 產品

搭配 **Amazon FSX for NetApp ONTAP** 使用 **Trident**

"Amazon FSX for NetApp ONTAP 產品" 是完全託管的AWS服務、可讓客戶啟動及執行採用NetApp ONTAP 資訊儲存作業系統的檔案系統。FSX for ONTAP VMware可讓您運用熟悉的NetApp功能、效能和管理功能、同時充分發揮儲存AWS資料的簡易性、敏捷度、安全性和擴充性。FSX for ONTAP Sfor支援ONTAP Isf供 檔案系統功能和管理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 叢集可配置由 ONTAP 備份的區塊和檔案持續磁碟區。

檔案系統是Amazon FSX的主要資源、類似ONTAP 於內部部署的一個叢集。在每個SVM中、您可以建立一個或多個磁碟區、這些磁碟區是儲存檔案系統中檔案和資料夾的資料容器。Amazon FSX for NetApp ONTAP 將以雲端託管檔案系統的形式提供。新的檔案系統類型稱為* NetApp ONTAP Sing*。

使用 Trident 搭配 Amazon FSX for NetApp ONTAP 、您可以確保在 Amazon Elastic Kubernetes Service （EKS）中執行的 Kubernetes 叢集可以佈建由 ONTAP 支援的區塊和檔案持續性磁碟區。

需求

除了"Trident 需求"、若要將適用於 ONTAP 的 FSX 與 Trident 整合、您還需要：

- 現有的Amazon EKS叢集或自行管理的Kubernetes叢集、已安裝「kubecll」。
- 可從叢集工作節點存取的現有 Amazon FSX for NetApp ONTAP 檔案系統和儲存虛擬機器（SVM）。
- 已準備好的工作節點 "NFS或iSCSI"。

> 請務必遵循Amazon Linux和Ubuntu所需的節點準備步驟 "Amazon機器映像" （AMis）、視您的EKS AMI類型而定。

考量

- SMB Volume：
  - 使用支援SMB磁碟區 `ontap-nas` 僅限驅動程式。
  - Trident EKS 附加元件不支援 SMB Volume 。
  - Trident 僅支援掛載至 Windows 節點上執行的 Pod 的 SMB 磁碟區。如 "準備配置SMB磁碟區" 需詳細資訊、請參閱。
- 在 Trident 24.02 之前、在已啟用自動備份的 Amazon FSX 檔案系統上建立的磁碟區、無法由 Trident 刪除。若要在 Trident 24.02 或更新版本中避免此問題、請在 AWS FSX for ONTAP 的後端組態檔案中指定 `fsxFilesystemID`、 AWS `apiRegion`、 AWS `apikey` 和 AWS `secretKey` 。

> 如果您要將 IAM 角色指定給 Trident 、則可以省略將、 `apiKey`和 `secretKey`欄位明確指定 `apiRegion`給 Trident 。如需詳細資訊、請 "FSX提供ONTAP 各種組態選項和範例"參閱。

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）一起使用，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。為了確保多路徑功能不會幹擾同一節點上的 EBS 磁碟，您需要在多路徑設定中排除 EBS。這個例子展示了 `multipath.conf`包含所需Trident設定的文件，同時將 EBS 磁碟排除在多路徑之外：

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

驗證

Trident 提供兩種驗證模式。

- 認證型（建議）：在 AWS Secrets Manager 中安全地儲存認證。您可以將使用者用於檔案系統、或是使用 `fsxadmin` `vsadmin` 為 SVM 設定的使用者。

> Trident 應以 SVM 使用者或具有相同角色之不同名稱的使用者身分執行 `vsadmin`。Amazon FSX for NetApp ONTAP 的 `fsxadmin``使用者僅能有限地取代 `ONTAP` `admin` 叢集使用者。我們強烈建議搭配 Trident 使用 `vsadmin`。

- 憑證型： Trident 將使用 SVM 上安裝的憑證、與 FSX 檔案系統上的 SVM 通訊。

如需啟用驗證的詳細資訊、請參閱您的驅動程式類型驗證：

- "ASNAS驗證ONTAP"
- "支援SAN驗證ONTAP"

已測試的 **Amazon Machine** 映像（**Amis**）

EKS 叢集支援各種作業系統，但 AWS 已針對容器和 EKS 最佳化某些 Amazon Machine 映像（Amis）。以下 AMI 已通過 NetApp Trident 25.02 測試。

Ami	NAS	NAS 經濟效益	iSCSI	iSCSI經濟型
AL2023_x86_64_STANDARD	是的	是的	是的	是的
AL2_x86_64	是的	是的	是*	是*
BOTTLEROCKET_x86_64	是 **	是的	不適用	不適用
AL2023_ARM_64_STANDARD	是的	是的	是的	是的
AL2_ARM_64	是的	是的	是*	是*
BOTTLEROCKET_ARM_64	是 **	是的	不適用	不適用

- * 如果不重新啟動節點，則無法刪除 PV
- ** 不適用於Trident版本 25.02 的 NFSv3。

> 如果此處未列出您想要的 AMI ，並不表示不支援，只是表示尚未測試。此清單可作為已知可運行的 AMI 的指南。

- 使用 * 執行的測試：
- EKS版本：1.32
- 安裝方法：Helm 25.06 和 AWS 附加元件 25.06
- 對於 NAS ， NFSv3 和 NFSv4.1 都已經過測試。

- 僅針對 SAN 進行 iSCSI 測試，非 NVMe 型。
- 已執行的測試 * ：
- 建立：儲存類別， PVC ， Pod
- 刪除： Pod ， PVC （一般， qtree /LUN –經濟， NAS 搭配 AWS 備份）

如需詳細資訊、請參閱

- "Amazon FSX for NetApp ONTAP 的支援文件"
- "Amazon FSX for NetApp ONTAP 的部落格文章"

建立 **IAM** 角色和 **AWS** 密碼

您可以將 Kubernetes Pod 設定為以 AWS IAM 角色進行驗證、而非提供明確的 AWS 認證、以存取 AWS 資源。

> 若要使用 AWS IAM 角色進行驗證、您必須使用 EKS 部署 Kubernetes 叢集。

建立 **AWS Secrets Manager** 密碼

由於 Trident 將針對 FSX Vserver 發行 API ，以便為您管理儲存設備，因此需要認證才能這麼做。傳遞這些認證的安全方法是透過 AWS Secrets Manager 密碼。因此，如果您還沒有，就必須建立 AWS Secrets Manager 密碼，其中包含 vsadmin 帳戶的認證。

此範例建立 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** 原則

Trident 也需要 AWS 權限才能正確執行。因此，您需要建立一個原則，讓 Trident 擁有所需的權限。

下列範例使用 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"
```

- 政策 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"
}
```

為服務帳戶關聯 **(IRSA)** 建立 **Pod Identity** 或 **IAM** 角色

您可以使用 EKS Pod Identity 設定 Kubernetes 服務帳戶，使其代入 AWS Identity and Access Management (IAM) 角色，或使用 IAM 角色進行服務帳戶關聯 (IRSA)。任何已配置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 身份

Amazon EKS Pod Identity 關聯提供了管理應用程式憑證的能力，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。

在您的 **EKS** 叢集上安裝 **Pod Identity**：

您可以透過 AWS 主控台或使用下列 AWS CLI 指令建立 Pod 身分：

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

更多資訊請參閱"設定 Amazon EKS Pod Identity Agent" 。

創建 **trust-relationship.json**：

建立 trust-relationship.json 文件，使 EKS 服務主體能夠承擔 Pod Identity 的此角色。然後建立一個具有以下信任策略的角色：

```
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 文件：

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

將角色策略附加到 **IAM** 角色：

將上一個步驟中的角色策略附加到已建立的 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
```

- 信任關係 .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"
        }
      }
    }
  ]
}
```

更新檔案中的下列值 `trust-relationship.json`：

- * <account_id> * - 您的 AWS 帳戶 ID
- * <oidc_provider> * - EKS 叢集的 OIDC 。您可以執行下列項目來取得 oidc_provider ：

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

- 使用 IAM 原則附加 IAM 角色 * ：

建立角色後，請使用以下命令將原則（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 * ：

確認您的 OIDC 供應商與您的叢集相關聯。您可以使用下列命令來驗證：

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

如果輸出為空，請使用下列命令將 IAM OIDC 與叢集建立關聯：

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

如果您使用 **eksctl**，請使用下列範例為 EKS 中的服務帳戶建立 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 中適用於 NetApp ONTAP 儲存管理的 Amazon FSX 、讓開發人員和管理員能夠專注於應用程式部署。

您可以使用下列其中一種方法來安裝 Trident ：

- 掌舵
- EKS 附加元件

如果您想要使用快照功能，請安裝 CSI Snapshot 控制器附加元件。如需詳細資訊、請參閱 "啟用 CSI Volume 的快照功能" 。

透過 helm 安裝 Trident

Pod 身份

1. 新增Trident Helm儲存庫：

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

2. 使用下列範例安裝 Trident：

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

您可以使用 `helm list`命令檢閱安裝詳細資料，例如名稱，命名空間，圖表，狀態，應用程式版本和修訂版編號。

```
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
```

服務帳戶協會 (IRSA)

1. 新增Trident Helm儲存庫：

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

2. 設定「雲端提供者」和「雲端身分」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`命令檢閱安裝詳細資料，例如名稱，命名空間，圖表，狀態，應用程式版本和修訂版編號。

```
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，請確保在用戶端電腦上啟用了 iSCSI。如果您使用的是 AL2023 工作節點作業系統，則可以透過在 helm 安裝中新增 node prep 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** 附加元件安裝 **Trident**

Trident EKS 附加元件包含最新的安全性修補程式、錯誤修正、並經過 AWS 驗證、可與 Amazon EKS 搭配使用。EKS 附加元件可讓您持續確保 Amazon EKS 叢集安全穩定、並減少安裝、設定及更新附加元件所需的工作量。

先決條件

在設定 AWS EKS 的 Trident 附加元件之前、請確定您具有下列項目：

- 具有附加訂閱的 Amazon EKS 叢集帳戶
- AWS 對 AWS 市場的權限：
  `"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 **AWS** 的 **Trident** 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. 在左側導航窗格中，選擇 **Clusters** 。
3. 選取您要設定 NetApp Trident CSI 附加元件的叢集名稱。
4. 選取 * 附加元件 * ，然後選取 * 取得更多附加元件 * 。
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 NetApp 的 Trident 方塊右上角的核取方塊。
   c. 選擇*下一步*。
6. 在 * 設定選取的附加元件 * 設定頁面上、執行下列步驟：

   *如果您使用 Pod Identity 關聯，請跳過這些步驟。 *

   a. 選擇您要使用的 * 版本 * 。
   b. 如果您使用 IRSA 驗證，請確保設定可選組態設定中可用的設定值：
      - 選擇您要使用的 * 版本 * 。
      - 依照 附加元件配置模式，將 配置值 部分中的 **configurationValues** 參數設定為您在上一個步驟中建立的角色 ARN（值應採用以下格式）：

```
{

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

}
```

   +
   如果您為衝突解決方法選取「覆寫」、則現有附加元件的一或多個設定可以使用 Amazon EKS 附加元件設定覆寫。如果您未啟用此選項、且與現有設定發生衝突、則作業將會失敗。您可以使用產生的錯誤訊息來疑難排解衝突。選取此選項之前、請確定 Amazon EKS 附加元件不會管理您需要自行管理的設定。

7. 選擇 * 下一步 * 。
8. 在 * 檢閱及新增 * 頁面上、選擇 * 建立 * 。

   附加元件安裝完成後、您會看到已安裝的附加元件。

AWS CLI

1.創建 `**add-on.json**`文件：

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** 驗證，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"
  }
}
```

取代 `<role ARN>`前一步驟所建立角色的 ARN 。

*2.安裝 Trident EKS 外掛程式。 *

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

eksctl

下列範例命令會安裝 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. 打開 Amazon EKS 控制檯 https://console.aws.amazon.com/eks/home#/clusters。
2. 在左側導航窗格中，選擇 **Clusters** 。
3. 選取您要更新 NetApp Trident CSI 附加元件的叢集名稱。
4. 選取 * 附加元件 * 索引標籤。
5. 選取 * Trident by NetApp * ，然後選取 * 編輯 * 。
6. 在 * Configure Trident by NetApp * 頁面上、執行下列步驟：
   a. 選擇您要使用的 * 版本 * 。
   b. 展開 * 選用組態設定 * ，並視需要修改。
   c. 選取*儲存變更*。

AWS CLI

下列範例更新 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 FSxN Trident CSI 附加元件的目前版本。以叢集名稱取代 `my-cluster`。

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

- 輸出範例： *

```
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個步驟輸出中可用更新所傳回的版本。

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

如果您移除此 `--force` 選項、且任何 Amazon EKS 附加元件設定與您現有的設定發生衝突、則更新 Amazon EKS 附加元件會失敗；您會收到錯誤訊息、協助您解決衝突。在指定此選項之前、請確定 Amazon EKS 附加元件不會管理您需要管理的設定、因為這些設定會以此選項覆寫。如需此設定的其他選項的詳細資訊，請參閱 "附加元件"。如需 Amazon EKS Kubernetes 現場管理的詳細資訊、請參閱 "Kubernetes 現場管理"。

解除安裝 / 移除 **Trident EKS** 附加元件

您有兩種移除 Amazon EKS 附加元件的選項：

- * 保留叢集上的附加軟體 * –此選項會移除 Amazon EKS 對任何設定的管理。它也會移除 Amazon EKS 通知您更新的功能、並在您啟動更新後自動更新 Amazon EKS 附加元件。不過、它會保留叢集上的附加軟體。此選項可讓附加元件成為自我管理的安裝、而非 Amazon EKS 附加元件。有了這個選項、附加元件就不會停機。保留 `--preserve` 命令中的選項以保留附加元件。
- * 從叢集完全移除附加軟體 * – NetApp 建議您只有在叢集上沒有任何相關資源的情況下，才從叢集移除 Amazon EKS 附加元件。從命令中移除 `--preserve` 選項 `delete` 以移除附加元件。

