# 管理及監控 Trident

Trident

NetApp
March 05, 2026

# 目錄

# 管理及監控 Trident

# 升級 Trident

## 升級 Trident

從 24.02 版開始、 Trident 遵循四個月的發行步調、每個日曆年度提供三個主要版本。每個新版本均以舊版為基礎、並提供新功能、效能增強、錯誤修正及改善功能。我們建議您每年至少升級一次、以充分利用 Trident 的新功能。

### 升級前的考量

升級至最新版的 Trident 時、請考慮下列事項：

- 在指定的 Kubernetes 叢集中、所有命名空間都應該只安裝一個 Trident 執行個體。
- Trident 23.07 及更新版本需要 v1 Volume 快照、不再支援 Alpha 或 beta 快照。
- 如果您在中建立了 Cloud Volumes Service for Google Cloud "CVS服務類型"、則從 Trident 23.01 升級時、必須更新後端組態、才能使用 `standardsw`或 `zoneredundantstandardsw`服務層級。如果無法在後端更新 `serviceLevel`、可能會導致磁碟區失敗。如 "CVS 服務類型範例" 需詳細資訊、請參閱。
- 升級時、 Trident 必須提供 `parameter.fsType`使用中的 `StorageClasses`資訊。您可以在不中斷現有磁碟區的情況下刪除及重新建立 `StorageClasses`磁碟區。
  - 這是強制執行 SAN 磁碟區的 需求 "安全性內容" 。
  - https://github.com/的 NetApp / Trident / 樹狀結構 / 主機 / TridentTrident 安裝程式 / 範例輸入 [ 範例輸入 ^] 目錄包含範例、例如 https://github.com/NetApp / Trident / blob / 主機 / Trident 安裝程式 / 範例輸入 / 儲存類別 - 範例 / 儲存類別 - basic.yaml.templ[`storage-class-basic.yaml.templ` ^] 和連結： https://github.com/NetApp / Trident / blob/master/installer-installer-installer-sample 儲存類別 - yaml-store-class-store-sample[`storage-class-bronze-default.yaml` 。
  - 如需詳細資訊、請 "已知問題"參閱。

### 步驟 1 ：選取版本

Trident 版本遵循日期 `YY.MM`命名慣例、其中「是」是年份的最後兩位數、「 MM 」是月份。DOT 版本遵循 `YY.MM.X`慣例、其中「 X 」是修補程式層級。您將根據要升級的版本、選擇要升級的版本。

- 您可以直接升級至安裝版本的四個版本範圍內的任何目標版本。例如、您可以直接從 24.06 （或任何 24.06 點版本）升級至 25.02 。
- 如果您要從四個版本的外部版本升級、請執行多步驟升級。使用升級說明從升級至最新版本、以符合四個版本的 "舊版" 視窗。例如、如果您執行的是 23.01 、而且想要升級至 25.02 ：
  a. 第一次從 23.01 升級至 24.02 。
  b. 然後從 24.02 升級至 25.02 。

> 在 OpenShift Container Platform 上使用 Trident 運算子進行升級時、您應升級至 Trident 21.01.1 或更新版本。隨21.01.0一起發行的Trident運算子包含已在21.01.1中修正的已知問題。如需詳細資訊、請 "GitHub問題詳細資料"參閱。

步驟 **2** ：確定原始安裝方法

若要判斷您最初安裝 Trident 的版本：

1. 用於 `kubectl get pods -n trident`檢查 Pod 。
   - 如果沒有運算子 Pod 、則使用安裝 Trident `tridentctl` 。
   - 如果有操作員 Pod 、則 Trident 是使用 Trident 操作員手動或使用 Helm 來安裝。
2. 如果有操作員 Pod 、請使用 `kubectl describe torc`判斷是否使用 Helm 安裝 Trident 。
   - 如果有 Helm 標籤、則使用 Helm 安裝 Trident 。
   - 如果沒有 Helm 標籤、則會使用 Trident 操作員手動安裝 Trident 。

