參考
Trident

NetApp
January 15, 2026

目錄

參考

Trident港口

了解更多關於Trident用於通訊的連接埠的資訊。

Trident港口

Trident使用下列連接埠在 Kubernetes 內部進行通訊：

港口	目的
8443	反向通道 HTTPS
8001	Prometheus 指標端點
8000	Trident REST 伺服器
17546	Trident守護程序集 pod 使用的存活/就緒偵測埠

安裝過程中可以使用以下方法變更活性/就緒探針端口： `--probe-port`旗幟。務必確保工作節點上的其他進程沒有使用該連接埠。

Trident REST API

儘管"tridentctl 指令和選項"這是與Trident REST API 互動的最簡單方法，如果您願意，也可以直接使用 REST 端點。

何時使用 REST API

REST API 適用於在非 Kubernetes 部署中使用Trident作為獨立二進位的進階安裝。

為了更好的安全性，Trident `REST API`在 pod 內運作時，預設僅限於 localhost。要改變這種行為，你需要設定 Trident 的 `-address`在其 pod 配置中設定參數。

使用 REST API

若要查看這些 API 的呼叫範例，請傳遞偵錯資訊。(`-d`） 旗幟。更多信息，請參閱"使用 tridentctl 管理 Trident"。

API 的工作原理如下：

得到

`GET <trident-address>/trident/v1/<object-type>`

列出該類型的所有物件。

GET <trident-address>/trident/v1/<object-type>/<object-name>

取得指定物件的詳細資訊。

郵政

POST <trident-address>/trident/v1/<object-type>

建立指定類型的物件。

- 建立物件需要 JSON 配置。有關每種物件類型的詳細說明，請參閱："使用 tridentctl 管理Trident" 。
- 如果對像已存在，則行為會有所不同：後端會更新現有對象，而所有其他對象類型都會使操作失敗。

刪除

DELETE <trident-address>/trident/v1/<object-type>/<object-name>

刪除指定的資源。

與後端或儲存類別關聯的磁碟區將繼續存在；這些磁碟區必須單獨刪除。更多信息，請參閱"使用 tridentctl 管理Trident"。

命令列選項

Trident為Trident編排器提供了幾個命令列選項。您可以使用這些選項來修改您的部署。

日誌記錄

-debug

啟用調試輸出。

-loglevel <level>

設定日誌等級（debug、info、warn、error、fatal）。預設顯示訊息。

Kubernetes

-k8s_pod

使用此選項或 `-k8s_api_server`啟用 Kubernetes 支援。設定此項目後， Trident將使用其所在 pod 的 Kubernetes 服務帳戶憑證來聯絡 API 伺服器。只有當Trident作為 Pod 在啟用了服務帳戶的 Kubernetes 叢集中運行時，此方法才有效。

-k8s_api_server <insecure-address:insecure-port>

使用此選項或 `-k8s_pod`啟用 Kubernetes 支援。指定後， Trident將使用提供的不安全位址和連接埠連接到 Kubernetes API 伺服器。這使得Trident可以部署在 pod 之外；但是，它僅支援與 API 伺服器的不安全連線。為了安全連接，請將Trident部署在具有以下元件的 pod 中： `-k8s_pod`選項。

Docker

-volume_driver <name>

註冊 Docker 外掛程式時使用的驅動程式名稱。預設為 `netapp`。

-driver_port <port-number>

監聽此端口，而不是 UNIX 域套接字。

-config <file>

必填項；您必須指定後端設定檔的路徑。

休息

-address <ip-or-host>

指定 Trident REST 伺服器應該監聽的位址。預設為本機。當監聽本機主機並在 Kubernetes pod 內運作時，REST 介面無法從 pod 外部直接存取。使用 `-address ""`使 REST 介面可從 pod IP 位址存取。

> Trident REST 介面可以設定為僅監聽和提供服務於 127.0.0.1（對於 IPv4）或 [::1]（對於 IPv6）。

-port <port-number>

指定 Trident REST 伺服器應監聽的連接埠。預設值為 8000。

-rest

啟用 REST 介面。預設為真。

Kubernetes 和Trident對象

您可以使用 REST API 透過讀取和寫入資源物件與 Kubernetes 和Trident進行互動。有幾個資源物件決定了 Kubernetes 與Trident、 Trident與儲存以及 Kubernetes 與儲存之間的關係。其中一些物件透過 Kubernetes 進行管理，有些物件則透過Trident進行管理。