如果附加元件有相關的 IAM 帳戶、則不會移除 IAM 帳戶。

管理主控台

1. 開啟 Amazon EKS 主控台： https://console.aws.amazon.com/eks/home#/clusters。
2. 在左導覽窗格中，選取 * 叢集 * 。
3. 選取您要移除 NetApp Trident CSI 附加元件的叢集名稱。
4. 選擇 **Add-ons** 標籤，然後選擇 Trident by NetApp * 。 *
5. 選擇*移除*。
6. 在 * 移除 NetApp_trident 操作員確認 * 對話方塊中、執行下列步驟：
   a. 如果您想要 Amazon EKS 停止管理附加元件的設定、請選取 * 保留在叢集 * 上。如果您想要保留叢集上的附加軟體、以便自行管理附加元件的所有設定、請執行此動作。
   b. 輸入 **NetApp_trident － operer** 。
   c. 選擇*移除*。

AWS CLI

以叢集名稱取代 `my-cluster` 、然後執行下列命令。

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

eksctl

下列命令會解除安裝 Trident EKS 附加元件：

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

設定儲存後端

整合SAN和NAS驅動程式ONTAP

若要建立儲存後端，您需要以 JSON 或 YAML 格式建立組態檔案。檔案需要指定您想要的儲存類型（ NAS 或 SAN ），檔案系統和 SVM ，才能從中取得，以及如何驗證。以下範例說明如何定義 NAS 型儲存設備，以及如何使用 AWS 密碼將認證儲存至您要使用的 SVM ：

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及驗證 Trident 後端組態（TBC）：

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

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

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

- 驗證已成功建立 Trident 後端組態（TBC）：

```
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
```

適用於 **ONTAP** 驅動程式詳細資料的 **FSX**

您可以使用下列驅動程式、將 Trident 與 Amazon FSX for NetApp ONTAP 整合：

- `ontap-san`：配置的每個 PV 都是其各自 Amazon FSX 中的 LUN （用於 NetApp ONTAP Volume）。建議用於區塊儲存。
- `ontap-nas`：配置的每個 PV 都是 NetApp ONTAP Volume 的完整 Amazon FSX 。建議用於 NFS 和 SMB 。
- 「ONTAP-san經濟型」：每個配置的PV都是LUN、每個Amazon FSX for NetApp ONTAP 的LUN數量可設定。
- 「ONTAP-NAS-EAS'：每個提供的PV都是qtree、每個Amazon FSX的NetApp ONTAP 功能是可設定的配額樹數。
- 「ONTAP-NAS-Flexgroup」：每個提供的PV都是適用於NetApp ONTAP FlexGroup 的完整Amazon FSX。

如需驅動程式詳細資料、請參閱 "NAS 驅動程式" 和 "SAN 驅動程式"。

建立組態檔案後，請執行此命令，在 EKS 中建立：

```
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

如需後端組態選項、請參閱下表：

參數	說明	範例
「分度」		永遠為1
「torageDriverName」	儲存驅動程式名稱	`ontap-nas`、 `ontap-nas-economy`、 `ontap-nas-flexgroup`、 `ontap-san`、 `ontap-san-economy`
「後端名稱」	自訂名稱或儲存後端	驅動程式名稱 + "_" + dataLIF
《馬納格門達利》	叢集或 SVM 管理 LIF 的 IP 位址可以指定完整網域名稱（ FQDN ）。如果使用 IPv6 旗標安裝 Trident 、則可設定為使用 IPv6 位址。IPv6位址必須以方括弧來定義、例如[28e8：d9fb：a825：b7bf：69a8：d02f：9e7b：3555]。如果您在欄位下方 `aws`提供、則 `fsxFilesystemID`不需要提供、 `managementLIF`因為 `Trident` 會從 `AWS` 擷取 `SVM` `managementLIF` 資訊。因此、您必須在 SVM 下提供使用者的認證（例如： vsadmin ）、且使用者必須具有該 `vsadmin` 角色。	"10.0.0.1 "，"[2001:1234:abcd::fefe]

參數	說明	範例
「DataLIF」	傳輸協定LIF的IP位址。* ONTAP NAS 驅動程式 * ： NetApp 建議指定 dataLIF 。如果未提供， Trident 會從 SVM 擷取 dataLIFs 。您可以指定完整網域名稱（ FQDN ），以用於 NFS 裝載作業，讓您建立循環 DNS ，以便在多個 dataLIFs 之間進行負載平衡。可在初始設定之後變更。請參閱 。《**SAN**驅動程式：請勿指定用於iSCSI》ONTAP 。Trident 使用 ONTAP 選擇性 LUN 對應來探索建立多重路徑工作階段所需的 iSCI 生命。如果明確定義dataLIF、就會產生警告。如果使用 IPv6 旗標安裝 Trident 、則可設定為使用 IPv6 位址。IPv6位址必須以方括弧來定義、例如[28e8：d9fb：a825：b7bf：69a8：d02f：9e7b：3555]。	
「AutoExpportPolicy」	啟用自動匯出原則建立及更新[布 林值]。使用 `autoExportPolicy`和 `autoExportCIDRs`選項、 Trident 可以自動管理匯出原則。	「假」
《AutoExpportCIDR》（自動匯出CTR）	將 Kubernetes 節點 IP 篩選在啟用時的 CIDR 清單 `autoExportPolicy`。使用 `autoExportPolicy`和 `autoExportCIDRs`選項、 Trident 可以自動管理匯出原則。	"["0.0.0/0" ， ":/0"]"
《標籤》	套用到磁碟區的任意JSON-格式化標籤集	"
「用戶端憑證」	用戶端憑證的Base64編碼值。用於憑證型驗證	"
「clientPrivate Key」	用戶端私密金鑰的Base64編碼值。用於憑證型驗證	"
「可信賴的CACertifcate」	受信任CA憑證的Base64編碼值。選用。用於憑證型驗證。	"
《使用者名稱》	連線至叢集或SVM的使用者名稱。用於認證型驗證。例如、vsadmin。	
密碼	連線至叢集或SVM的密碼。用於認證型驗證。	
《虛擬機器》	要使用的儲存虛擬機器	指定SVM管理LIF時衍生。
「toragePrefix」	在SVM中配置新磁碟區時所使用的前置碼。無法在建立後修改。若要更新此參數、您需要建立新的後端。	`trident`

參數	說明	範例
「限制Aggregateusage」	* 請勿指定 Amazon FSX for NetApp ONTAP 。 *提供的 `fsxadmin`和 `vsadmin`不包含使用 Trident 擷取彙總使用量並加以限制所需的權限。	請勿使用。
《限制Volume大小》	如果要求的磁碟區大小高於此值、則資源配置失敗。也會限制其管理 qtree 和 LUN 的最大磁碟區大小，而且此 `qtreesPerFlexvol`選項可讓您自訂每個 FlexVol volume 的最大 qtree 數量	""（預設不強制執行）
《lunsPerFlexvol》	每個 FlexVol volume 的最大 LUN 數必須在 [50 ， 200] 範圍內。僅限 SAN 。	“100”
「DebugTraceFlags」	疑難排解時要使用的偵錯旗標。例如、 { "api" ： false 、 "method" ： true} 請勿使用 `debugTraceFlags` 除非您正在疑難排解並需要詳細的記錄傾印。	null
「nfsMountOptions」	以逗號分隔的NFS掛載選項清單。Kubernetes-Persistent Volume 的掛載選項通常是在儲存類別中指定、但如果儲存類別中未指定掛載選項、則 Trident 會回復為使用儲存後端組態檔案中指定的掛載選項。如果儲存類別或組態檔案中未指定任何掛載選項、 Trident 將不會在關聯的持續磁碟區上設定任何掛載選項。	"
`nasType`	設定NFS或SMB磁碟區建立。選項包括 `nfs`、 `smb``或`null`。*必須設定為 `smb` 對於SMB Volume。*設定為null、預設為NFS Volume。	`nfs`
"qtreesPerFlexvol"	每個 FlexVol volume 的最大 qtree 數必須在範圍 [50 ， 300]	`"200"`
`smbShare`	您可以指定下列其中一項：使用 Microsoft 管理主控台或 ONTAP CLI 建立的 SMB 共用名稱、或是允許 Trident 建立 SMB 共用的名稱。ONTAP 後端的 Amazon FSX 需要此參數。	`smb-share`

參數	說明	範例
《useREST》	使用ONTAP Isrest API的布林參數。設為 `true`時、 `Trident` 將使用 `ONTAP REST API` 與後端通訊。此功能需要`ONTAP` 使用更新版本的版本。此外、使用的 `ONTAP` 登入角色必須具有應用程式存取權 `ontap` 。這是預先定義的和角色所滿足 `vsadmin` `cluster-admin` 的。	「假」
`aws`	您可以在 AWS FSX for ONTAP 的組態檔中指定下列項目： - `fsxFilesystemID`：指定 AWS FSX 檔案系統的 ID 。 - `apiRegion`： AWS API 區域名稱。 - `apikey`： AWS API 金鑰。 - `secretKey`： AWS 秘密金鑰。	"" "" ""
`credentials`	指定要儲存在 AWS Secrets Manager 中的 FSX SVM 認證。- `name`：機密的 Amazon 資源名稱（ ARN ）、其中包含 SVM 的認證。- `type`：設為 `awsarn`。如需詳細資訊、請參閱 "建立 AWS Secrets Manager 密碼" 。	

用於資源配置磁碟區的後端組態選項

您可以使用中的這些選項來控制預設資源配置 `defaults` 組態區段。如需範例、請參閱下列組態範例。

參數	說明	預設
"paceAllocate（配置）"	LUN的空間分配	"真的"
《保護區》	空間保留模式；「無」（精簡）或「 Volume 」（粗）	無
「快照原則」	要使用的Snapshot原則	無
「qosPolicy」	要指派給所建立磁碟區的QoS原則群組。選擇每個儲存集區或後端的其中一個qosPolicy或adaptiveQosPolicy。搭配 Trident 使用 QoS 原則群組需要 ONTAP 9 8 或更新版本。您應該使用非共用的 QoS 原則群組、並確保個別將原則群組套用至每個成員。共享 QoS 原則群組會強制執行所有工作負載總處理量的上限。	"

參數	說明	預設
《adaptiveQosPolicy》	要指派給所建立磁碟區的調適性QoS原則群組。選擇每個儲存集區或後端的其中一個qosPolicy或adaptiveQosPolicy。不受ONTAP-NAS-經濟支援。	"
「快照保留區」	為快照保留的磁碟區百分比「 0 」	如果 `snapshotPolicy`是 `none`，`else`
「PlitOnClone」	建立複本時、從其父複本分割複本	「假」
加密	在新磁碟區上啟用 NetApp Volume Encryption （ NVE ）；預設為 `false`。必須在叢集上授權並啟用NVE、才能使用此選項。如果在後端啟用 NAE 、則 Trident 中配置的任何 Volume 都將啟用 NAE 。如需更多資訊、請參閱"Trident 如何與 NVE 和 NAE 搭配運作"：。	「假」
`luksEncryption`	啟用LUKS加密。請參閱 "使用Linux統一金鑰設定（LUKS）"。僅限SAN。	"
「分層政策」	要使用的分層原則 `none`	
「unixPermissions」	新磁碟區的模式。如果是**SMB**磁碟區、請保留空白。	"
《生態樣式》	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預設為 `unix`。SMB預設為 `ntfs`。

提供 **SMB** 卷

您可以使用以下方式設定 SMB 磁碟區：`ontap-nas`司機。在你完成之前 整合SAN和NAS驅動程式ONTAP 請完成以下步驟： "準備配置SMB磁碟區"。

設定儲存類別和 **PVC**

設定 Kubernetes StorageClass 物件並建立儲存類別、以指示 Trident 如何配置磁碟區。建立 PersistentVolume Claim （ PVC ），使用設定的 Kubernetes StorageClass 要求存取 PV 。然後、您可以將 PV 掛載至 Pod 。

建立儲存類別

設定 **Kubernetes StorageClass** 物件

這 "Kubernetes StorageClass 物件"物件將Trident標識為用於該類別的供應器，並指示Trident如何供應磁碟區。使用此範例為使用 NFS 的磁碟區設定 Storageclass（有關屬性的完整列表，請參閱下面的Trident屬性部分）：

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

使用此範例為使用 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 上佈建 NFSv3 磁碟區，請將必要的新增 `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
```

如需儲存類別如何與互動的詳細資訊 `PersistentVolumeClaim`、以及控制 Trident 配置磁碟區的參數、請參閱"Kubernetes和Trident物件"。

建立儲存類別

步驟

1. 這是 Kubernetes 物件、請使用 `kubectl` 在Kubernetes中建立。

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

2. 現在您應該會在 Kubernetes 和 Trident 中同時看到 * base-csi* 儲存類別、而 Trident 應該已經在後端上探索到這些集區。

```
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["_PersistentVolume Claim"^] （ PVC ）是存取叢集上 PersistentVolume
的要求。
```

可將 PVC 設定為要求儲存特定大小或存取模式。叢集管理員可以使用相關的 StorageClass 來控制超過 PersistentVolume 大小和存取模式的權限、例如效能或服務層級。

建立 PVC 之後，您可以將磁碟區裝入 Pod 。

範例資訊清單

PersistentVolume Claim 範例資訊清單

這些範例顯示基本的 PVC 組態選項。

可存取 **RWX** 的 **PVC**

此範例顯示具有 rwx 存取權的基本 PVC 、與名稱為的 StorageClass 相關聯 `basic-csi`。

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

使用 **iSCSI** 範例的 **PVC**

此範例展示了具有 RWO 存取權限的 iSCSI 基本 PVC，它與名為 `protection-gold`。

```
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. 確認 PVC 狀態。