步驟 **3** ：選擇升級方法

一般而言，您應該使用初始安裝所使用的相同方法進行升級"在安裝方法之間移動"，不過您可以。升級 Trident 有兩個選項。

- "使用Trident營運者進行升級"

  我們建議您在與營運商一起升級之前、先進行審查"瞭解營運商升級工作流程"。

*

## 與營運者一起升級

瞭解營運商升級工作流程

在使用 Trident 操作員升級 Trident 之前、您應該先瞭解升級期間所發生的背景程序。其中包括 Trident 控制器、控制器 Pod 和節點 Pod 的變更、以及啟用循環更新的節點示範集。

**Trident** 營運商升級處理

安裝和升級 Trident 的其中一項"使用 Trident 運算子的優點"、是在不中斷現有掛載磁碟區的情況下、自動處理 Trident 和 Kubernetes 物件。如此一來、 Trident 就能支援零停機的升級、或"滾動更新 _"。尤其是 Trident 運算子會與 Kubernetes 叢集通訊、以便：

- 刪除並重新建立 Trident Controller 部署和節點示範集。
- 以新版本更換 Trident 控制器 Pod 和 Trident 節點 Pod 。
  - 如果節點未更新、則不會阻止其餘節點更新。
  - 只有執行中 Trident Node Pod 的節點才能裝載磁碟區。

有關 Kubernetes 叢集上 Trident 架構的詳細資訊"Trident 架構"、請參閱。

營運商升級工作流程

當您使用 Trident 運算子啟動升級時：

1. * Trident 運算子 * ：
    a. 偵測目前安裝的 Trident 版本（版本 *n* ）。
    b. 更新所有 Kubernetes 物件、包括 CRD 、 RBAC 和 Trident SVC 。
    c. 刪除版本 *n* 的 Trident 控制器部署。
    d. 為版本 *n+1* 建立 Trident Controller 部署。
2. * Kubernetes* 為 _n+1 建立 Trident 控制器 Pod 。
3. * Trident 運算子 * ：
    a. 刪除 *n* 的 Trident 節點示範集。操作人員不會等待節點 Pod 終止。
    b. 為 *n+1* 建立 Trident 節點 Demont 。
4. * Kubernetes* 會在未執行 Trident Node Pod 的節點上建立 Trident Node Pod 。如此可確保節點上的任何版本、都不會有超過一個 Trident Node Pod 。

使用 **Trident** 營運商或 **Helm** 升級 **Trident** 安裝

## 您可以手動或使用 Helm 、使用 Trident 營運商來升級 Trident 。您可以從 Trident 營運商安裝升級至其他 Trident 營運商安裝、或從安裝升級 `tridentctl`至 Trident 營運商版本。在升級 Trident 操作員安裝之前、請先檢閱"選擇升級方法"。

### 升級手動安裝

您可以從叢集範圍的 Trident 運算子安裝升級到另一個叢集範圍的 Trident 運算子安裝。所有 Trident 版本 21.01 及更新版本均使用叢集範圍的運算子。

> 若要從使用命名空間範圍運算子（ 20.07 至 20.10 版）安裝的 Trident 升級、請使用 Trident 的升級指示"您已安裝的版本"。

#### 關於這項工作

Trident 提供一個套件檔案、可讓您用來安裝運算子、並為 Kubernetes 版本建立相關的物件。

- 對於運行 Kubernetes 1.24 的羣集，請使用 "bunder_pre_1_25.yaml"。
- 對於運行 Kubernetes 1.25 或更高版本的羣集，請使用 "bunder_POST_1_25.yaml"。

#### 開始之前

確保您使用的是運行的 Kubernetes 羣集"支援的Kubernetes版本"。

#### 步驟

1. 驗證您的 Trident 版本：

```
./tridentctl -n trident version
```

2. 刪除用於安裝目前 Trident 執行個體的 Trident 運算子。例如、如果您是從 23.07 升級、請執行下列命令：

```
kubectl delete -f 23.07.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

3. 如果使用屬性自訂初始安裝 `TridentOrchestrator`、您可以編輯 `TridentOrchestrator`物件來修改安裝參數。這可能包括針對離線模式指定鏡射Trident和csi映像登錄、啟用偵錯記錄或指定映像提取機密所做的變更。
4. 使用適用於您環境的正確套件 YAML 檔案安裝 Trident 、其中 _Kubernetes <bundle.yaml> 版本為 _ `bundle_pre_1_25.yaml` 或 `bundle_post_1_25.yaml`以 Kubernetes 版本為基礎。例如、如果您要安裝 Trident 25.02 、請執行下列命令：