這些物體之間是如何相互作用的？

要了解這些物件、它們的用途以及它們如何交互，最簡單的方法或許是追蹤 Kubernetes 用戶發出的單一儲存請求：

1. 使用者創建 `PersistentVolumeClaim`請求一個新的 `PersistentVolume`來自 Kubernetes 的特定大小 `StorageClass`這是管理員之前配置好的。
2. Kubernetes `StorageClass`將其配置器標識為Trident ，並包含告訴Trident如何為請求的類別配置磁碟區的參數。
3. Trident審視自身 `StorageClass`名稱相同，用於標識匹配項 `Backends`和 `StoragePools`它可以用來為該類別配置磁碟區。
4. Trident在匹配的後端配置儲存並建立兩個物件：a `PersistentVolume`在 Kubernetes 中，它告訴 Kubernetes 如何尋找、掛載和處理磁碟區；在Trident中，它維護著兩者之間的關係。 `PersistentVolume`以及實際儲存。
5. Kubernetes 綁定了 `PersistentVolumeClaim`新的 `PersistentVolume`。包含下列部件的艙體 `PersistentVolumeClaim`將該持久性磁碟區掛載到它執行的任何主機上。

6. 使用者創建 `VolumeSnapshot`利用現有的PVC管材， `VolumeSnapshotClass`這顯示Trident有問題。
7. Trident識別與 PVC 關聯的磁碟區，並在其後端建立該磁碟區的快照。它也創造了一個 `VolumeSnapshotContent`指示 Kubernetes 如何辨識快照。
8. 使用者可以創建 `PersistentVolumeClaim`使用 `VolumeSnapshot`作為來源。
9. Trident會識別所需的快照，並執行與建立快照相同的步驟。 `PersistentVolume`和 `Volume`。

如需進一步了解 Kubernetes 對象，我們強烈建議您閱讀以下內容： "持久卷" Kubernetes 文件的這一部分。

Kubernetes `PersistentVolumeClaim`物件

Kubernetes `PersistentVolumeClaim` object 是 Kubernetes 叢集使用者發出的儲存請求。

除了標準規格之外， Trident還允許使用者指定以下磁碟區特定的註釋，以便覆寫您在後端設定中設定的預設值：

註解	成交量選擇權	支援的驅動程式
trident.netapp.io/fileSystem	檔案系統	ontap-san、solidfire-san、ontap-san-economy
trident.netapp.io/cloneFromPVC	克隆源磁碟區	ontap-nas、ontap-san、solidfire-san、azure-netapp-files、gcp-cvs、ontap-san-economy
trident.netapp.io/splitOnClone	splitOnClone	ontap-nas，ontap-san
trident.netapp.io/protocol	協定	任何
trident.netapp.io/exportPolicy	出口政策	ontap-nas、ontap-nas-economy、ontap-nas-flexgroup
trident.netapp.io/snapshotPolicy	快照策略	ontap-nas、ontap-nas-economy、ontap-nas-flexgroup、ontap-san
trident.netapp.io/snapshotReserve	快照儲備	ontap-nas、ontap-nas-flexgroup、ontap-san、gcp-cvs
trident.netapp.io/snapshotDirectory	快照目錄	ontap-nas、ontap-nas-economy、ontap-nas-flexgroup
trident.netapp.io/unixPermissions	unix權限	ontap-nas、ontap-nas-economy、ontap-nas-flexgroup
trident.netapp.io/blockSize	區塊大小	solidfire-san

如果創建的PV具有 `Delete`根據回收策略，當 `PV` 被釋放時（即用戶刪除 `PVC` 時）， `Trident`會同時刪除 `PV` 和支援卷。如果刪除操作失敗， `Trident`會將 `PV` 標記為失敗，並定期重試該操作，直到操作成功或手動刪除 `PV` 為止。如果光伏發電使用 `Retain`策略方面， `Trident`會忽略它，並假定管理員會從 `Kubernetes` 和後端清理它，從而允許在刪除磁碟區之前對其進行備份或檢查。請注意，刪除 `PV` 不會導致`Trident`刪除備份磁碟區。您應該使用 `REST API` 將其刪除。（`tridentctl`）。

Trident支援使用 CSI 規範建立磁碟區快照：您可以建立磁碟區快照並將其用作資料來源來複製現有的 PVC。這樣，就可以將 PV 的特定時間點副本以快照的形式暴露給 Kubernetes。然後可以使用這些快照建立新的PV。看

看 `On-Demand Volume Snapshots`看看這種方法是否可行。

Trident也提供 `cloneFromPVC`和 `splitOnClone`用於建立克隆的註釋。您可以使用這些註解來克隆 PVC，而無需使用 CSI 實作。

例如：如果使用者已經有一個名為 PVC 的 `mysql`用戶可以建立一個名為「新PVC」的 `mysqlclone`透過使用註釋，例如 `trident.netapp.io/cloneFromPVC: mysql`。有了這組註釋， Trident會克隆與 mysql PVC 對應的捲，而不是從頭開始配置卷。

請考慮以下幾點：

- NetApp建議克隆空閒磁碟區。
- PVC 及其克隆應該位於同一個 Kubernetes 命名空間中，並且具有相同的儲存類別。
- 隨著 `ontap-nas`和 `ontap-san`對於驅動程式來說，設定 PVC 註釋可能是有益的。 `trident.netapp.io/splitOnClone`與 `trident.netapp.io/cloneFromPVC`。和 `trident.netapp.io/splitOnClone`設定為 `true`Trident將克隆卷與父卷分離，從而將克隆卷的生命週期與其父卷完全解耦，但代價是損失了一些儲存效率。未設定 `trident.netapp.io/splitOnClone`或將其設為 `false`這樣做可以減少後端空間佔用，但代價是在父捲和克隆卷之間建立依賴關係，使得除非先刪除克隆卷，否則無法刪除父卷。在克隆空資料庫磁碟區時，拆分克隆是有意義的，因為預計該磁碟區及其克隆磁碟區將有很大的不同，並且無法從ONTAP提供的儲存效率中受益。