```
kubectl get pvc
```

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

如需儲存類別如何與互動的詳細資訊 `PersistentVolumeClaim`、以及控制 Trident 配置磁碟區的參數、請參閱"Kubernetes和Trident物件"。

Trident 屬性

這些參數決定應使用哪些Trident託管儲存資源池來配置特定類型的磁碟區。

屬性	類型	價值	優惠	申請	支援者
媒體1^	字串	HDD、混合式、SSD	資源池包含此類型的媒體、混合式表示兩者	指定的媒體類型	ONTAP-NAS、ONTAP-NAS-經濟型、ONTAP-NAS-flexgroup、ONTAP-SAN、solidfire-san
資源配置類型	字串	纖薄、厚實	Pool支援此資源配置方法	指定的資源配置方法	厚：全ONTAP 是邊、薄：全ONTAP 是邊、邊、邊、邊、邊、邊、邊、邊、邊、邊、邊、邊
後端類型	字串	ontap-nas、ontap-nas-economy、ontap-nas-flexgroup、ontap-san、solidfire-san、azure-netapp-files、ontap-san-economy	集區屬於此類型的後端	指定後端	所有驅動程式
快照	布爾	對、錯	集區支援具有快照的磁碟區	已啟用快照的Volume	ontap-nas、ontap-san、solidfire-san
複製	布爾	對、錯	資源池支援複製磁碟區	已啟用複本的Volume	ontap-nas、ontap-san、solidfire-san

屬性	類型	價值	優惠	申請	支援者
加密	布爾	對、錯	資源池支援加密磁碟區	已啟用加密的Volume	ONTAP-NAS、ONTAP-NAS-經濟型、ONTAP-NAS-FlexGroups、ONTAP-SAN
IOPS	內部	正整數	集區能夠保證此範圍內的IOPS	Volume保證這些IOPS	solidfire-san

[1]：ONTAP Select 不受支援

部署範例應用程式

建立儲存類別和 PVC 時，您可以將 PV 掛載到 Pod 。本節列出範例命令和組態，以將 PV 附加至 Pod 。

步驟

1. 將磁碟區裝入 Pod 。

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

這些範例顯示將 PVC 附加至 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
```

您可以使用監控進度 `kubectl get pod --watch`。

2. 確認磁碟區已掛載到上 `/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
```

您現在可以刪除 Pod 。Pod 應用程式將不再存在、但該磁碟區仍會保留。

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

在 **EKS** 叢集上設定 **Trident EKS** 附加元件

NetApp Trident 簡化了 Kubernetes 中適用於 NetApp ONTAP 儲存管理的 Amazon FSX ，讓開發人員和管理員能夠專注於應用程式部署。NetApp Trident EKS 附加元件包含最新的安全性修補程式，錯誤修正，並經過 AWS 驗證，可與 Amazon EKS 搭配使用。EKS 附加元件可讓您持續確保 Amazon EKS 叢集安全穩定、並減少安裝、設定及更新附加元件所需的工作量。

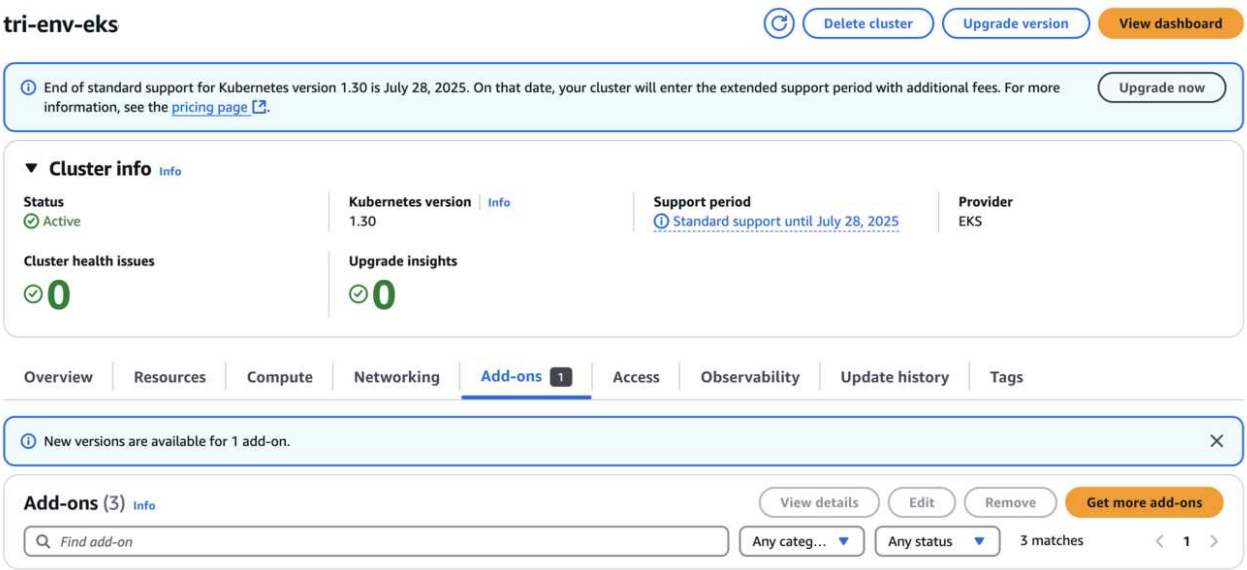
先決條件

在設定 AWS EKS 的 Trident 附加元件之前、請確定您具有下列項目：

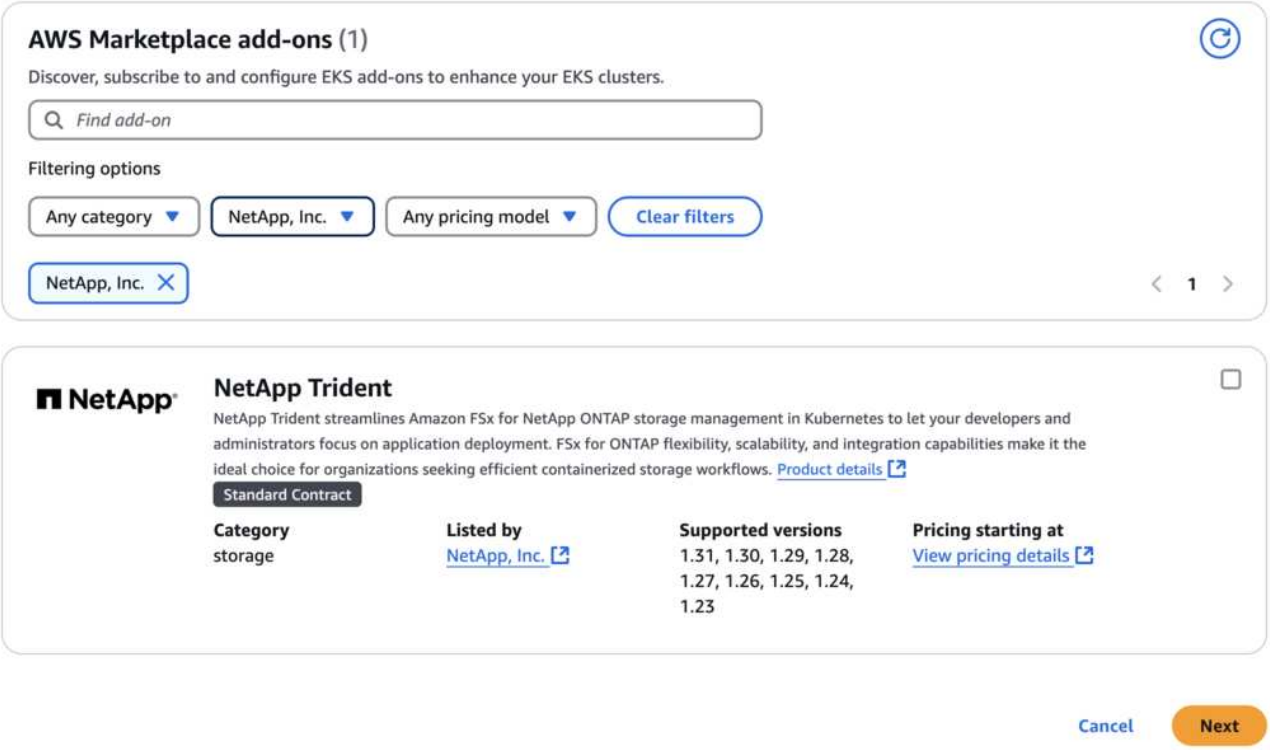
- 具有附加元件使用權限的 Amazon EKS 叢集帳戶。請參閱 "Amazon EKS 附加元件"。
- AWS 對 AWS 市場的權限：
  `"aws-marketplace:ViewSubscriptions",`
  `"aws-marketplace:Subscribe",`
  `"aws-marketplace:Unsubscribe`
- AMI 類型： Amazon Linux 2 （AL2_x86_64 ）或 Amazon Linux 2 ARM （AL2_ARM_64 ）
- 節點類型： AMD 或 ARM
- 現有的 Amazon FSX for NetApp ONTAP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

步驟

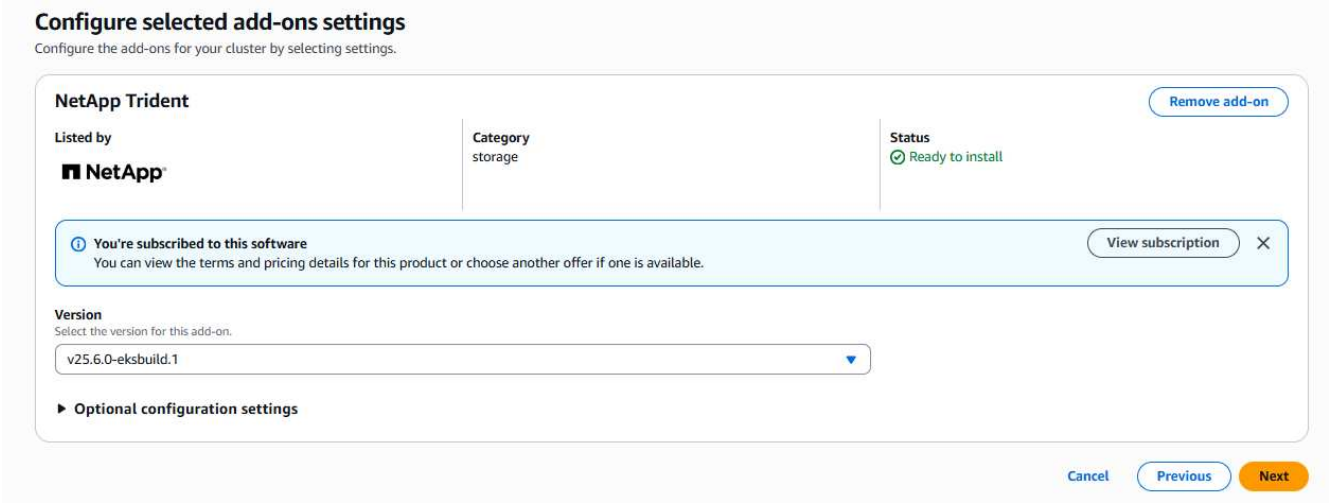
1. 請務必建立 IAM 角色和 AWS 密碼，讓 EKS Pod 能夠存取 AWS 資源。有關說明，請參閱"建立 IAM 角色和 AWS 密碼"。
2. 在 EKS Kubernetes 叢集上，瀏覽至 * 附加元件 * 索引標籤。



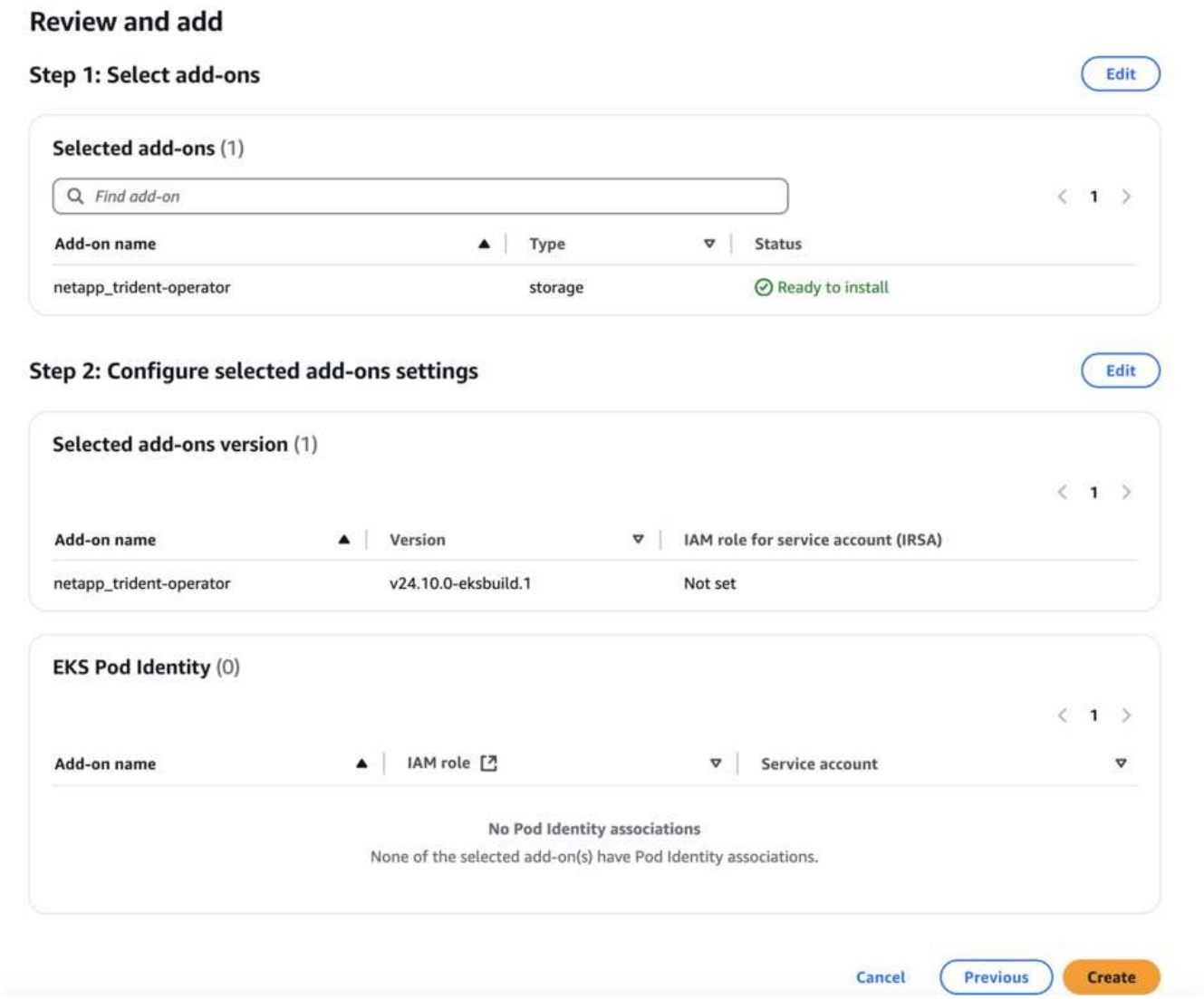
3. 前往 * AWS Marketplace 附加元件 * 並選擇 _storage 類別。



4. 找到 * NetApp Trident * 並選取 Trident 附加元件的核取方塊，然後按一下 * 下一步 * 。
5. 選擇所需版本的附加元件。

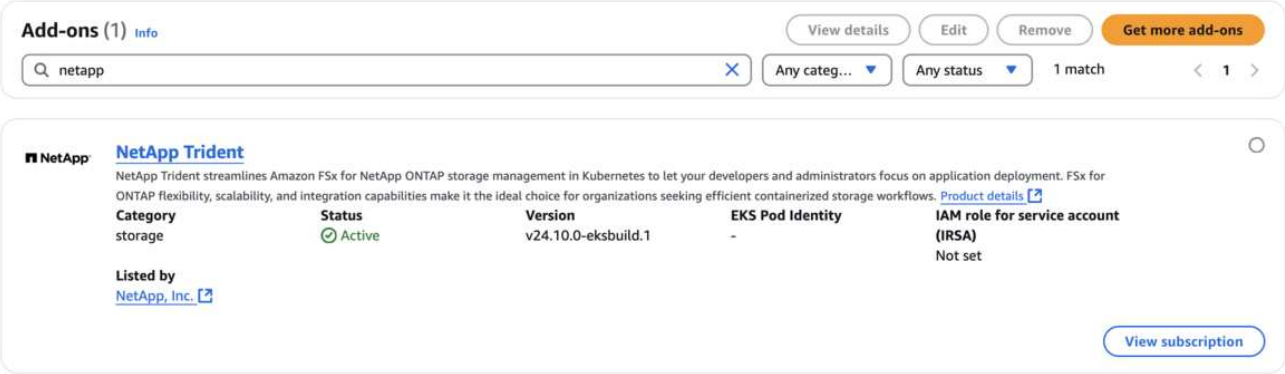


6. 配置所需的附加元件設定。



7. 如果您使用 IRSA（服務帳戶的 IAM 角色），請參閱其他設定步驟"請按這裡"。
8. 選擇* Create （建立）。

9. 確認附加元件的狀態為 *Active* 。



10. 執行下列命令，確認叢集上已正確安裝 Trident ：

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

11. 繼續設定並設定儲存後端。如需相關資訊，請參閱 "設定儲存後端"。

使用 **CLI** 安裝 / 解除安裝 **Trident EKS** 附加元件

使用 **CLI** 安裝 **NetApp Trident EKS** 附加元件：

以下範例指令安裝Trident EKS 附加元件：

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

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** 解除安裝 **NetApp Trident EKS** 附加元件：

下列命令會解除安裝 Trident EKS 附加元件：

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

使用**kubecl**建立後端

後端定義 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如何從該儲存系統配置磁碟區。安裝 Trident 之後、下一步是建立後端。`TridentBackendConfig`自訂資源定義（CRD）可讓您直接透過 Kubernetes 介面建立及管理 Trident 後端。您可以使用或等效的 CLI 工具來 `kubectl`進行 Kubernetes 發佈。

`TridentBackendConfig`

`TridentBackendConfig`(`tbc`、 `tbconfig`、 `tbackendconfig`）為前端、命名 CRD 、可讓您使用管理 Trident 後端 `kubectl`。Kubernetes 和儲存管理員現在可以直接透過 Kubernetes CLI 建立和管理後端（`tridentctl`、而不需要專用的命令列公用程式）。

建立「TridentBackendConfig」物件之後、會發生下列情況：

- Trident 會根據您提供的組態自動建立後端。這在內部表示爲 A `TridentBackend` (`tbe`、`tridentbackend`) CR 。
- `TridentBackendConfig``與由 `Trident` 建立的唯一繫結 `TridentBackend`。

每個「TridentBackendConfig」都有一對一的對應、並有「TridentBackend」。前者是提供給使用者設計及設定後端的介面、後者是Trident代表實際後端物件的方式。

> `TridentBackend`CRS` 是由 `Trident` 自動建立。您*不應該*修改這些項目。如果您想要更新後端、請修改物件以進行更新 `TridentBackendConfig`。

請參閱下列範例、以瞭解「TridentBackendConfig」CR的格式：