```
kubectl create -f 25.02.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

#### 升級 **Helm** 安裝

您可以升級 Trident Helm 安裝。

> 將已安裝 Trident 的 Kubernetes 叢集從 1.24 升級至 1.25 或更新版本時、您必須 `true`先更新 values.yaml 以設定 `excludePodSecurityPolicy`或新增 `--set excludePodSecurityPolicy=true`至 `helm upgrade`命令、才能升級叢集。

如果您已經將 Kubernetes 叢集從 1.24 升級至 1.25 、而不升級 Trident helm 、則 helm 升級將會失敗。若要順利完成升級、請先執行下列步驟：

1. 從安裝 helm-mapkubeaps 外掛程式 https://github.com/helm/helm-mapkubeapis。
2. 在安裝 Trident 的命名空間中、為 Trident 版本執行演習。這會列出將會清除的資源。

```
helm mapkubeapis --dry-run trident  --namespace trident
```

3. 以 helm 執行完整執行以進行清理。

```
helm mapkubeapis trident  --namespace trident
```

#### 步驟

1. 如果您 "已使用 Helm 安裝 Trident"是、您可以在單一步驟中使用 `helm upgrade trident netapp-trident/trident-operator --version 100.2502.0` 進行升級。如果您未新增 Helm repo 或無法使用它來升級：
   a. 從下載最新的 Trident 版本"GitHub的_Assets區段"。
   b. 使用 `helm upgrade` 反映您要升級至的版本的命令 `trident-operator-25.02.0.tgz` 。

```
helm upgrade <name> trident-operator-25.02.0.tgz
```

如果您在初始安裝期間設定自訂選項（例如指定 Trident 和 CSI 映像的私有、鏡射登錄）、請使用附加命令 `--set`、 `helm upgrade`以確保升級命令中包含這些選項、否則這些值會重設為預設值。

2. 執行 `helm list`以確認圖表和應用程式版本均已升級。執行 `tridentctl logs`以檢閱任何偵錯訊息。

#### 從安裝升級 `tridentctl`至 Trident 營運商

您可以從安裝升級至最新版本的 Trident 操作員 `tridentctl`。現有的後端和 PVC 將會自動提供使用。

在安裝方法之間切換之前，請參閱"在安裝方法之間移動"。

步驟

1. 下載最新的 Trident 版本。

```
# Download the release required [25.02.0]
mkdir 25.02.0
cd 25.02.0
wget
https://github.com/NetApp/trident/releases/download/v25.02.0/trident-
installer-25.02.0.tar.gz
tar -xf trident-installer-25.02.0.tar.gz
cd trident-installer
```

2. 從資訊清單建立 `tridentorchestrator`客戶需求日。

```
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
```

3. 在同一個命名空間中部署叢集範圍的運算子。

```
kubectl create -f deploy/<bundle-name.yaml>

serviceaccount/trident-operator created
clusterrole.rbac.authorization.k8s.io/trident-operator created
clusterrolebinding.rbac.authorization.k8s.io/trident-operator created
deployment.apps/trident-operator created
podsecuritypolicy.policy/tridentoperatorpods created

#Examine the pods in the Trident namespace
NAME                                  READY   STATUS    RESTARTS   AGE
trident-controller-79df798bdc-m79dc   6/6     Running   0          150d
trident-node-linux-xrst8              2/2     Running   0          150d
trident-operator-5574dbbc68-nthjv     1/1     Running   0          1m30s
```