這 `sample-input`目錄包含可用於Trident的 PVC 定義範例。請參閱有關Trident音量相關參數和設定的完整說明。

Kubernetes `PersistentVolume`物件

Kubernetes `PersistentVolume`此物件代表一塊可供 Kubernetes 叢集使用的儲存空間。它的生命週期與使用它的艙等無關。

> Trident創造 `PersistentVolume`根據其配置的捲，自動將物件註冊到 Kubernetes 叢集。您無需自行管理它們。

當您建立 PVC 時，指的是基於 Trident 的 `StorageClass`Trident使用對應的儲存類別來設定一個新磁碟區，並為該磁碟區註冊一個新的 PV。在配置已設定磁碟區和對應的 PV 時， Trident遵循以下規則：

- Trident會為 Kubernetes 產生一個 PV 名稱，以及一個用於設定儲存的內部名稱。無論哪種情況，名稱在其範圍內都是獨一無二的，這一點都令人放心。
- 容量大小與 PVC 中要求的容量大小盡可能接近，但可能會向上取整到最接近的可分配數量，具體取決於平台。

Kubernetes `StorageClass`物件

Kubernetes `StorageClass`物件透過名稱指定。 `PersistentVolumeClaims`為儲存配置一組屬性。儲存類別本身標識要使用的配置器，並以配置器能夠理解的方式定義該群組屬性。