```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
```

您也可以查看中的範例 "Trident安裝程式" 所需儲存平台/服務的範例組態目錄。

。 `spec` 採用後端特定的組態參數。在此範例中、後端使用 `ontap-san` 儲存驅動程式、並使用此處列出的組態參數。如需所需儲存驅動程式的組態選項清單、請參閱 "儲存驅動程式的後端組態資訊"。

在《TridentBackendConfig》（CRR）中新推出的「sPEC」一節也包含「認證」和「刪除原則」欄位：

- 「認證資料」：此參數為必填欄位、包含用於驗證儲存系統/服務的認證資料。此設定為使用者建立的Kubernetes Secret。認證資料無法以純文字格式傳遞、因此會產生錯誤。
- 「刪除原則」：此欄位可定義刪除「TridentBackendConfig」時應發生的情況。可能需要兩種可能的值之一：
  - 「刪除」：這會同時刪除「TridentBackendConfig」和相關後端。這是預設值。
  - 「保留」：刪除「TridentBackendConfig」（TridentBackendConfig）CR時、後端定義仍會存在、並可

使用「tridentctl」進行管理。將刪除原則設為「保留」可讓使用者降級至較早版本（21.04之前）、並保留建立的後端。此欄位的值可在建立「TridentBackendConfig」之後更新。

> 後端名稱是使用「sPEC.backendName」來設定。如果未指定、則會將後端名稱設為「TridentBackendConfig」物件（metadata.name）的名稱。建議使用「sPEC.backendName」明確設定後端名稱。

> 使用建立的後端 `tridentctl`沒有關聯的 `TridentBackendConfig`物件。您可以建立 `CR` 來 `TridentBackendConfig`選擇管理此類後端 `kubectl`。必須注意指定相同的組態參數（例如 `spec.backendName`、`spec.storagePrefix`、`spec.storageDriverName`等）。`Trident` 會自動將新建立的後端與先前存在的後端繫結 `TridentBackendConfig`。

步驟總覽

若要使用「kubecll」建立新的後端、您應該執行下列動作：

1. 建立 "Kubernetes機密"。密碼包含 Trident 與儲存叢集 / 服務通訊所需的認證。
2. 建立「TridentBackendConfig」物件。其中包含有關儲存叢集/服務的詳細資訊、並參考上一步建立的機密。

建立後端之後、您可以使用「kubecl Get tbc <tbc-name>-n <trident命名空間>」來觀察其狀態、並收集其他詳細資料。

步驟**1**：建立**Kubernetes**機密

建立包含後端存取認證的秘密。這是每個儲存服務/平台所獨有的功能。範例如下：

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

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

下表摘要說明每個儲存平台的機密必須包含的欄位：

儲存平台機密欄位說明	秘密	欄位說明
Azure NetApp Files	ClientID	應用程式註冊的用戶端ID
元素（NetApp HCI / SolidFire）	端點	MVIP、適用於SolidFire 採用租戶認證的不含用戶身分證明的叢集

儲存平台機密欄位說明	秘密	欄位說明
ONTAP	使用者名稱	連線至叢集/ SVM的使用者名稱。用於認證型驗證
ONTAP	密碼	連線至叢集/ SVM的密碼。用於認證型驗證
ONTAP	用戶端權限金鑰	用戶端私密金鑰的Base64編碼值。用於憑證型驗證
ONTAP	chap使用 者名稱	傳入使用者名稱。如果useCHAP=true則需要。適用於「ONTAP-SAN」和「ONTAP-san經濟」
ONTAP	chapInitiator機密	CHAP啟動器密碼。如果useCHAP=true則需要。適用於「ONTAP-SAN」和「ONTAP-san經濟」
ONTAP	chapTargetUsername	目標使用者名稱。如果useCHAP=true則需要。適用於「ONTAP-SAN」和「ONTAP-san經濟」
ONTAP	chapTargetInitiator機密	CHAP目標啟動器機密。如果useCHAP=true則需要。適用於「ONTAP-SAN」和「ONTAP-san經濟」

在此步驟中建立的機密會參照下一步所建立之「TridentBackendConfig」物件的「sapec.ecent」欄位。

步驟**2**：建立 `TridentBackendConfig` **CR**

您現在可以建立「TridentBackendConfig」的CR了。在此範例中、使用「ONTAP-SAN」驅動程式的後端是使用「TridentBackendConfig」物件建立、如下所示：

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

```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：確認的狀態 `TridentBackendConfig` CR

現在您已經建立了「TridentBackendConfig」（TridentBackendConfig）CR、您就可以驗證其狀態。請參閱下列範例：

```
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
```

已成功建立後端、並連結至「TridentBackendConfig」CR。

階段可以採用下列其中一個值：

- `Bound`： `TridentBackendConfig` CR與後端相關聯、且後端包含 `configRef` 設定為 `TridentBackendConfig` CR 的 uid 。
- 《Unbound》：使用「」表示。「TridentBackendConfig」物件不會繫結至後端。根據預設、所有新建立的「TridentBackendConfig」CRS均處於此階段。階段變更之後、就無法再恢復為Unbound（未綁定）。
- `Deleting`： `TridentBackendConfig` CR 的 `deletionPolicy` 已設定為刪除。當 `TridentBackendConfig` 系統會刪除CR、並轉換為「刪除」狀態。
  - 如果後端不存在持續磁碟區宣告（ PVCS ）、刪除 `TridentBackendConfig`**將會導致** `Trident` **刪除後端和** `TridentBackendConfig` CR 。
  - 如果後端上有一個或多個PVCS、則會進入刪除狀態。隨後、「TridentBackendConfig」CR也會進入刪除階段。只有刪除所有的PVCS之後、才會刪除後端和「TridentBackendConfig」。
- 「遺失」：與「TridentBackendConfig」CR相關的後端意外或刻意刪除、而「TridentBackendConfig」CR仍有刪除後端的參考資料。無論「刪除原則」值為何、「TridentBackendConfig」CR仍可刪除。
- `Unknown`： Trident 無法確定與 CR 關聯的後端的狀態或存在 `TridentBackendConfig`。例如、如果 API 伺服器沒有回應、或 `tridentbackends.trident.netapp.io` CRD 遺失。這可能需要介入。

在此階段、成功建立後端！還有多種作業可以額外處理、例如 "後端更新和後端刪除"。

（選用）步驟**4**：取得更多詳細資料

您可以執行下列命令來取得有關後端的詳細資訊：

```
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、您也可以取得「TridentBackendConfig」的YAML/Json傾印。

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

```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`包含回應 `CR` 所建立後端 `TridentBackendConfig`的 `backendName`和 `backendUUID`。此 `lastOperationStatus`欄位代表 `CR` 上次操作的狀態 `TridentBackendConfig`、可由使用者觸發（例如、使用者在中變更項目 `spec`）或由 Trident 觸發（例如、在 Trident 重新啟動期間）。可能是「成功」或「失敗」。 `phase`代表 `CR` 與後端之間關係的狀態 `TridentBackendConfig`。在上述範例中、 `phase`有值界限、表示 `TridentBackendConfig` CR 與後端相關聯。

您可以執行「kubeclt -n triident描述tbc <tbc-cr-name>」命令、以取得事件記錄的詳細資料。

> 您無法使用「tridentctl」來更新或刪除包含相關「TridentBackendConfig」物件的後端。若要瞭解在「tridentctl」和「TridentBackendConfig」之間切換的步驟、 "請參閱此處"。

管理後端

以KECBECVL執行後端管理

瞭解如何使用「kubecll」來執行後端管理作業。

刪除後端

刪除 `TridentBackendConfig`後、您會指示 `Trident` 刪除 / 保留後端（根據 `deletionPolicy`）。若要刪除後端、請確定已 `deletionPolicy`設定為刪除。若要僅刪除 `TridentBackendConfig`、請確定已 `deletionPolicy`設定為保留。這可確保後端仍存在、並可使用進行管理 `tridentctl`。

執行下列命令：

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

Trident 不會刪除使用中的 Kubernetes 機密 `TridentBackendConfig`。Kubernetes使用者負責清除機密。刪除機密時必須小心。只有在後端未使用機密時、才應刪除這些機密。

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」、以取得所有後端的清單。這份清單也會包含以「tridentctl」建立的後端。

更新後端

更新後端可能有多種原因：

- 儲存系統的認證資料已變更。若要更新認證、必須更新物件中使用的 Kubernetes Secret `TridentBackendConfig` 。Trident 會使用提供的最新認證、自動更新後端。執行下列命令以更新Kubernetes Secret：