4. 建立 `TridentOrchestrator` CR 以安裝 Trident 。

```
cat deploy/crds/tridentorchestrator_cr.yaml
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident

kubectl create -f deploy/crds/tridentorchestrator_cr.yaml

#Examine the pods in the Trident namespace
NAME                                READY   STATUS    RESTARTS   AGE
trident-csi-79df798bdc-m79dc        6/6     Running   0          1m
trident-csi-xrst8                   2/2     Running   0          1m
trident-operator-5574dbbc68-nthjv   1/1     Running   0          5m41s
```

5. 確認 Trident 已升級至所需版本。

```
kubectl describe torc trident | grep Message -A 3

Message:                Trident installed
Namespace:              trident
Status:                 Installed
Version:                v25.02.0
```

## 使用**tridentctl**進行升級

### 您可以使用輕鬆升級現有的 Trident 安裝 `tridentctl`。

關於這項工作

解除安裝及重新安裝 Trident 即為升級。當您解除安裝 Trident 時、不會刪除 Trident 部署所使用的持續 Volume Claim （ PVC ）和持續 Volume （ PV ）。Trident 離線時、已佈建的 PV 仍可繼續使用、而 Trident 會在恢復上線後、為在此期間建立的任何 PVC 配置磁碟區。

開始之前

使用升級前請先 `tridentctl`檢閱"選擇升級方法"。

步驟

1. 執行中的解除安裝命令 `tridentctl`、移除 CRD 和相關物件以外的所有與 Trident 相關的資源。

```
./tridentctl uninstall -n <namespace>
```

2. 重新安裝 Trident 。請參閱 "使用 tridentctl 安裝 Trident"。

請勿中斷升級程序。確保安裝程式執行完成。

# 使用 tridentctl 管理 Trident

`https://github.com/NetApp/trident/releases["Trident安裝程式套裝組合"^]包含 `tridentctl`命令列公用程式、可讓您輕鬆存取 `Trident` 。擁有足夠 `Privileges` 的 Kubernetes 使用者可以使用它來安裝 `Trident` 或管理包含 `Trident Pod` 的命名空間。

## 命令和全域旗標

您可以執行 `tridentctl help`以取得可用命令清單 `tridentctl`、或將旗標附加 `--help`至任何命令、以取得該特定命令的選項和旗標清單。

```
tridentctl [command] [--optional-flag]
```

Trident `tridentctl` 公用程式支援下列命令和全域旗標。

命令

**create**
將資源新增至 Trident 。

**delete**
從 Trident 移除一或多個資源。

**get**
從 Trident 取得一或多個資源。

**help**
任何命令的相關說明。

**images**
列印 Trident 所需的容器影像表格。

**import**
將現有資源匯入 Trident 。

**install**
安裝Trident。

**logs**
從 Trident 列印記錄。

**send**
從 Trident 傳送資源。

**uninstall**
解除安裝 Trident 。

**update**
在 Trident 中修改資源。

**update backend state**
暫時暫停後端作業。

**upgrade**
在 Trident 中升級資源。

**version**
列印 Trident 版本。

全域旗標

**-d、 --debug**

除錯輸出。

**-h、 --help**

的說明 `tridentctl`。

**-k、 --kubeconfig string**

指定 `KUBECONFIG`在本機或從一個 Kubernetes 叢集到另一個叢集執行命令的路徑。

或者、您也可以匯出 `KUBECONFIG`變數以指向特定的 Kubernetes 叢集、然後向該叢集發出 `tridentctl`命令。

**-n、 --namespace string**

Trident 部署的命名空間。

**-o、 --output string**

輸出格式。json之一| yaml| name| wl| ps（預設）。

**-s、 --server string**

Trident REST 介面的位址 / 連接埠。

Trident REST介面可設定為偵聽、僅適用於127.0.0.1（適用於IPV4）或[:1]（適用於IPv6）。

## 命令選項和旗標

建立

使用 `create`命令將資源新增至 Trident 。

```
tridentctl create [option]
```

選項

`backend`：將後端新增至 Trident 。

刪除

使用 `delete`命令從 Trident 中移除一或多個資源。

```
tridentctl delete [option]
```

選項

`backend`：從 Trident 刪除一個或多個儲存設備後端。
`snapshot`：從 Trident 刪除一個或多個 Volume 快照。
`storageclass`：從 Trident 刪除一個或多個儲存類別。