它是管理員需要建立和管理的兩個基本物件之一。另一個是Trident後端物件。

Kubernetes `StorageClass`使用Trident的物件看起來像這樣：

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: <Name>
provisioner: csi.trident.netapp.io
mountOptions: <Mount Options>
parameters: <Trident Parameters>
allowVolumeExpansion: true
volumeBindingMode: Immediate
```

這些參數是 Trident 特有的，用於告訴Trident如何為該類別配置磁碟區。

儲存類別參數如下：

屬性	類型	必需的	描述
屬性	映射[字串]字串	不	請參閱以下屬性部分。
儲存池	map[string]StringList	不	後端名稱到儲存池清單的映射
附加儲存池	map[string]StringList	不	後端名稱到儲存池清單的映射
排除儲存池	map[string]StringList	不	後端名稱到儲存池清單的映射

儲存屬性及其可能的值可以分為儲存池選擇屬性和 Kubernetes 屬性。

儲存池選擇屬性

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

屬性	類型	價值觀	提供	要求	由...支持
媒體[1]	細繩	機械式硬碟、混合式硬碟、固態硬碟	Pool 包含此類媒體；混合型媒體是指兩者兼具。	指定的媒體類型	ontap-nas、ontap-nas-economy、ontap-nas-flexgroup、ontap-san、solidfire-san
供應類型	細繩	薄的，厚的	池支援這種配置方法	指定的配置方法	厚：全部 ontap；薄：全部 ontap 和 solidfire-san

屬性	類型	價值觀	提供	要求	由...支持
後端類型	細繩	ontap-nas 、ontap-nas-economy、ontap-nas-flexgroup 、ontap-san 、solidfire-san 、gcp-cvs 、azure-netapp-files、ontap-san-economy	池屬於這種類型的後端	指定的後端	所有司機
快照	布林值	真，假	儲存池支援帶快照的磁碟區	已啟用快照的磁碟區	ontap-nas 、ontap-san 、solidfire-san 、gcp-cvs
複製	布林值	真，假	儲存池支援磁碟區克隆	已啟用克隆的磁碟區	ontap-nas 、ontap-san 、solidfire-san 、gcp-cvs
加密	布林值	真，假	儲存池支援加密磁碟區	已啟用加密的磁碟區	ontap-nas 、ontap-nas-economy、ontap-nas-flexgroups、ontap-san
每秒輸入/輸出次數	整數	正整數	Pool 能夠保證在此範圍內的 IOPS	容量保證了這些IOPS	solidfire-san

[1]： ONTAP Select系統不支援此系統

在大多數情況下，所要求的值會直接影響配置；例如，請求厚配置會導致厚配置磁碟區的產生。但是，Element 儲存池使用其提供的最小和最大 IOPS 來設定 QoS 值，而不是使用請求的值。在這種情況下，請求的值僅用於選擇儲存池。

理想情況下，你可以使用 `attributes`單獨建模，以滿足特定類別的需求所需的儲存品質。 Trident會自動發現並選擇符合所有條件的儲存池 `attributes`您指定的。

如果您發現自己無法使用 `attributes`若要自動為類別選擇合適的池，您可以使用 `storagePools`和 `additionalStoragePools`參數用於進一步細化池子，甚至選擇一組特定的池子。

您可以使用 `storagePools`參數用於進一步限制與任何指定參數相符的池集合。 `attributes` 。換句話說， Trident使用了由以下方式識別的池的交集： `attributes`和 `storagePools`配置參數。您可以單獨使用其中一個參數，也可以同時使用兩個參數。

您可以使用 `additionalStoragePools`此參數用於擴展Trident用於配置的池集，而不管 Trident 選擇的任何池。 `attributes`和 `storagePools`參數。

您可以使用 `excludeStoragePools`用於篩選Trident用於資源配置的池集合的參數。使用此參數會移除所有符合的池。

在 `storagePools`和 `additionalStoragePools`參數，每個條目都採用以下形式 `<backend>:<storagePoolList>`，在哪裡 `<storagePoolList>`是指定後端儲存池的逗號分隔清單。例如，一個值 `additionalStoragePools`可能看起來像 `ontapnas_192.168.1.100:aggr1,aggr2;solidfire_192.168.1.101:bronze`。這些清單接受後端值和清單值的正規表示式值。您可以使用 `tridentctl get backend`取得後端及其連線池的清單。

Kubernetes屬性

這些屬性對Trident在動態設定期間選擇儲存池/後端沒有任何影響。相反，這些屬性只是提供 Kubernetes 持久卷支援的參數。工作節點負責檔案系統建立操作，可能需要檔案系統實用程序，例如 xfsprogs。

屬性	類型	價值觀	描述	相關驅動因素	**Kubernetes** 版本