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

- 需要ONTAP 更新參數（例如使用的SVM名稱）。
  - 您可以更新 `TridentBackendConfig` 使用下列命令直接透過 Kubernetes 執行物件：

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

  - 或者、您也可以變更現有的 `TridentBackendConfig` 使用下列命令的 CR ：

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

- 如果後端更新失敗、後端仍會繼續維持其最後已知的組態。您可以執行「kubecl Get tbc <tbc-name>-o yaml -n trident」或「kubecl描述tbc <tbc-name>-n trident」來檢視記錄以判斷原因。
- 識別並修正組態檔的問題之後、即可重新執行update命令。

使用**tridentctl**執行後端管理

瞭解如何使用「tridentctl」來執行後端管理作業。

建立後端

建立之後 "後端組態檔"，執行下列命令：

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

如果後端建立失敗、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 -n trident
```

識別並修正組態檔的問題之後、您只需再次執行「create」命令即可。

刪除後端

若要從 Trident 刪除後端、請執行下列步驟：

1. 擷取後端名稱：

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

2. 刪除後端：

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

如果 Trident 已從這個後端佈建磁碟區和快照、但該後端仍存在、則刪除後端將會阻止新磁碟區由其進行佈建。後端將繼續存在於「刪除」狀態。

檢視現有的後端

若要檢視Trident知道的後端、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之後、請執行下列命令：

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

如果後端更新失敗、表示後端組態有問題、或是您嘗試了無效的更新。您可以執行下列命令來檢視記錄、以判斷原因：

```
tridentctl logs -n trident
```

識別並修正組態檔的問題之後、您只需再次執行「update」命令即可。

識別使用後端的儲存類別

這是您可以用Json回答的問題類型範例、其中的「tridentctl」會輸出後端物件。這會使用您需要安裝的「jq」公用程式。

```
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切換

瞭解在 Trident 中管理後端的不同方法。

管理後端的選項

隨之推出 `TridentBackendConfig`管理員現在有兩種獨特的後端管理方法。這會提出下列問題：

- 使用「tridentctl」建立的後端、是否能以「TridentBackendConfig」來管理？
- 使用「TridentBackendConfig」建立的後端、是否可以使用「tridentctl」來管理？

管理 `tridentctl` 後端使用 `TridentBackendConfig`

本節說明透過Kubernetes介面建立「TridentBackendConfig」物件、直接透過「tridentctl」建立的後端管理所需的步驟。

這將適用於下列案例：

- 沒有的既有後端 `TridentBackendConfig` 因為它們是使用建立的 `tridentctl`。
- 使用「tridentctl」建立的新後端、而其他「TridentBackendConfig」物件則存在。

在這兩種情況下、都會繼續出現後端、並在 Trident 排程磁碟區上運作。系統管理員有兩種選擇之一：

- 繼續使用「tridentctl」來管理使用它建立的後端。
- 將使用「tridentctl」建立的後端連結至新的「TridentBackendConfig」物件。這樣做將意味着後端將使用“kubedl”而不是“tridentctl”來管理。

若要使用「kubecll」管理預先存在的後端、您需要建立連結至現有後端的「TridentBackendConfig」。以下是如何運作的總覽：

1. 建立Kubernetes機密。機密包含 Trident 與儲存叢集 / 服務通訊所需的認證。
2. 建立「TridentBackendConfig」物件。其中包含有關儲存叢集/服務的詳細資訊、並參考上一步建立的機密。必須謹慎指定相同的組態參數（例如「s.pec.backendName」、「sec.storagePrefix」、「sPEec.storageDriverName」等）。必須將「Pec.backendName」設定為現有後端的名稱。

步驟**0**：識別後端

以建立 `TridentBackendConfig` 若要連結至現有的後端、您必須取得後端組態。在此範例中、假設使用下列Json定義建立後端：

```
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機密

建立包含後端認證的秘密、如以下範例所示：

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

```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

下一步是建立一個「TridentBackendConfig」（TridentBackendConfig）CR、它會自動連結至現有的「ONTAP-NAS-backend」（如本範例所示）。確保符合下列要求：

- 相同的後端名稱是在「s.pec.backendName」中定義。
- 組態參數與原始後端相同。
- 虛擬資源池（若有）必須維持與原始後端相同的順序。
- 認證資料是透過Kubernetes Secret提供、而非以純文字提供。

在這種情況下、「TridentBackendConfig」將會如下所示：

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

```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：確認的狀態 `TridentBackendConfig` CR

在建立「TridentBackendConfig」之後、其階段必須是「綁定」。它也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 |
+---------------------+----------------
+--------------------------------------+--------+---------+
```

現在可以使用「tbc-ontap-nas-backend」「TridentBackendConfig」物件來完全管理後端。

管理 `TridentBackendConfig` 後端使用 `tridentctl`

可以使用「tridentctl」來列出使用「TridentBackendConfig」建立的後端。此外、系統管理員也可以刪除「TridentBackendConfig」、並確定「pec.deletionPolicy`」設為「效能」、藉此選擇透過「tridentctl」來完全管理此類後端。

步驟0：識別後端

例如、假設使用「TridentBackendConfig」建立下列後端：

```
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 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

從輸出中可以看出這一點 `TridentBackendConfig` 已成功建立並繫結至後端 [觀察後端的 UUID] 。

步驟1：確認 `deletionPolicy` 設為 `retain`

讓我們來看看的價值 `deletionPolicy`。這需要設為 `retain`。如此可確保刪除 CR 時 `TridentBackendConfig`、後端定義仍會存在、並可透過進行管理 `tridentctl`。

```
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
```

除非將「刪除原則」設定為「需要」、否則請勿繼續下一步。

步驟2：刪除 `TridentBackendConfig` CR

最後一個步驟是刪除「TridentBackendConfig」（TridentBackendConfig）。確認「刪除原則」設為「保留」之後、您可以繼續刪除：

```
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 |
+-------------------+----------------
+--------------------------------------+--------+---------+
```

刪除物件後 `TridentBackendConfig`、 Trident 只是將其移除、而不會實際刪除後端本身。

建立及管理儲存類別

建立儲存類別

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

設定 Kubernetes StorageClass 物件

會 "Kubernetes StorageClass 物件"將 Trident 識別為該類別所使用的資源配置程式、並指示 Trident 如何資源配置 Volume 。例如：

```yaml
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
```

如需儲存類別如何與互動的詳細資訊 `PersistentVolumeClaim`、以及控制 Trident 配置磁碟區的參數、請參閱"Kubernetes和Trident物件"。

建立儲存類別

建立 StorageClass 物件之後、即可建立儲存類別。 [儲存類別範例] 提供一些您可以使用或修改的基本範例。

步驟

1. 這是 Kubernetes 物件、請使用 `kubectl` 在Kubernetes中建立。

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

2. 現在您應該會在 Kubernetes 和 Trident 中同時看到 * base-csi* 儲存類別、而 Trident 應該已經在後端上探索到這些集區。

```
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、您也可以編輯 `sample-input/storage-class-csi.yaml.templ` 安裝程式隨附並取代的檔案 *`BACKEND_TYPE`* 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

您可以檢視現有的儲存類別、設定預設的儲存類別、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儲存類別詳細資料、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「持續磁碟區宣告」（PVc）中指定一個、則此儲存類別將用於配置「持續磁碟區」。

- 在儲存類別定義中、將「shorageclass.Kubernetes.IO/as-default-Class」註釋設為true、以定義預設儲存類別。根據規格、任何其他值或不存在附註都會解譯為假。
- 您可以使用下列命令、將現有的儲存類別設定為預設的儲存類別：

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

- 同樣地、您也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附註的範例。

> 叢集中一次只應有一個預設儲存類別。Kubernetes在技術上並不妨礙您擁有多個儲存類別、但它的行為方式就如同完全沒有預設的儲存類別一樣。

識別儲存類別的後端

這是您可以使用 JSON 來回答的問題類型範例、此問題 `tridentctl`會針對 Trident 後端物件輸出。這會使用 `jq` 您可能需要先安裝的公用程式。

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

刪除儲存類別

若要從Kubernetes刪除儲存類別、請執行下列命令：

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

「<storage－class>'」應改用您的儲存類別。

透過此儲存類別建立的任何持續磁碟區都將保持不變、 Trident 將繼續管理這些磁碟區。

> Trident 會對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 、然後使用指定的方式重新建立它們 `parameters.fsType`。

資源配置與管理磁碟區

配置 **Volume**

建立 PersistentVolume Claim （ PVC ），使用設定的 Kubernetes StorageClass 要求存取 PV 。然後、您可以將 PV 掛載至 Pod 。

總覽

```
https://kubernetes.io/docs/concepts/storage/persistent-
volumes["_PersistentVolume Claim"^]（ PVC ）是存取叢集上 PersistentVolume
的要求。
```

可將 PVC 設定為要求儲存特定大小或存取模式。叢集管理員可以使用相關的 StorageClass 來控制超過 PersistentVolume 大小和存取模式的權限、例如效能或服務層級。

建立 PVC 之後，您可以將磁碟區裝入 Pod 。

建立 **PVC**

步驟

1. 建立 PVC 。

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

2. 確認 PVC 狀態。

```
kubectl get pvc
```

```
NAME         STATUS  VOLUME     CAPACITY   ACCESS MODES  STORAGECLASS AGE
pvc-storage Bound   pv-name    1Gi        RWO                   5m
```

1. 將磁碟區裝入 Pod 。

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

> 您可以使用監控進度 `kubectl get pod --watch`。

2. 確認磁碟區已掛載到上 `/my/mount/path`。

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

3. 您現在可以刪除 Pod 。Pod 應用程式將不再存在、但該磁碟區仍會保留。

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

範例資訊清單

PersistentVolume Claim 範例資訊清單

這些範例顯示基本的 PVC 組態選項。

可存取 RWO 的 PVC

此範例顯示具有 rwo 存取權的基本 PVC 、與命名的 StorageClass 相關聯 `basic-csi`。

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

採用 NVMe / TCP 的 PVC

此範例顯示 NVMe / TCP 的基本 PVC 、並提供與命名 StorageClass 相關的 rwo 存取 `protection-gold`。

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

Pod 資訊清單範例

這些範例顯示將 PVC 連接至 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
```

如需儲存類別如何與互動的詳細資訊 `PersistentVolumeClaim`、以及控制 Trident 配置磁碟區的參數、請參閱"Kubernetes和Trident物件"。

展開Volume

Trident為 Kubernetes 使用者提供了在建立磁碟區後擴充磁碟區的能力。尋找有關擴展 iSCSI、NFS、SMB、NVMe/TCP 和 FC 磁碟區所需的配置的資訊。

展開iSCSI Volume

您可以使用「SCSI資源配置程式」來擴充iSCSI持續磁碟區（PV）。

> iSCSI磁碟區擴充支援「ontap－san」、「ONTAP－san經濟」、「Poolidfire－san」等驅動程式、需要Kubernetes 1.16及更新版本。

步驟1：設定StorageClass以支援Volume擴充

編輯 StorageClass 定義以設定 `allowVolumeExpansion` 欄位至 `true`。

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

```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、請編輯此類以包含「owalumVolume Expansion」參數。

步驟2：使用您建立的StorageClass建立一個永久虛擬儲存設備

編輯 PVC 定義並更新 `spec.resources.requests.storage` 以反映新的所需大小、此大小必須大於原始大小。

```
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 Volume （ PV ）、並將其與此持續 Volume Claim （ PVC ）相關聯。

```
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：定義一個連接至PVc的Pod

將 PV 附加至 Pod 、以便調整大小。調整iSCSI PV的大小有兩種情況：

- 如果 PV 附加至 Pod 、 Trident 會在儲存後端擴充磁碟區、重新掃描裝置、並調整檔案系統的大小。
- 當嘗試調整未附加 PV 的大小時、 Trident 會在儲存後端擴充磁碟區。在將永久虛擬磁碟綁定至Pod之後、Trident會重新掃描裝置並重新調整檔案系統的大小。然後、Kubernetes會在擴充作業成功完成後、更新PVc大小。

在此範例中、會建立使用「shan -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

若要調整從1Gi建立至2Gi的PV大小、請編輯PVc定義、並將「sec.resumes.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    |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
```

展開FC Volume

您可以使用 CSI 資源配置程式來擴充 FC 持續 Volume （PV）。

> 驅動程式支援 FC Volume 擴充 `ontap-san`，需要 Kubernetes 1.16 及更新版本。

步驟1：設定StorageClass以支援Volume擴充

編輯 StorageClass 定義以設定 `allowVolumeExpansion` 欄位至 `true`。

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

```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、請編輯此類以包含「owalumVolume Expansion」參數。

步驟2：使用您建立的StorageClass建立一個永久虛擬儲存設備

編輯 PVC 定義並更新 `spec.resources.requests.storage` 以反映新的所需大小、此大小必須大於原始大小。

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

```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 Volume （ PV ）、並將其與此持續 Volume Claim （ PVC ）相關聯。

```
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：定義一個連接至PVc的Pod

將 PV 附加至 Pod 、以便調整大小。調整 FC PV 的大小有兩種情況：

- 如果 PV 附加至 Pod 、 Trident 會在儲存後端擴充磁碟區、重新掃描裝置、並調整檔案系統的大小。
- 當嘗試調整未附加 PV 的大小時、 Trident 會在儲存後端擴充磁碟區。在將永久虛擬磁碟綁定至Pod之後、Trident會重新掃描裝置並重新調整檔案系統的大小。然後、Kubernetes會在擴充作業成功完成後、更新PVc大小。

在此範例中、會建立使用「shan -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

若要調整從1Gi建立至2Gi的PV大小、請編輯PVc定義、並將「sec.resumes.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    |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
```

展開NFS Volume

Trident支援為已設定的 NFS PV 擴充容量。 `ontap-nas` ， `ontap-nas-economy` ， `ontap-nas-flexgroup` ， 和 `azure-netapp-files`後端。

步驟1：設定StorageClass以支援Volume擴充

若要調整NFS PV的大小、管理員必須先將「ow淺Volume Expansion」欄位設定為「true」、以設定儲存類別以允許磁碟區擴充：