`volume`：從 Trident 刪除一個或多個儲存磁碟區。

取得

使用 `get`命令從 Trident 取得一或多個資源。

```
tridentctl get [option]
```

選項

`backend`：從 Trident 獲得一個或多個儲存設備後端。
`snapshot`：從 Trident 獲取一個或多個快照。
`storageclass`：從 Trident 獲取一個或多個儲存類。
`volume`：從 Trident 獲取一個或多個卷。

旗標

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

映像

使用 `images`旗標來列印 Trident 所需的容器影像表格。

```
tridentctl images [flags]
```

旗標

`-h、 --help`：影像說明。
`-v --k8s-version string`： Kubernetes 叢集的語義版本。

匯入**Volume**

使用 `import volume`命令將現有磁碟區匯入 Trident 。

```
tridentctl import volume <backendName> <volumeName> [flags]
```

別名

`volume、 v`

旗標

`-f --filename string`： YAML 或 JSON PVC 檔案路徑。
`-h --help`： Volume 說明。
`--no-manage`：僅建立 PV/PVC 。不要假設磁碟區生命週期管理。

安裝

使用 `install`旗標來安裝 Trident 。

```
tridentctl install [flags]
```

旗標

--autosupport-image string：AutoSupport 遙測的容器影像（預設為「 NetApp / Trident AutoSupport ： <current-version> 」）。
--autosupport-proxy string：用於發送 AutoSupport 遙測的代理的地址 / 端口。
--enable-node-prep：嘗試在節點上安裝所需的軟件包。
--generate-custom-yaml：生成 YAML 文件而無需安裝任何內容。
-h --help：安裝說明。
--http-request-timeout：覆寫 Trident 控制器 REST API 的 HTTP 要求逾時（預設值為 1m30s ）。
--image-registry string：內部映像登錄的位址 / 連接埠。
--k8s-timeout duration：所有 Kubernetes 作業的逾時時間（預設為 30 個月）。
--kubelet-dir string： kubelet 內部狀態的主機位置（預設為「 /var/lib/kubelet 」）。
--log-format string： Trident 記錄格式（ text 、 json ）（預設「 text 」（文字））。
--node-prep：可讓 Trident 準備 Kubernetes 叢集的節點、以使用指定的資料儲存傳輸協定來管理磁碟區。* 目前 iscsi`是唯一支援的值。 *
`--pv string： Trident 使用的舊版 PV 名稱、請確定不存在（預設為「 Trident 」）。
--pvc string： Trident 使用的傳統 PVC 名稱、請確定不存在（預設的「 Trident 」）。
--silence-autosupport：不要自動將 AutoSupport 套裝軟體傳送至 NetApp （預設為 true ）。
--silent：在安裝過程中禁用大多數輸出。
--trident-image string：要安裝的 Trident 映像。
--use-custom-yaml：使用安裝目錄中存在的任何現有 YAML 文件。
--use-ipv6：使用 IPv6 進行 Trident 通訊。

記錄

使用 `logs`旗標從 Trident 列印記錄。

```
tridentctl logs [flags]
```

旗標

-a，--archive：創建包含所有日誌的支持歸檔文件（除非另有指定）。
-h，--help：日誌幫助。
-l --log string：要顯示的 Trident 日誌。其中一個是 Trident | auto| Trident 運算子 | All （預設為「自動」）。
--node string：要從中收集節點 Pod 日誌的 Kubernetes 節點名稱。
-p --previous：獲取以前的 Container 實例的日誌（如果存在）。
--sidecars：獲取 sidecar 容器的日誌。

傳送

使用 `send`命令從 Trident 傳送資源。

```
tridentctl send [option]
```

選項

autosupport：將 AutoSupport 歸檔文件傳送至 NetApp 。

解除安裝

使用 `uninstall`旗標來解除安裝 Trident 。

```
tridentctl uninstall [flags]
```

旗標

-h, --help：卸載幫助。
--silent：在卸載過程中禁用大多數輸出。

更新

使用 `update`命令修改 Trident 中的資源。

```
tridentctl update [option]
```

選項

backend：在 Trident 中更新後端。

更新後端狀態

使用 `update backend state`命令暫停或恢復後端作業。

```
tridentctl update backend state <backend-name> [flag]
```