檔案系統類型	細繩	ext4、ext3、xfs	區塊卷的檔案系統類型	solidfire-san、ontap-nas、ontap-nas-economy、ontap-nas-flexgroup、ontap-san、ontap-san-economy	全部
允許卷擴展	布林值	真，假	啟用或停用對增大PVC尺寸的支持	ontap-nas、ontap-nas-economy、ontap-nas-flexgroup、ontap-san、ontap-san-economy、solidfire-san、gcp-cvs、azure-netapp-files	1.11+
容量綁定模式	細繩	立即，等待第一位消費者	選擇何時進行磁碟區綁定和動態配置	全部	1.19 - 1.26

- 這 `fsType`此參數用於控制 SAN LUN 所需的檔案系統類型。此外，Kubernetes 還利用了以下資訊： `fsType`在儲存類別中表示檔案系統存在。可以透過以下方式控製卷所有權： `fsGroup`僅當 `fsType`已設定。請參閱"Kubernetes：為 Pod 或容器設定安全上下文"有關如何使用設定卷所有權的概述 `fsGroup`情境. Kubernetes 將會應用 `fsGroup`僅當滿足以下條件時才有值：

  - `fsType`設定在儲存類別中。
  - PVC接取方式為RWO。

  對於 NFS 儲存驅動程序，檔案系統已作為 NFS 匯出的一部分存在。為了使用 `fsGroup`儲存類別仍然需要指定一個 `fsType`您可以將其設定為 `nfs`或任何非空值。

- 請參閱"擴大銷量"有關擴容的更多詳情。
- Trident安裝程式包提供了幾個範例儲存類別定義，可供Trident使用。`sample-input/storage-class-*.yaml` 。刪除 Kubernetes 儲存類別會導致對應的Trident儲存類別也被刪除。

Kubernetes `VolumeSnapshotClass`物件

Kubernetes `VolumeSnapshotClass`物體類似於 `StorageClasses`。它們有助於定義多種儲存類別，並被磁碟區快照引用，以將快照與所需的快照類別關聯起來。每個磁碟區快照都與一個磁碟區快照類別相關聯。

一個 `VolumeSnapshotClass`應由管理員定義以建立快照。建立卷宗快照類別時，定義如下：

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

這 `driver`向 Kubernetes 指定請求卷快照的 `csi-snapclass`類別由Trident處理。這 `deletionPolicy`指定必須刪除快照時要執行的操作。什麼時候 `deletionPolicy`設定為 `Delete`當刪除快照時，儲存叢集上的磁碟區快照物件以及底層快照也會被刪除。或者，將其設為 `Retain`意味著 `VolumeSnapshotContent`並保留實體快照。

Kubernetes `VolumeSnapshot`物件

Kubernetes `VolumeSnapshot` object 是建立磁碟區快照的請求。 PVC 代表使用者對磁碟區的請求，磁碟區快照則是使用者對現有 PVC 建立快照的請求。

當收到磁碟區快照請求時， Trident會自動在後端建立磁碟區快照，並透過建立唯一識別碼來公開該快照。 `VolumeSnapshotContent`目的。您可以從現有的 PVC 建立快照，並在建立新的 PVC 時將這些快照用作資料來源。

VolumeSnapshot 的生命週期與來源 PVC 無關：即使來源 PVC 被刪除，快照仍然會存在。刪除具有關聯快照的 PVC 時， Trident會將此 PVC 的後備卷標記為「正在刪除」狀態，但不會完全刪除。當所有關聯的快照都被刪除後，該磁碟區也會被移除。

Kubernetes `VolumeSnapshotContent`物件

Kubernetes `VolumeSnapshotContent`該物件表示從已配置的磁碟區中取得的快照。它類似於`PersistentVolume`表示儲存叢集上已配置的快照。類似 `PersistentVolumeClaim`和 `PersistentVolume`當創建快照時，物件會… `VolumeSnapshotContent`物件與…保持一對一的映射關係 `VolumeSnapshot`該物件請求建立快照。

這 `VolumeSnapshotContent`物件包含唯一標識快照的詳細信息，例如： `snapshotHandle` 。這`snapshotHandle`是PV名稱和名稱的獨特組合 `VolumeSnapshotContent`目的。

當收到快照請求時， Trident會在後端建立快照。快照建立完成後， Trident會配置一個`VolumeSnapshotContent`對象，從而將快照暴露給 Kubernetes API。

通常情況下，你不需要管理 `VolumeSnapshotContent`目的。但也有例外情況，例如你想…"匯入磁碟區快照"在Trident之外建立。

Kubernetes `VolumeGroupSnapshotClass`物件

Kubernetes `VolumeGroupSnapshotClass`物體類似於 `VolumeSnapshotClass`。它們有助於定義多種儲存類別，並被磁碟區組快照引用，以將快照與所需的快照類別關聯起來。每個磁碟區組快照都與一個磁碟區組快照類別相關聯。

一個 `VolumeGroupSnapshotClass`應由管理員定義以建立快照群組。使用下列定義建立磁碟區組快照類別：