```
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
```

如果您已經建立了沒有此選項的儲存類別、只要使用「kubecl Edit storageclass'」來編輯現有的儲存類別、即可進行磁碟區擴充。

步驟**2**：使用您建立的**StorageClass**建立一個永久虛擬儲存設備

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

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

Trident應該為該 PVC 建立一個 20 MiB 的 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**

若要將新建立的 20 MiB PV 調整為 1 GiB，請編輯 PVC 並設定 `spec.resources.requests.storage`至 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、以驗證調整大小是否正常運作：

```
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。

總覽與考量

您可以將磁碟區匯入 Trident 、以便：

- 將應用程式容器化、並重新使用其現有的資料集
- 針對臨時應用程式使用資料集的複本
- 重建故障的 Kubernetes 叢集
- 在災難恢復期間移轉應用程式資料

考量

匯入 Volume 之前、請先檢閱下列考量事項。

- Trident 只能匯入 RW （讀寫）類型的 ONTAP Volume 。DP （資料保護）類型磁碟區是 SnapMirror 目的地磁碟區。您應該先中斷鏡射關係、再將磁碟區匯入 Trident 。

- 我們建議您在沒有作用中連線的情況下匯入磁碟區。若要匯入使用中的 Volume 、請複製該 Volume 、然後執行匯入。

> 這對區塊磁碟區特別重要、因為 Kubernetes 不會知道先前的連線、而且很容易將作用中的磁碟區附加到 Pod 。這可能導致資料毀損。

- 雖然必須在 PVC 上指定、但 `StorageClass` Trident 在匯入期間不會使用此參數。建立磁碟區時會使用儲存類別、根據儲存特性從可用的集區中選取。由於該磁碟區已經存在、因此在匯入期間不需要選取任何集區。因此、即使磁碟區存在於與 PVC 中指定的儲存類別不相符的後端或集區、匯入也不會失敗。
- 現有的 Volume 大小是在 PVC 中決定和設定的。儲存驅動程式匯入磁碟區之後、PV會以PVc的ClaimRef建立。
  - 回收原則一開始設定為 `retain` 在PV中。Kubernetes成功繫結了PVc和PV之後、系統會更新回收原則以符合儲存類別的回收原則。
  - 如果儲存類別的回收原則為 `delete`、儲存磁碟區會在PV刪除時刪除。
- 預設情況下， Trident管理 PVC，並在後端重新命名FlexVol volume和 LUN。你可以透過 `--no-manage`導入非託管磁碟區的標誌和 `--no-rename`標記以保留卷名。
  - `--no-manage`* - 如果您使用 `--no-manage`標誌表明， Trident在物件的生命週期內不會對 PVC 或 PV 執行任何額外的操作。刪除 PV 時，儲存磁碟區不會被刪除，其他操作（如磁碟區複製和磁碟區調整大小）也會被忽略。
  - `--no-rename`* - 如果您使用 `--no-rename`標誌， `Trident`在導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。此選項僅支援以下情況： `ontap-nas` ， `ontap-san` （含ASA r2 系統）`ontap-san-economy`司機。

> 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。

- 將註釋新增至PVc和PV、這有兩種用途、表示已匯入磁碟區、以及是否管理了PVc和PV。不應修改或移除此附註。

匯入 **Volume**

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`。

   以下是最低規格的範例：

```
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`用於指定包含磁碟區的Trident後端名稱以及唯一標識儲存上磁碟區的名稱的命令（例如： ONTAP FlexVol、Element Volume）。這 `-f`需要提供參數來指定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

請參閱下列 Volume 匯入範例、瞭解支援的驅動程式。

ONTAP NAS 和 ONTAP NAS FlexGroup

Trident 支援使用和 `ontap-nas-flexgroup`驅動程式進行 `Volume` 匯入 `ontap-nas`。

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

使用驅動程式建立的每個磁碟區都 `ontap-nas`是 `ONTAP` 叢集上的 `FlexVol volume` 。使用驅動程式匯入 `FlexVol` 磁碟區 `ontap-nas`的運作方式相同。`ONTAP` 叢集上已存在的 `FlexVol Volume` 可匯入為 `ontap-nas` PVC 。同樣地、 FlexGroup Vols 也可以匯入為 `ontap-nas-flexgroup` PVCS 。

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 Volume

以下範例會匯入名為的 Volume `managed_volume` 在名為的後端上 `ontap_nas`：

```
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    |
+------------------------------------------+---------+---------------
+----------+--------------------------------------+--------+---------+
```

非託管 Volume

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

以下範例匯入 `unmanaged_volume` 在上 `ontap_nas` 後端：

```
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 Volume

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>
```

非託管 Volume

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>
```

SAN ONTAP

Trident支援使用卷宗導入 `ontap-san` （iSCSI、NVMe/TCP 和 FC）和 `ontap-san-economy` 司機。

Trident可以匯入包含單一 LUN 的ONTAP SAN FlexVol磁碟區。這與 `ontap-san` 驅動程序，它為每個 PVC 建立一個FlexVol volume，並在FlexVol volume內建立一個 LUN。Trident導入FlexVol volume並將其與 PVC 定義關聯。Trident可以導入 `ontap-san-economy` 包含多個 LUN 的磁碟區。

以下範例展示如何匯入託管和非託管磁碟區：

託管 Volume

對於託管卷， Trident 將 FlexVol volume 重命名爲格式，並將 FlexVol volume 中的 LUN `lun0` 重命名爲 `pvc-<uuid>`。

下列範例會匯入 `ontap-san-managed`**後端上的** `FlexVol volume` `ontap_san_default`：

```
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    |
+------------------------------------------+--------+---------------
+----------+--------------------------------------+--------+---------+
```

非託管 Volume

以下範例匯入 `unmanaged_example_volume` 在上 `ontap_san` 後端：

```
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 對應至與 Kubernetes 節點 IQN 共用 IQN 的 igroup 、如下列範例所示、您將會收到錯誤訊息：`LUN already mapped to initiator(s) in this group`。您需要移除啟動器或取消對應 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 支援使用驅動程式的 NetApp Element 軟體和 NetApp HCI Volume 匯入 `solidfire-san`。

> Element驅動程式支援重複的Volume名稱。不過、如果有重複的磁碟區名稱、 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。

下列範例會匯入 `element-managed` 後端上的 Volume `element_default`。

```
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 支援使用驅動程式進行 Volume 匯入 `azure-netapp-files`。

> 若要匯入 Azure NetApp Files Volume 、請依磁碟區路徑識別該磁碟區。Volume 路徑是之後 Volume 匯出路徑的一部分 `:/`。例如、如果掛載路徑為 `10.0.0.2:/importvol1`、磁碟區路徑為 `importvol1`。

下列範例會匯入 `azure-netapp-files` 後端上的 Volume `azurenetappfiles_40517` 磁碟區路徑 `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 支援使用驅動程式進行 Volume 匯入 `google-cloud-netapp-volumes`。

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
|
+------------------------------------------+---------
+----------------------+----------+--------------------------------------
+--------+---------+
```

下列範例會在兩個磁碟區位於同一個區域時匯入 `google-cloud-netapp-volumes` Volume ：

```
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

有了 Trident 、您就可以為您建立的磁碟區指派有意義的名稱和標籤。這有助於您識別並輕鬆地將磁碟區對應至各自的 Kubernetes 資源（ PVCS ）。您也可以在後端層級定義範本、以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守這些範本。

開始之前

可自訂的 Volume 名稱和標籤支援：

- Volume 建立、匯入及複製作業。
- 就此而言 `ontap-nas-economy` 驅動程式中，只有 Qtree 磁碟區的名稱符合名稱範本。
- 就此而言 `ontap-san-economy` 驅動程式中，只有 LUN 名稱符合名稱範本。

限制

- 自訂磁碟區名稱僅與ONTAP本機驅動程式相容。
- 僅對以下情況支援自訂標籤： `ontap-san`， `ontap-nas`， 和 `ontap-nas-flexgroup` 司機。
- 自訂磁碟區名稱不適用於現有磁碟區。

可自訂 **Volume** 名稱的主要行為

- 如果名稱範本中的語法無效而導致失敗、則後端建立會失敗。但是、如果範本應用程式失敗、則會根據現有的命名慣例來命名磁碟區。

- 如果使用後端組態的名稱範本命名磁碟區、則不適用儲存前置詞。任何所需的前置字元值都可以直接新增至範本。

名稱範本和標籤的後端組態範例

自訂名稱範本可在根和 / 或集區層級定義。

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}}"
  }
}
```

集區層級範例

```
{
  "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 }}""
```

需要考量的重點

1. 在 Volume 匯入的情況下、只有現有的 Volume 具有特定格式的標籤時、標籤才會更新。例如 `{"provisioning":{"Cluster":"ClusterA", "PVC": "pvcname"}}`：。
2. 在託管 Volume 匯入的情況下、 Volume 名稱會遵循在後端定義的根層級所定義的名稱範本。
3. Trident 不支援使用含有儲存前置碼的 Slice 運算子。
4. 如果範本未產生唯一的磁碟區名稱、 Trident 會附加幾個隨機字元、以建立唯一的磁碟區名稱。
5. 如果 NAS 經濟 Volume 的自訂名稱長度超過 64 個字元、 Trident 會根據現有的命名慣例來命名磁碟區。對於所有其他 ONTAP 驅動程式、如果磁碟區名稱超過名稱限制、磁碟區建立程序就會失敗。

跨命名空間共用**NFS**磁碟區

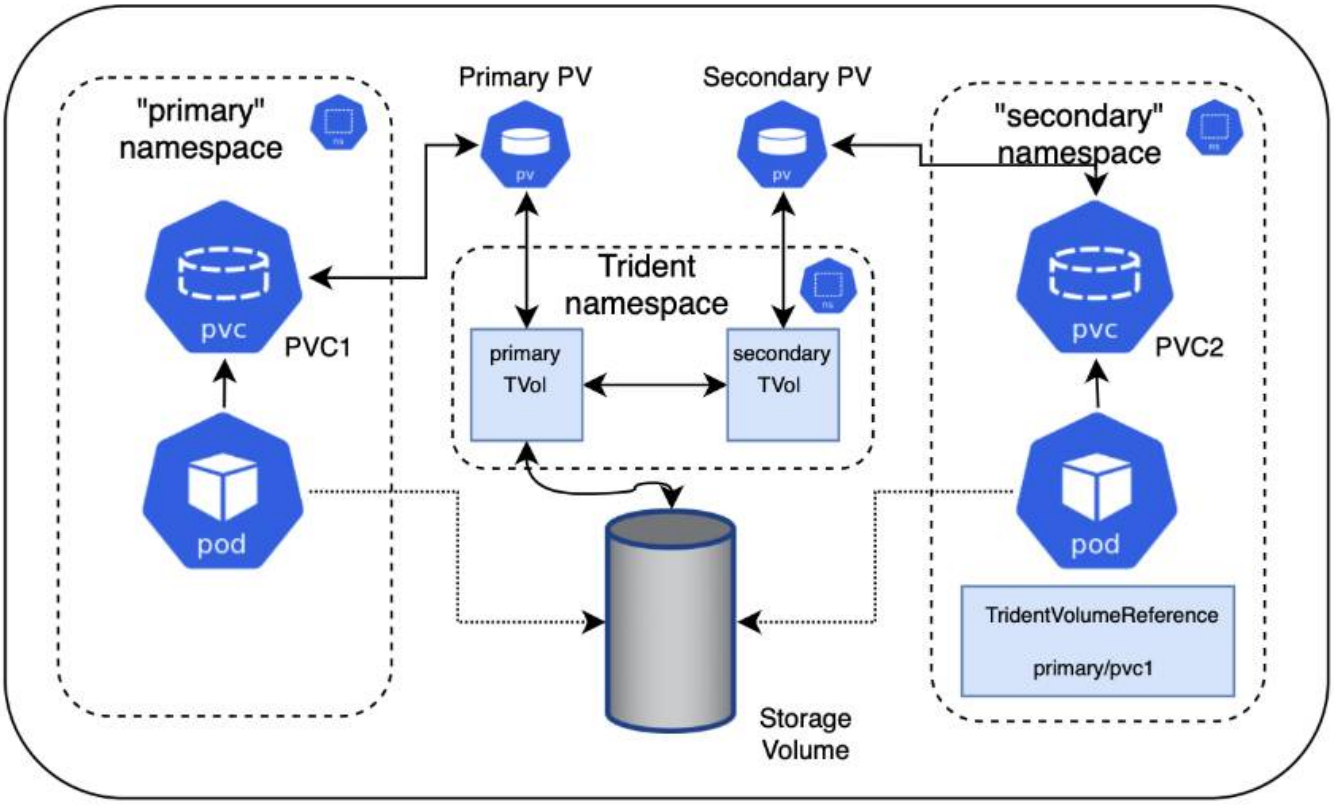
使用 Trident 、您可以在主要命名空間中建立磁碟區、並在一或多個次要命名空間中共用該磁碟區。

功能

TridentVolume Reference CR 可讓您安全地跨一或多個 Kubernetes 命名空間共用 ReadWriteMany （ rwx ） NFS 磁碟區。此Kubernetes原生解決方案具有下列優點：

- 多層存取控制、確保安全性
- 可搭配所有Trident NFS Volume驅動程式使用
- 不依賴tridentctl或任何其他非原生Kubernetes功能

此圖說明兩個Kubernetes命名空間之間的NFS Volume共用。



快速入門

您只需幾個步驟就能設定NFS Volume共享。

1 設定來源**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 授予在目的地命名空間中建立**CR**的權限

叢集管理員授予目的地命名空間擁有者建立TridentVolume Reference CR的權限。

3 在目的地命名空間中建立**TridentVolume Reference**

目的地命名空間的擁有者會建立TridentVolume Reference CR來參照來源PVC。

4 在目的地命名空間中建立從屬的**PVc**

目的地命名空間的擁有者會建立從屬的PVc、以使用來源PVc的資料來源。

設定來源和目的地命名空間

為了確保安全性、跨命名空間共用需要來源命名空間擁有者、叢集管理員和目的地命名空間擁有者的協同作業與行動。使用者角色會在每個步驟中指定。

步驟

1. *來源命名空間擁有者：*建立PVC (`pvc1`) (`namespace2`) 使用 `shareToNamespace` 註釋：