需要考量的重點

- 如果使用 TridentBackendConfig （ tbc ）建立後端、則無法使用檔案更新後端 backend.json 。
- 如果已在 tbc 中設定、則 userState 無法使用命令加以修改 tridentctl update backend state <backend-name> --user-state suspended/normal 。
- 若要在透過 tbc 設定 Via tridentctl 之後重新取得設定 userState`功能、 `userState`必須從 tbc 移除該欄位。這可以使用命令來完成 `kubectl edit tbc 。`userState`欄位移除後、您可以使用 `tridentctl update backend state`命令來變更 `userState`後端的。
- 使用 tridentctl update backend state 變更 userState。您也可以更新 userState 使用 TridentBackendConfig 或 backend.json 檔案、這會觸發後端的完整重新初始化、而且可能會耗費時間。

  旗標

  -h --help：後端狀態說明。
  --user-state：設為 suspended`暫停後端作業。設為 `normal`以恢復後端作業。設為時 `suspended：

- AddVolume 和 Import Volume 已暫停。
- CloneVolume、、 ResizeVolume PublishVolume UnPublishVolume CreateSnapshot、、 GetSnapshot RestoreSnapshot、、 DeleteSnapshot RemoveVolume GetVolumeExternal、、 ReconcileNodeAccess 保持可用狀態。

您也可以使用後端組態檔案或中的欄位來更新後端狀態 userState TridentBackendConfig backend.json。如需詳細資訊、請參閱 "管理後端的選項" 和 "以KECBECVL執行後端管理"。

範例：

**JSON**

請依照下列步驟使用檔案更新 `userState` `backend.json` ：

1. 編輯 `backend.json` 檔案、 `userState` 將欄位的值設為「已待定」。
2. 使用命令和更新檔案的路徑來更新後端 `tridentctl backend update` `backend.json` 。
   - 範例 * ： `tridentctl backend update -f /<path to backend JSON file>/backend.json`

```
{
  "version": 1,
  "storageDriverName": "ontap-nas",
  "managementLIF": "<redacted>",
  "svm": "nas-svm",
  "backendName": "customBackend",
  "username": "<redacted>",
  "password": "<redacted>",
  "userState": "suspended"
}
```

**YAML**

您可以在使用命令套用 tbc 之後編輯它 `kubectl edit <tbc-name> -n <namespace>` 。下列範例會使用選項更新後端狀態以暫停 `userState: suspended` ：

```
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-ontap-nas
spec:
  version: 1
  backendName: customBackend
  storageDriverName: ontap-nas
  managementLIF: <redacted>
  svm: nas-svm
  userState: suspended
  credentials:
    name: backend-tbc-ontap-nas-secret
```

版本

使用 `version`旗標來列印和執行中 `Trident` 服務的版本 `tridentctl`。

```
tridentctl version [flags]
```

旗標

`--client`：僅限用戶端版本（不需要伺服器）。
`-h, --help`：版本說明。

### 外掛程式支援

Tridentctl 支援類似 kubectl 的外掛程式。如果外掛程式二進位檔案名稱遵循「 <plugin> 」配置、則 Tridentctl 會偵測外掛程式、且二進位檔案位於列出 PATH 環境變數的資料夾中。所有偵測到的外掛程式都會列在 tridentctl 說明的外掛程式區段中。或者、您也可以在 enviornment 變數 TRIDENTCTL_plugin_path 中指定外掛程式資料夾來限制搜尋（例如： `TRIDENTCTL_PLUGIN_PATH=~/tridentctl-plugins/`）。如果使用此變數、則 tridenctl 只會在指定的資料夾中搜尋。