```
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
```

這 `driver`向 Kubernetes 指定請求卷宗組快照的權限。 `csi-group-snap-class`類別由Trident處理。這`deletionPolicy`指定必須刪除群組快照時要執行的操作。什麼時候 `deletionPolicy`設定為 `Delete`當刪除快照時，磁碟區組快照物件以及儲存叢集上的基礎快照也會被刪除。或者，將其設為 `Retain`意味著`VolumeGroupSnapshotContent`並保留物理快照。

Kubernetes `VolumeGroupSnapshot`物件

Kubernetes `VolumeGroupSnapshot`該物件是建立多個磁碟區的快照的請求。就像 PVC 代表使用者對磁碟區的請求一樣，磁碟區組快照是使用者對現有 PVC 建立快照的請求。

當收到磁碟區組快照請求時， Trident會自動在後端管理磁碟區的群組快照創建，並透過建立唯一識別碼來公開該快照。 `VolumeGroupSnapshotContent`目的。您可以從現有的 PVC 建立快照，並在建立新的 PVC 時將這些快照用作資料來源。

VolumeGroupSnapshot 的生命週期與來源 PVC 無關：即使來源 PVC 被刪除，快照仍然會保留。刪除具有關聯快照的 PVC 時， Trident會將此 PVC 的後備卷標記為「正在刪除」狀態，但不會完全刪除。當所有關聯的快照都被刪除時，磁碟區組快照也會被刪除。

Kubernetes `VolumeGroupSnapshotContent`物件

Kubernetes `VolumeGroupSnapshotContent`該物件表示從已配置的磁碟區中取得的群組快照。它類似於`PersistentVolume`表示儲存叢集上已配置的快照。類似 `PersistentVolumeClaim`和 `PersistentVolume`當創建快照時，物件會… `VolumeSnapshotContent`物件與…保持一對一的映射關係 `VolumeSnapshot`該物件請求建立快照。

這 `VolumeGroupSnapshotContent`物件包含用於標識快照組的詳細信息，例如：`volumeGroupSnapshotHandle`以及儲存系統上存在的各個 volumeSnapshotHandles。

當收到快照請求時， Trident會在後端建立磁碟區組快照。卷冊組快照建立完成後， Trident會進行設定。`VolumeGroupSnapshotContent`對象，從而將快照暴露給 Kubernetes API。

Kubernetes `CustomResourceDefinition`物件

Kubernetes 自訂資源是 Kubernetes API 中的端點，由管理員定義，用於對類似物件進行分組。 Kubernetes 支援建立自訂資源來儲存物件集合。您可以透過執行以下命令來取得這些資源定義。 `kubectl get crds` 。

Kubernetes 將自訂資源定義 (CRD) 及其關聯的物件元資料儲存在其元資料儲存中。這樣就無需為Trident單獨開設商店了。

Trident的使用 `CustomResourceDefinition`用於保存Trident物件識別的對象，例如Trident後端、 Trident儲存類別和Trident磁碟區。這些物件由Trident管理。此外，CSI 卷快照框架引入了一些定義卷快照所需的 CRD。

CRD 是 Kubernetes 的一種建構。上述資源的物件由Trident建立。舉個簡單的例子，當使用以下方式建立後端時 `tridentctl`相應的 `tridentbackends`建立 CRD 物件供 Kubernetes 使用。

關於Trident的CRD，需要記住以下幾點：

- 安裝Trident時，會建立一組 CRD，可以像使用任何其他資源類型一樣使用這些 CRD。
- 使用以下方式卸載Trident時 `tridentctl uninstall`使用指令後， Trident pod 會被刪除，但已建立的 CRD 不會被清理。請參閱"解除安裝Trident"了解如何將Trident完全移除並從頭開始重新配置。

Trident `StorageClass`物件

Trident為 Kubernetes 建立相符的儲存類 `StorageClass`指定對象 `csi.trident.netapp.io`在他們的供應領域。儲存類別名稱與 Kubernetes 的名稱相符。 `StorageClass`它所代表的對象。

使用 Kubernetes 時，這些物件會在 Kubernetes 叢集啟動時自動建立。 `StorageClass`已註冊使用Trident作為設定器的設定器。

儲存類別包含對磁碟區的一系列要求。 Trident會將這些要求與每個儲存池中存在的屬性進行匹配；如果匹配，則該儲存池是使用該儲存類別配置磁碟區的有效目標。

您可以使用 REST API 建立儲存類別配置，直接定義儲存類別。但是，對於 Kubernetes 部署，我們期望在註冊新的 Kubernetes 執行個體時建立它們。 `StorageClass`物體。

Trident後端對象

後端代表儲存提供者， Trident在其上配置磁碟區；單一Trident實例可以管理任意數量的後端。

這是您可以自行建立和管理的兩種物件類型之一。另一個是 Kubernetes。 `StorageClass`目的。