```
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。

> ◦ 您可以使用以逗號分隔的清單、將永久虛擬儲存設備共用至多個命名空間。例如、`trident.netapp.io/shareToNamespace: namespace2,namespace3,namespace4`。
> ◦ 您可以使用共用至所有命名空間 `*`。例如、`trident.netapp.io/shareToNamespace: *`
> ◦ 您可以更新PVC,以納入 `shareToNamespace` 隨時註釋。

2. *叢集管理員：*確保已建立適當的 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。
3. *目的地命名空間擁有者：*在參照來源命名空間的目的地命名空間中建立TridentVolume Reference CR `pvc1`。

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

4. *目的地命名空間擁有者：*建立一個PVC (`pvc2`) (`namespace2`) 使用 `shareFromPVC` 註釋以指定來源PVc。

```yaml
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 會讀取 `shareFromPVC`目的地 PVC 上的附註、並將目的地 PV 建立為次級磁碟區、而不會有本身的儲存資源指向來源 PV 、並共用來源 PV 儲存資源。目的地的PVc和PV似乎正常連結。

刪除共享Volume

您可以刪除跨多個命名空間共用的磁碟區。Trident 將移除來源命名空間上的磁碟區存取權、並維護共用該磁碟區的其他命名空間的存取權。移除所有參照該 Volume 的命名空間時、 Trident 會刪除該 Volume 。

使用 `tridentctl get` 查詢從屬Volume

使用[`tridentctl` 公用程式、您可以執行 `get` 取得從屬磁碟區的命令。如需詳細資訊、請參閱連結：../ Trident 參考/ tridentctl.html[`tridentctl` 命令與選項]。

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

旗標：

- `-h, --help`：Volume的說明。
- `--parentOfSubordinate string`：將查詢限制在從屬來源Volume。
- `--subordinateOf string`：將查詢限制在Volume的下屬。

限制

- Trident 無法防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Volume資料。

- 您無法藉由移除來撤銷對來源PVc的存取權 `shareToNamespace` 或 `shareFromNamespace` 註釋或刪除 `TridentVolumeReference` CR.若要撤銷存取權、您必須刪除從屬的PVc。
- 在從屬磁碟區上無法執行快照、複製和鏡射。

以取得更多資訊

若要深入瞭解跨命名空間Volume存取：

- 請造訪 "在命名空間之間共用磁碟區：歡迎使用跨命名空間磁碟區存取"。
- 觀看上的示範 "NetAppTV"。

跨命名空間複製磁碟區

使用 Trident ，您可以使用同一個 Kubernetes 叢集中不同命名空間中的現有磁碟區或磁碟區快照來建立新的磁碟區。

先決條件

在複製磁碟區之前，請確定來源和目的地後端的類型相同，而且具有相同的儲存類別。

> 跨命名空間克隆僅支援 `ontap-san`和 `ontap-nas`儲存驅動程式。不支援只讀克隆。

快速入門

只需幾個步驟即可設定磁碟區複製。

1. 設定來源 **PVC** 來複製磁碟區

來源命名空間擁有者授予存取來源PVc中資料的權限。

2. 授予在目的地命名空間中建立**CR**的權限

叢集管理員授予目的地命名空間擁有者建立TridentVolume Reference CR的權限。

3. 在目的地命名空間中建立**TridentVolume Reference**

目的地命名空間的擁有者會建立TridentVolume Reference CR來參照來源PVC。

4. 在目的地命名空間中建立複製 **PVC**

目的地命名空間的擁有者會建立 PVC ，從來源命名空間複製 PVC 。

設定來源和目的地命名空間

為了確保安全性，跨命名空間複製磁碟區需要來源命名空間擁有者，叢集管理員和目的地命名空間擁有者共同作業和採取行動。使用者角色會在每個步驟中指定。

步驟

1. * 來源命名空間擁有者：* (`pvc1`在來源命名空間中建立 `PVC` (`namespace1`)，可授予與目的地命名空間共用的權限(`namespace2` `cloneToNamespace`。

```
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 會建立 PV 及其後端儲存磁碟區。

> ◦ 您可以使用以逗號分隔的清單、將永久虛擬儲存設備共用至多個命名空間。例如 `trident.netapp.io/cloneToNamespace: namespace2,namespace3,namespace4`：。
> ◦ 您可以使用共用所有命名空間 `*`。例如、 `trident.netapp.io/cloneToNamespace: *`
> ◦ 您可以隨時更新 PVC 以納入 `cloneToNamespace`附註。

2. 叢集管理員：確保已建立適當的 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. *目的地命名空間擁有者：*在參照來源命名空間的目的地命名空間中建立TridentVolume Reference CR `pvc1`。

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

4. * 目的地命名空間擁有者：* (`pvc2`在目的地命名空間中建立 `PVC` (`namespace2`) 使用

`cloneFromPVC`或 `cloneFromSnapshot`和 `cloneFromNamespace`註釋來指定來源 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 經濟型驅動程式配置的 PVC ，不支援唯讀複本。

使用 **SnapMirror** 複寫磁碟區

Trident 支援一個叢集上的來源磁碟區與對等叢集上的目的地磁碟區之間的鏡射關係，以便複寫資料以進行災難恢復。 您可以使用名為 Trident 鏡像關係 (TMR) 的命名空間自訂資源定義 (CRD) 來執行下列操作：

- 建立磁碟區之間的鏡射關係（ PVCS ）
- 移除磁碟區之間的鏡射關係
- 中斷鏡射關係
- 在災難情況（容錯移轉）期間提升次要 Volume
- 在計畫性容錯移轉或移轉期間、將應用程式從叢集無損移轉至叢集

複寫先決條件

在您開始之前、請確定符合下列先決條件：

叢集 **ONTAP**

- * Trident * ： Trident 22.10 版或更新版本必須同時存在於使用 ONTAP 作為後端的來源叢集和目的地 Kubernetes 叢集上。
- * 授權 * ：使用資料保護套件的 ONTAP SnapMirror 非同步授權必須同時在來源和目的地 ONTAP 叢集上啟用。如需詳細資訊、請參閱 "SnapMirror授權概述ONTAP" 。

從 ONTAP 9.10.1 開始、所有授權都會以 NetApp 授權檔案（NLF）的形式交付、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 隨附授權" 。

> 僅支援 SnapMirror 非同步保護。

對等關係

- * 叢集與 SVM* ：必須對 ONTAP 儲存設備的後端進行對等處理。如需詳細資訊、請參閱 "叢集與SVM對等概觀" 。

> 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* ：對等的遠端 SVM 必須可用於目的地叢集上的 Trident 。

支援的驅動程式

NetApp Trident 支援使用 NetApp SnapMirror 技術進行磁碟區複製，使用由下列驅動程式支援的儲存類別：**`ontap-nas`** ：**NFS** `ontap-san` ：iSCSI **`ontap-san`** ： **FC** `ontap-san` ：NVMe/TCP（要求最低 ONTAP 版本 9.15.1）

> ASA r2 系統不支援使用 SnapMirror 進行磁碟區複製。有關 ASA r2 系統的信息，請參閱"瞭解 ASA R2 儲存系統" 。

建立鏡射 **PVC**

請遵循下列步驟、並使用 CRD 範例在主要和次要磁碟區之間建立鏡射關係。

步驟

1. 在主 Kubernetes 叢集上執行下列步驟：
   a. 使用參數建立 StorageClass 物件 `trident.netapp.io/replication: true` 。

      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. 使用本機資訊建立 MirrorRelationship CR 。

範例

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

Trident 會擷取磁碟區的內部資訊和磁碟區目前的資料保護（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 Kubernetes 叢集上執行下列步驟：
   a. 使用 trident.netapp.io/replication: true 參數建立 StorageClass 。

      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. 使用目的地和來源資訊建立 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. 使用先前建立的 StorageClass 建立 PVC 、作為次要（ SnapMirror 目的地）。

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 會檢查 TridentMirrorRelationship CRD ，如果關係不存在，則無法建立 Volume 。如果存在這種關係， Trident 將確保新的 FlexVol volume 放置在與 MirrorRelationship 中定義的遠端 SVM 對等的 SVM 上。

Volume 複寫狀態

Trident Mirror Relationship （ TMR ）是一種 CRD 、代表 PVC 之間複寫關係的一端。目的地 TMR 具有狀態，可告知 Trident 所需的狀態。目的地 TMR 有下列狀態：

- * 建立 * ：本機 PVC 是鏡射關係的目的地 Volume 、這是新的關係。
- * 升級 * ：本機 PVC 為可讀寫且可掛載、目前無鏡射關係。
- * 重新建立 * ：本機 PVC 是鏡射關係的目的地 Volume 、先前也屬於該鏡射關係。
  - 如果目的地磁碟區與來源磁碟區有任何關係、則必須使用重新建立的狀態、因為它會覆寫目的地磁碟區內容。
  - 如果磁碟區先前未與來源建立關係、則重新建立的狀態將會失敗。

在非計畫性容錯移轉期間升級次要 **PVC**

在次 Kubernetes 叢集上執行下列步驟：

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

在規劃的容錯移轉期間升級次要 **PVC**

在計畫性容錯移轉（移轉）期間、請執行下列步驟來升級次要 PVC ：

步驟

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

   範例

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

3. 在次要 Kubernetes 叢集上、將 *TridentMirrorRelationship _ CR* 的 *_spec.state* 欄位更新為 *updated* 、 *spec.promotedSnapshotHandle* 更新為快照的內部名稱。
4. 在次要 Kubernetes 叢集上、確認要升級的 TridentMirrorRelationship 狀態（ STATUS.STATUS 欄位）。

在容錯移轉後還原鏡射關係

還原鏡射關係之前、請先選擇要設為新主要的一面。

步驟

1. 在次要 Kubernetes 叢集上、確保已更新 TridentMirrorRelationship 上 *spec.remoteVolumeHandle* 欄位的值。
2. 在次 Kubernetes 叢集上、將 TridentMirrorRelationship 的 _spec.mirror 欄位更新為 `reestablished`。

其他作業

Trident 支援在主要和次要磁碟區上執行下列作業：

將主要 **PVC** 複製到新的次要 **PVC**

請確定您已擁有主要 PVC 和次要 PVC 。

步驟

1. 從已建立的次要（目的地）叢集刪除 PersistentVolume Claim 和 TridentMirrorRelationship CRD 。
2. 從主（來源）叢集刪除 TridentMirrorRelationship CRD 。
3. 在主要（來源）叢集上建立新的 TridentMirrorRelationship CRD 、以用於您要建立的新次要（目的地）PVC 。

調整鏡射、主要或次要 **PVC** 的大小

PVC 可以正常調整大小、如果資料量超過目前大小、 ONTAP 會自動擴充任何目的地 flevxols 。

從 **PVC** 移除複寫

若要移除複寫、請在目前的次要磁碟區上執行下列其中一項作業：

- 刪除次要 PVC 上的 MirrorRelationship 。這會中斷複寫關係。
- 或者、將 spec.state 欄位更新為 *updated* 。

刪除 **PVC** （先前已鏡射）

Trident 會檢查複寫的 PVCS ，並在嘗試刪除磁碟區之前先釋放複寫關係。

刪除 **TMR**

刪除鏡射關係一側的 TMR 會導致其餘 TMR 在 Trident 完成刪除之前轉換至 升 遷狀態。如果選取要刪除的 *TMR* 已處於 _*Promive* 狀態，則沒有現有的鏡射關係， TMR 將會移除，而 Trident 會將本機 PVC 升級為 *ReadWrite* 。此刪除作業會在 ONTAP 中針對本機磁碟區釋出 SnapMirror 中繼資料。如果此磁碟區在未來的鏡射關係中使用、則在建立新的鏡射關係時、它必須使用具有 _ 建立 _ 磁碟區複寫狀態的新 TMR 。

當 **ONTAP** 連線時、請更新鏡射關係

建立鏡射關係之後、可以隨時更新它們。您可以使用 `state: promoted` 或 `state: reestablished` 欄位來更新關聯。將目的地 Volume 升級為一般 ReadWrite Volume 時、您可以使用 *promotedSnapshotHandle* 來指定特定快照、將目前的 Volume 還原至。

當 **ONTAP** 離線時更新鏡射關係

您可以使用 CRD 來執行 SnapMirror 更新，而無需 Trident 直接連線至 ONTAP 叢集。請參閱下列 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。它可以取自 *sued* 、 *in progress* 或 *Failed* 的值。

使用「**csi**拓撲」

Trident 可以利用來選擇性地建立磁碟區、並將其附加至 Kubernetes 叢集中的節點 "「csi 拓撲」功能"。

總覽

使用「csi拓撲」功能、可根據區域和可用性區域、限制對磁碟區的存取、只能存取一部分節點。如今、雲端供應商可讓Kubernetes管理員建立以區域為基礎的節點。節點可位於某個區域內的不同可用度區域、或位於不同區域之間。為了協助在多區域架構中為工作負載配置磁碟區、 Trident 使用 CSI 拓撲。

深入瞭解「csi拓撲」功能 "請按這裡"。

Kubernetes提供兩種獨特的Volume繫結模式：

- 如果 `VolumeBindingMode`**設置爲** `Immediate`，則 Trident 會在沒有任何拓撲感知的情況下創建卷。建立永久虛擬磁碟時、即會處理磁碟區繫結和動態資源配置。這是預設值 `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需依賴要求的 Pod 排程需求。
- 將「Volume BindingMode」設為「WaitForFirst消費者」時、會延遲建立和繫結永久磁碟區、直到排程並建立使用永久磁碟的Pod為止。如此一來、就能建立磁碟區、以符合拓撲需求所強制執行的排程限制。