# 監控 Trident

Trident 提供一組 Prometheus 指標端點、可用於監控 Trident 效能。

### 總覽

Trident 提供的計量可讓您執行下列動作：

- 保留 Trident 健全狀況和組態的索引標籤。您可以檢查作業的成功程度、以及是否能如預期般與後端進行通訊。
- 檢查後端使用資訊、並瞭解後端上配置的磁碟區數量、以及所耗用的空間量等。
- 維護可用後端配置的磁碟區數量對應。
- 追蹤效能。您可以查看 Trident 與後端通訊和執行作業所需的時間。

根據預設、 Trident 的度量會公開在端點的 `/metrics`**目標連接埠上** `8001`。安裝Trident時*預設會啟用這些度量。

您需要的產品

- 安裝 Trident 的 Kubernetes 叢集。
- Prometheus執行個體。這可以是 "容器化Prometheus部署"或，您可以選擇將 Prometheus 作爲運行 "原生應用程式"。

### 步驟1：定義Prometheus目標

您應該定義 Prometheus 目標、以收集指標、並取得 Trident 管理的後端、建立的磁碟區等相關資訊。這 "部落格"將說明如何搭配 Trident 使用 Prometheus 和 Grafana 來擷取計量。部落格會說明如何在 Kubernetes 叢集中以營運者的身份執行 Prometheus 、以及如何建立 ServiceMonitor 來取得 Trident 指標。

### 步驟2：建立Prometheus ServiceMonitor

若要使用 Trident 度量、您應該建立 Prometheus ServiceMonitor 、以監控 `trident-csi`**服務並在連接埠上接聽** `metrics`。ServiceMonitor範例如下所示：

```
apiVersion: monitoring.coreos.com/v1
kind: ServiceMonitor
metadata:
  name: trident-sm
  namespace: monitoring
  labels:
    release: prom-operator
spec:
  jobLabel: trident
  selector:
    matchLabels:
      app: controller.csi.trident.netapp.io
  namespaceSelector:
    matchNames:
      - trident
  endpoints:
    - port: metrics
      interval: 15s
```

此 ServiceMonitor 定義會擷取服務傳回的度量 `trident-csi`、並特別尋找 `metrics`服務的端點。因此、Prometheus 現在已設定為瞭解 Trident 的指標。

除了可直接從 Trident 取得的指標之外、 kibelet 還會透過自己的指標端點來公開許多 `kubelet_volume_*`指標。Kubelet可提供有關所附加磁碟區、Pod及其處理的其他內部作業的資訊。請參閱 "請按這裡"。

## 步驟3：使用PromQL查詢Trident度量

PromQL適用於建立傳回時間序列或表格資料的運算式。

以下是一些您可以使用的PromQL查詢：

**取得Trident健全狀況資訊**

- 來自 **Trident** 的 **HTTP 2XX** 回應百分比

```
(sum (trident_rest_ops_seconds_total_count{status_code=~"2.."} OR on()
vector(0)) / sum (trident_rest_ops_seconds_total_count)) * 100
```

- **Trident** 透過狀態代碼的 **REST** 回應百分比

```
(sum (trident_rest_ops_seconds_total_count) by (status_code)  / scalar
(sum (trident_rest_ops_seconds_total_count))) * 100
```

- 由 **Trident** 執行之作業的平均持續時間（以毫秒為單位）

```
sum by (operation)
(trident_operation_duration_milliseconds_sum{success="true"}) / sum by
(operation)
(trident_operation_duration_milliseconds_count{success="true"})
```

取得 **Trident** 使用資訊

- 平均Volume大小*

```
trident_volume_allocated_bytes/trident_volume_count
```

- 每個後端配置的Volume空間總計*

```
sum (trident_volume_allocated_bytes) by (backend_uuid)
```

取得個別**Volume**使用量

只有同時收集kubelet度量時、才會啟用此功能。

- 每個Volume的已用空間百分比*

```
kubelet_volume_stats_used_bytes / kubelet_volume_stats_capacity_bytes *
100
```