有關如何建構這些物件的更多信息，請參閱："配置後端" 。

Trident `StoragePool`物件

儲存池代表每個後端可用於配置的不同位置。對於ONTAP而言，這些對應於 SVM 中的聚合。對於NetApp HCI/SolidFire，這些對應於管理員指定的 QoS 頻段。對於Cloud Volumes Service，這些對應於雲端提供者區域。每個儲存池都有一組獨特的儲存屬性，這些屬性定義了其效能特徵和資料保護特徵。

與此處的其他物件不同，儲存池候選對象始終會自動發現和管理。

Trident `Volume`物件

磁碟區是配置的基本單元，包括後端端點（例如 NFS 共用）以及 iSCSI 和 FC LUN。在 Kubernetes 中，這些直接對應於 `PersistentVolumes`。建立磁碟區時，請確保它具有儲存類別（決定磁碟區的部署位置）和大小。

- 在 Kubernetes 中，這些物件是自動管理的。您可以查看這些信息，以了解Trident 的部署情況。
- 刪除帶有關聯快照的 PV 時，對應的Trident磁碟區將更新為 正在刪除 狀態。若要刪除Trident磁碟區，您應該刪除該磁碟區的快照。

卷配置定義了已配置磁碟區應具有的屬性。

屬性	類型	必需的	描述
版本	細繩	不	Trident API 版本（「1」）
姓名	細繩	是的	要建立的磁碟區的名稱
儲存類別	細繩	是的	配置磁碟區時要使用的儲存類
尺寸	細繩	是的	要配置的磁碟區的大小（以位元組為單位）
協定	細繩	不	使用的協定類型：“檔案”或“區塊”
內部名稱	細繩	不	儲存系統中物件的名稱；由Trident產生
克隆源磁碟區	細繩	不	ontap（nas、san）和solidfire-*：要克隆的捲的名稱
splitOnClone	細繩	不	ontap（nas，san）：將克隆體從其父體中分離出來

屬性	類型	必需的	描述
快照策略	細繩	不	ontap-*：要使用的快照策略
快照儲備	細繩	不	ontap-*：為快照預留的磁碟區百分比
出口政策	細繩	不	ontap-nas*：要使用的匯出策略
快照目錄	布林值	不	ontap-nas*：快照目錄是否可見
unix權限	細繩	不	ontap-nas*：初始 UNIX 權限
區塊大小	細繩	不	solidfire-*：塊/扇區大小
檔案系統	細繩	不	檔案系統類型

Trident生成 `internalName`建立卷時。這包括兩個步驟。首先，它會添加儲存前綴（預設值）。 `trident`或後端配置中的前綴）加到磁碟區名稱，從而得到如下形式的名稱 `<prefix>-<volume-name>`。然後它會對名稱進行清理，替換後端不允許的字元。對於ONTAP後端，它會將連字號替換為底線（因此，內部名稱變為 <prefix>_<volume-name>）。對於 Element 後端，它會將下劃線替換為連字符。

您可以使用磁碟區配置透過 REST API 直接設定卷，但在 Kubernetes 部署中，我們預計大多數使用者將使用標準的 Kubernetes 管理配置。 `PersistentVolumeClaim`方法。 Trident會在設定過程中自動建立此磁碟區物件。

Trident `Snapshot`物件

快照是磁碟區在特定時間點的副本，可用於設定新磁碟區或復原狀態。在 Kubernetes 中，這些直接對應於 `VolumeSnapshotContent`物體。每個快照都與一個磁碟區相關聯，該磁碟區是快照資料的來源。

每個 `Snapshot`物件包含以下屬性：

屬性	類型	必需的	描述
版本	細繩	是的	Trident API 版本（「1」）
姓名	細繩	是的	Trident快照物件的名稱
內部名稱	細繩	是的	儲存系統上Trident快照物件的名稱
卷名	細繩	是的	建立快照的持久磁碟區的名稱
volumeInternalName	細繩	是的	儲存系統上關聯的Trident磁碟區物件的名稱

在 Kubernetes 中，這些物件是自動管理的。您可以查看這些信息，以了解Trident 的部署情況。

當 Kubernetes `VolumeSnapshot`建立物件請求後， `Trident` 的工作原理是在後端儲存系統上建立快照物件。這 `internalName`此快照物件是透過組合前綴產生的。 `snapshot-`和 `UID`的

`VolumeSnapshot`物件（例如， `snapshot-e8d8a0ca-9826-11e9-9807-525400f3f660`）。`volumeName`和`volumeInternalName`透過取得支援卷的詳細資訊來填入。

Trident `ResourceQuota`目的

Trident守護程式消耗一個`system-node-critical`優先等級——Kubernetes 中可用的最高優先等級——以確保Trident能夠在節點優雅關閉期間識別和清理卷，並允許Trident daemonset pod 在資源壓力高的叢集中搶佔優先順序較低的工作負載。

為了實現這一目標， Trident採用了一種`ResourceQuota`確保Trident守護程式集上的「system-node-critical」優先權類別已滿足。在部署和建立守護程序集之前， Trident會查找`ResourceQuota`對象，如果未發現，則應用它。

如果您需要對預設資源配額和優先類別進行更多控制，您可以產生一個`custom.yaml`或配置`ResourceQuota`使用 Helm Chart 的物件。

以下是一個 `ResourceQuota` 物件優先考慮Trident守護程式集的範例。