「等待使用者」繫結模式不需要拓撲標籤。這可獨立於「csi拓撲」功能使用。

您需要的產品

若要使用「csi拓撲」、您需要下列項目：

- 執行的Kubernetes叢集 "支援的Kubernetes版本"

```
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"}
```

- 叢集中的節點應具有標籤、以引入拓撲感知(`topology.kubernetes.io/region`和 `topology.kubernetes.io/zone`）。在安裝 Trident 之前，這些標籤 * 應該存在於叢集 * 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**：建立可感知拓撲的後端

Trident 儲存設備後端可根據可用性區域、選擇性地配置磁碟區。每個後端都可以帶有一個可選的 `supportedTopologies` 區塊、代表支援的區域和區域清單。對於使用此類後端的StorageClass、只有在受支援地區/區域中排程的應用程式要求時、才會建立Volume。

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`用於提供每個後端的區域和區域清單。這些區域和區域代表StorageClass中可提供的允許值清單。對於包含後端所提供區域和區域子集的 StorageClasses 、 Trident 會在後端建立磁碟區。

您也可以定義每個儲存資源池的「支援拓撲」。請參閱下列範例：

```yaml
---
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」決定儲存資源池的使用來源。

步驟**2**：定義可感知拓撲的**StorageClass**

根據提供給叢集中節點的拓撲標籤、可以定義StorageClass以包含拓撲資訊。這將決定做為所提出之永久虛擬磁碟要求候選的儲存資源池、以及可以使用Trident所提供之磁碟區的節點子集。

請參閱下列範例：

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

在上面提供的 StorageClass 定義中 `volumeBindingMode`，設置爲 `WaitForFirstConsumer`。在Pod中引用此StorageClass所要求的PVCS之前、系統不會對其採取行動。此外、還 `allowedTopologies`提供要使用的區域和區域。`StorageClass` 會 `netapp-san-us-east1`在上述定義的後端建立 `PVC` `san-backend-us-east1` 。

步驟**3**：建立並使用**PVC**

建立StorageClass並對應至後端後端後端之後、您現在就可以建立PVCS。

請參閱以下「sPEC」範例：

```yaml
---
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永久虛擬環境可能會產生下列結果：

```
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Trident建立磁碟區並將其連結至PVc、請在Pod中使用PVc。請參閱下列範例：

```yaml
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
```

此pod化 規範會指示Kubernetes在「us-east1」區域的節點上排程pod、並從「us-east1-a」或「us-east1-b」區域中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
```

更新後端以納入 `supportedTopologies`

您可以使用「tridentctl後端更新」來更新現有的後端、以納入「最上層拓撲」清單。這不會影響已配置的磁碟區、而且只會用於後續的PVCS。

如需詳細資訊、請參閱

- "管理容器的資源"
- "節點選取器"
- "關聯性與反關聯性"
- "污染與容許"

使用快照

Kubernetes 持續磁碟區（PV）的磁碟區快照可啟用磁碟區的時間點複本。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、從現有快照建立新的磁碟區、以及從快照復原磁碟區資料。

總覽

卷快照受支援 `ontap-nas`、`ontap-nas-flexgroup`、`ontap-san`、`ontap-san-economy`、`solidfire-san`、`azure-netapp-files`、和 `google-cloud-netapp-volumes`司機。

開始之前

您必須擁有外部快照控制器和自訂資源定義（CRD）、才能使用快照。這是Kubernetes Orchestrator的責任（例如：Kubeadm、GKE、OpenShift）。

如果您的Kubernetes發佈版本未包含快照控制器和CRD、請參閱 部署 Volume Snapshot 控制器。

> 如果在 GKE 環境中建立隨需磁碟區快照、請勿建立快照控制器。GKE使用內建的隱藏式快照控制器。

建立磁碟區快照

步驟

1. 建立 `VolumeSnapshotClass`。如需詳細資訊、請參閱 "Volume SnapshotClass"。
   - `driver`指向 Trident CSI 驅動程式。
   - `deletionPolicy` 可以 `Delete` 或 `Retain`。設定為時 `Retain`、儲存叢集上的基礎實體快照、即使在 `VolumeSnapshot` 物件已刪除。

     範例

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

     ```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
     ```

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

   - 此範例會為名稱為 PVC 的 Volume Snapshot 物件建立一個 `pvc1` 快照名稱設為 `pvc1-snap`。Volume Snapshot類似於PVC、並與相關聯 `VolumeSnapshotContent` 代表實際快照的物件。

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

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

- 您可以識別 `VolumeSnapshotContent` 的物件 `pvc1-snap` 描述Volume Snapshot。。 `Snapshot Content Name` 識別提供此快照的Volume SnapshotContent物件。。 `Ready To Use` 參數表示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` 使用名為的 Volume Snapshot 建立 PVC `<pvc-name>` 做為資料來源。建立好永久虛擬基礎架構之後、就能將它附加到Pod上、就像使用任何其他永久虛擬基礎架構一樣使用。

> 將在來源 Volume 所在的同一個後端建立 PVC 。請參閱 "KB ：無法在替代後端建立 Trident PVC Snapshot 的 PVC"。

以下範例使用建立 PVC `pvc1-snap` 做為資料來源。

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

```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 **Volume** 快照

Trident 支援、"Kubernetes 預先配置的快照程序"可讓叢集管理員建立 `VolumeSnapshotContent`物件、並匯入在 Trident 之外建立的快照。

開始之前

Trident 必須已建立或匯入快照的父磁碟區。

步驟

1. * 叢集管理： * 建立 `VolumeSnapshotContent`參照後端快照的物件。這會在 Trident 中啟動快照工作流程。
   - 在中指定後端快照的名稱 `annotations` 做為 `trident.netapp.io/internalSnapshotName: <"backend-snapshot-name">`。
   - 請在中 `snapshotHandle`指定 `<name-of-parent-volume-in-trident>/<volume-snapshot-content-name>`。這是通話中外部快照機提供給 Trident 的唯一資訊 `ListSnapshots`。

     > ℹ 。 `<volumeSnapshotContentName>` 由於 CR 命名限制、無法永遠符合後端快照名稱。

     範例

     下列範例建立 `VolumeSnapshotContent` 參照後端快照的物件 `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. * 叢集管理： * 建立 `VolumeSnapshot` 參照的 CR `VolumeSnapshotContent` 物件：這會要求存取權以使用 `VolumeSnapshot` 在指定的命名空間中。

   範例

   下列範例建立 `VolumeSnapshot` CR 命名 `import-snap` 這是參考的 `VolumeSnapshotContent` 已命名 `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（不需採取任何行動）： * 外部快照機可辨識新建立的 `VolumeSnapshotContent`、並執行 `ListSnapshots`通話。`Trident` 會建立 `TridentSnapshot`。
   - 外部快照器會設定 `VolumeSnapshotContent` 至 `readyToUse` 和 `VolumeSnapshot` 至 `true`。
   - Trident 退貨 `readyToUse=true`。
4. * 任何使用者： * 建立 `PersistentVolumeClaim` 以參考新的 `VolumeSnapshot`、其中 `spec.dataSource` （或 `spec.dataSourceRef`）名稱為 `VolumeSnapshot` 名稱。

   範例

下列範例建立一個 PVC 參照 `VolumeSnapshot` 已命名 `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 **Volume** 資料

快照目錄預設為隱藏、以協助使用進行資源配置的磁碟區達到最大相容性 `ontap-nas` 和 `ontap-nas-economy` 驅動程式：啟用 `.snapshot` 直接從快照恢復資料的目錄。

使用 Volume Snapshot Restore ONTAP CLI 將磁碟區還原至先前快照中記錄的狀態。

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

> 當您還原快照複本時、會覆寫現有的 Volume 組態。建立快照複本之後對 Volume 資料所做的變更將會遺失。

從快照進行原位磁碟區還原

Trident 使用（TASR） CR 從快照提供快速的原位磁碟區還原 `TridentActionSnapshotRestore`。此 CR 是 Kubernetes 的必要行動、在作業完成後不會持續存在。

Trident支援快照恢復 `ontap-san`，`ontap-san-economy`，`ontap-nas`，`ontap-nas-flexgroup`，`azure-netapp-files`，`google-cloud-netapp-volumes`，和 `solidfire-san`司機。

開始之前

您必須擁有受約束的 PVC 和可用的 Volume 快照。

- 確認 PVC 狀態為「已連結」。

```
kubectl get pvc
```

- 驗證 Volume 快照是否已準備就緒可供使用。

```
kubectl get vs
```

步驟

1. 建立 TASR CR 。本示例爲 PVC 和 Volume Snapshot 創建 CR `pvc1` `pvc1-snapshot`。

   TASR CR 必須位於 PVC 與 VS 所在的命名空間中。

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

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

2. 套用 CR 以從快照還原。此示例從 Snapshot 恢復 `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 不會在發生故障時自動重試作業。您需要再次執行此作業。
- 不具備管理員存取權限的 Kubernetes 使用者可能必須獲得管理員的權限、才能在其應用程式命名空間中建立 TASR CR 。

刪除含有相關快照的 PV

刪除含相關快照的持續 Volume 時，對應的 Trident Volume 會更新為「刪除狀態」。移除磁碟區快照以刪除 Trident 磁碟區。

部署 Volume Snapshot 控制器

如果您的Kubernetes發佈版本未包含快照控制器和客戶需求日、您可以依照下列方式進行部署。

步驟

1. 建立Volume Snapshot客戶需求日。

```
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控制器。

```
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、請開啟 `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` 和更新 `namespace` 到您的命名空間。

相關連結

- "Volume快照"
- "Volume SnapshotClass"

使用磁碟區組快照

Kubernetes 持久性磁碟區 (PV) 磁碟區組快照 NetApp Trident 提供了建立多個磁碟區（一組磁碟區快照）的功能。此磁碟區組快照代表在同一時間點建立的多個磁碟區的副本。

> VolumeGroupSnapshot 是 Kubernetes 中的一個 Beta 功能，包含 Beta 版 API。VolumeGroupSnapshot所需的最低版本為 Kubernetes 1.32。

建立卷宗組快照

以下儲存驅動程式支援磁碟區組快照：

- `ontap-san`驅動程式 - 僅適用於 iSCSI 和 FC 協議，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 儲存系統不支援磁碟區組快照。

開始之前

- 確保您的 Kubernetes 版本是 K8s 1.32 或更高版本。
- 您必須擁有外部快照控制器和自訂資源定義（CRD）、才能使用快照。這是Kubernetes Orchestrator的責任（例如：Kubeadm、GKE、OpenShift）。

  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 控制器 。

  > 如果在 GKE 環境中建立按需磁碟區組快照，請不要建立快照控制器。GKE使用內建的隱藏式快照控制器。

- 在快照控制器 YAML 中，設定 `CSIVolumeGroupSnapshot`功能門控設定為“true”，以確保磁碟區組快照已啟用。
- 在建立磁碟區組快照之前，建立所需的磁碟區組快照類別。
- 確保所有 PVC/磁碟區都在同一個 SVM 上，以便能夠建立 VolumeGroupSnapshot。

步驟

- 在創建 VolumeGroupSnapshot 之前建立一個 VolumeGroupSnapshotClass。如需詳細資訊、請 "卷冊組快照類"參閱。

```
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 PVC。

  以下範例使用以下方式建立 PVC `pvc1-group-snap`作為資料來源和標籤 `consistentGroupSnapshot: groupA`.根據您的要求定義標籤鍵和值。

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc1-group-snap
  labels:
    consistentGroupSnapshot: groupA
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Mi
  storageClassName: sc1-1
```

- 建立具有相同標籤的 VolumeGroupSnapshot (`consistentGroupSnapshot: groupA`) 在 PVC 中指定。

  此範例建立磁碟區組快照：

```
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數據

您可以使用作為磁碟區組快照的一部分所建立的各個快照來還原各個持久性磁碟區。您無法將磁碟區組快照作為一個整體進行還原。

使用 Volume Snapshot Restore ONTAP CLI 將磁碟區還原至先前快照中記錄的狀態。

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

> 當您還原快照複本時、會覆寫現有的 Volume 組態。建立快照複本之後對 Volume 資料所做的變更將會遺失。

從快照進行原位磁碟區還原

Trident 使用（TASR） CR 從快照提供快速的原位磁碟區還原 `TridentActionSnapshotRestore`。此 CR 是 Kubernetes 的必要行動、在作業完成後不會持續存在。

如需更多資訊、請參閱 "從快照進行原位磁碟區還原"。

刪除與群組快照關聯的 **PV**

刪除群組磁碟區快照時：

- 您可以刪除整個 VolumeGroupSnapshots，而不是刪除群組中的單一快照。
- 如果在持久卷存在快照的情況下刪除了該持久卷，Trident 會將該卷移至「刪除」狀態，因為必須先刪除快照，然後才能安全刪除該卷。
- 如果已使用分組快照建立了克隆，然後要刪除該群組，則會開始克隆拆分操作，並且在拆分完成之前無法刪除該群組。

部署 **Volume Snapshot** 控制器

如果您的Kubernetes發佈版本未包含快照控制器和客戶需求日、您可以依照下列方式進行部署。

步驟

1. 建立Volume Snapshot客戶需求日。

```
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控制器。

```
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、請開啟 `deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml` 和更新 `namespace` 到您的命名空間。

相關連結

- "卷冊組快照類"
- "Volume快照"

版權資訊

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

由 NetApp 版權資料衍伸之軟體必須遵守下列授權和免責聲明：

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

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

本手冊所述之產品受到一項（含）以上的美國專利、國外專利或申請中專利所保障。

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

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

商標資訊

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