```
apiVersion: <version>
kind: ResourceQuota
metadata:
  name: trident-csi
  labels:
    app: node.csi.trident.netapp.io
spec:
  scopeSelector:
    matchExpressions:
      - operator: In
        scopeName: PriorityClass
        values:
          - system-node-critical
```

有關資源配額的更多信息，請參閱"Kubernetes：資源配額"。

清理`ResourceQuota`如果安裝失敗

在極少數情況下，如果安裝失敗，`ResourceQuota`物件已創建，首次嘗試"解除安裝"然後重新安裝。

如果這樣不行，就手動刪除。`ResourceQuota`目的。

消除 `ResourceQuota`

如果您希望自行控制資源分配，您可以移除Trident。`ResourceQuota`使用以下命令物件：

```
kubectl delete quota trident-csi -n trident
```

Pod 安全標準 (PSS) 和安全情境約束 (SCC)

Kubernetes Pod 安全標準 (PSS) 和 Pod 安全性原則 (PSP) 定義了權限等級並限制了 Pod 的行為。 OpenShift 安全性情境限制 (SCC) 類似地定義了 OpenShift Kubernetes Engine 特有的 pod 限制。為了實現這種自訂功能， Trident會在安裝過程中啟用某些權限。以下各節詳細介紹了Trident設定的權限。

PSS 取代了 Pod 安全性策略 (PSP)。 PSP 在 Kubernetes v1.21 中已被棄用，並將於 v1.25 中移除。更多信息，請參閱"Kubernetes：安全性"。

必需的 **Kubernetes** 安全上下文和相關字段

允許	描述
特權	CSI 要求掛載點是雙向的，這表示Trident節點 pod 必須執行特權容器。更多信息，請參閱"Kubernetes：掛載傳播"。
主機網路	iSCSI守護程式需要此元件。 `iscsiadm`管理 iSCSI 掛載點，並使用主機網路與 iSCSI 守護程式通訊。
主機 IPC	NFS 使用進程間通訊 (IPC) 與 NFSD 進行通訊。
主機 PID	開始需要 `rpc-statd`適用於 NFS。 Trident會查詢主機進程以確定是否 `rpc-statd`在掛載 NFS 磁碟區之前運行。

功能	這 `SYS_ADMIN`此功能作為特權容器的預設功能的一部分提供。例如，`Docker` 為特權容器設定了以下功能：
	`CapPrm: 0000003fffffffff`
	`CapEff: 0000003fffffffff`

賽康普	在特權容器中，Seccomp 設定檔始終為「Unconfined」；因此，它無法在Trident中啟用。
SELinux	在 OpenShift 上，特權容器運行在 `spc_t`（「超級特權容器」）域，而非特權容器則是運行在...域中。`container_t`領域。在 `containerd`， 和 `container-selinux`安裝完成後，所有容器都在運作。`spc_t`域，這實際上禁用了 SELinux。因此， Trident不會增加 `seLinuxOptions`到容器中。
DAC	特權容器必須以 root 使用者身分執行。非特權容器以 root 使用者身分執行，以存取 CSI 所需的 Unix 套接字。

艙體安全標準 **(PSS)**

標籤	描述	預設
pod-security.kubernetes.io/enforce pod-security.kubernetes.io/enforce-version	允許將Trident控制器和節點新增至安裝命名空間。請勿變更命名空間標籤。	enforce: privileged enforce-version: <version of the current cluster or highest version of PSS tested.>

更改命名空間標籤可能會導致 Pod 無法調度，出現「建立時發生錯誤：...」或「警告：trident-csi-...」等錯誤。如果發生這種情況，請檢查命名空間標籤是否為 `privileged`已更改。如果是這樣，請重新安裝Trident。

Pod 安全策略 (PSP)

場地	描述	預設
allowPrivilegeEscalation	特權容器必須允許權限提升。	true
allowedCSIDrivers	Trident不使用內嵌 CSI 臨時磁碟區。	空的
allowedCapabilities	非特權Trident容器不需要比預設集更多的功能，而特權容器將被授予所有可能的功能。	空的
allowedFlexVolumes	Trident不使用"FlexVolume驅動器"因此，它們不包含在允許的音量列表中。	空的
allowedHostPaths	Trident節點 pod 會掛載節點的根檔案系統，因此設定此清單沒有任何好處。	空的
allowedProcMountTypes	Trident不使用任何 ProcMountTypes。	空的
allowedUnsafeSysctls	Trident不需要任何不安全措施 sysctls。	空的
defaultAddCapabilities	特權容器無需添加任何功能。	空的
defaultAllowPrivilegeEscalation	權限提升權限是在每個Trident pod 中處理的。	false
forbiddenSysctls	不 `sysctls`允許。	空的
fsGroup	Trident容器以root權限運作。	RunAsAny
hostIPC	掛載 NFS 磁碟區需要主機 IPC 與主機通訊 nfsd	true
hostNetwork	iscsiadm 需要主機網路才能與 iSCSI 守護程式通訊。	true
hostPID	需要主機 PID 來進行檢查 `rpc-statd`正在節點上運行。	true
hostPorts	Trident不使用任何主機連接埠。	空的

場地	描述	預設
`privileged`	Trident節點 pod 必須運行特權容器才能掛載磁碟區。	`true`
`readOnlyRootFilesystem`	Trident節點 pod 必須寫入節點檔案系統。	`false`
`requiredDropCapabilities`	Trident節點 pod 運作的是特權容器，無法放棄任何功能。	`none`
`runAsGroup`	Trident容器以root權限運作。	`RunAsAny`
`runAsUser`	Trident容器以root權限運作。	`runAsAny`
`runtimeClass`	Trident不使用 `RuntimeClasses`。	空的
`seLinux`	Trident不設定 `seLinuxOptions`因為目前容器運行時和 Kubernetes 發行版在處理 SELinux 方面有差異。	空的
`supplementalGroups`	Trident容器以root權限運作。	`RunAsAny`
`volumes`	Trident pods 需要這些音量外掛。	`hostPath, projected, emptyDir`

安全上下文約束 (SCC)

標籤	描述	預設
`allowHostDirVolumePlugin`	Trident節點 pod 會掛載節點的根檔案系統。	`true`
`allowHostIPC`	掛載 NFS 磁碟區需要主機 IPC 與主機通訊 `nfsd`。	`true`
`allowHostNetwork`	iscsiadm 需要主機網路才能與 iSCSI 守護程式通訊。	`true`
`allowHostPID`	需要主機 PID 來進行檢查 `rpc-statd`正在節點上運行。	`true`
`allowHostPorts`	Trident不使用任何主機連接埠。	`false`
`allowPrivilegeEscalation`	特權容器必須允許權限提升。	`true`
`allowPrivilegedContainer`	Trident節點 pod 必須運行特權容器才能掛載磁碟區。	`true`
`allowedUnsafeSysctls`	Trident不需要任何不安全措施 `sysctls`。	`none`
`allowedCapabilities`	非特權Trident容器不需要比預設集更多的功能，而特權容器將被授予所有可能的功能。	空的
`defaultAddCapabilities`	特權容器無需添加任何功能。	空的
`fsGroup`	Trident容器以root權限運作。	`RunAsAny`

標籤	描述	預設
`groups`	此 SCC 專為Trident而設，並與其使用者綁定。	空的
`readOnlyRootFilesystem`	Trident節點 pod 必須寫入節點檔案系統。	`false`
`requiredDropCapabilities`	Trident節點 pod 運作的是特權容器，無法放棄任何功能。	`none`
`runAsUser`	Trident容器以root權限運作。	`RunAsAny`
`seLinuxContext`	Trident不設定 `seLinuxOptions`因為目前容器運行時和 Kubernetes 發行版在處理 SELinux 方面有差異。	空的
`seccompProfiles`	特權容器始終以“非限制”模式運作。	空的
`supplementalGroups`	Trident容器以root權限運作。	`RunAsAny`
`users`	提供了一個條目，用於將此 SCC 綁定到Trident命名空間中的Trident用戶。	無
`volumes`	Trident pods 需要這些音量外掛。	`hostPath, downwardAPI, projected, emptyDir`