## 瞭解 Trident AutoSupport 遙測

根據預設、 Trident 會在每日步調中、將 Prometheus 指標和基本後端資訊傳送至 NetApp 。

- 若要停止 Trident 傳送 Prometheus 度量和基本後端資訊至 NetApp 、請在 Trident 安裝期間傳遞 `--silence-autosupport`旗標。
- Trident 也可以透過將容器記錄傳送至 NetApp 隨選支援 `tridentctl send autosupport`。您需要觸發 Trident 來上傳其記錄檔。在提交日誌之前，您應該接受 NetApp 的https://www.netapp.com/company/legal/privacy-policy/["隱私權政策"^]。
- 除非另有說明、否則 Trident 會從過去 24 小時擷取記錄。
- 您可以使用旗標指定記錄保留時間範圍 `--since`。例如 `tridentctl send autosupport --since=1h`：。這些資訊會透過安裝在 Trident 旁邊的容器收集和傳送 `trident-autosupport`。您可以在上取得容器映像 "Trident AutoSupport 的"。
- Trident AutoSupport 無法收集或傳輸個人識別資訊（PII）或個人資訊。隨附 "EULA" 不適用於 Trident 容器映像本身的。您可以深入瞭解 NetApp 對資料安全與信任的承諾 "請按這裡"。

Trident 傳送的有效負載範例如下：

```
---
items:
  - backendUUID: ff3852e1-18a5-4df4-b2d3-f59f829627ed
    protocol: file
    config:
      version: 1
      storageDriverName: ontap-nas
      debug: false
      debugTraceFlags: null
      disableDelete: false
      serialNumbers:
        - nwkvzfanek_SN
      limitVolumeSize: ""
    state: online
    online: true
```

- 此資訊將傳送至NetApp的「不只是」端點。AutoSupport AutoSupport如果您使用私有登錄來儲存容器映像、則可以使用此 `--image-registry`旗標。
- 您也可以產生安裝Yaml檔案來設定Proxy URL。您可以使用來建立 YAML 檔案、並在中新增 `--proxy-url`容器的引數 `trident-autosupport`來 `trident-deployment.yaml`完成此 `tridentctl install --generate-custom-yaml`作業。

### 停用 Trident 計量

若要 停用報告的 度量、您應該產生自訂的 YAML （使用 `--generate-custom-yaml`旗標）、然後加以編輯、以移除 `--metrics`容器所叫用的旗標 `trident-main`。

# 解除安裝Trident

## 您應該使用與安裝 Trident 相同的方法來解除安裝 Trident 。

關於這項工作

- 如果您需要修正在升級、相依性問題或升級失敗或不完整之後所觀察到的錯誤，您應該解除安裝 Trident ，並使用該的特定指示重新安裝舊版"版本"。這是將 _ 降級 _ 降級至較早版本的唯一建議方法。
- 為了方便升級和重新安裝、解除安裝 Trident 並不會移除 Trident 所建立的 CRD 或相關物件。如果您需要完全移除 Trident 及其所有資料、請參閱"完全移除 Trident 和客戶需求日"。

開始之前

如果您要停用 Kubernetes 叢集、則必須先刪除所有使用 Trident 建立之 Volume 的應用程式、然後再解除安裝。如此可確保在刪除之前、不會在 Kubernetes 節點上發佈 PVC 。

### 確定原始安裝方法

您應該使用與安裝相同的方法來解除安裝 Trident 。在解除安裝之前、請先確認您原本安裝 Trident 的版本。

1. 用於 `kubectl get pods -n trident`檢查 Pod 。
   - 如果沒有運算子 Pod 、則使用安裝 Trident `tridentctl` 。
   - 如果有操作員 Pod 、則 Trident 是使用 Trident 操作員手動或使用 Helm 來安裝。
2. 如果有操作員 Pod 、請使用 `kubectl describe tproc trident`判斷是否使用 Helm 安裝 Trident 。
   - 如果有 Helm 標籤、則使用 Helm 安裝 Trident 。
   - 如果沒有 Helm 標籤、則會使用 Trident 操作員手動安裝 Trident 。

## 解除安裝 **Trident** 運算子安裝

您可以手動或使用 Helm 解除安裝 Trident 運算子安裝。

### 解除安裝手動安裝

如果您使用運算子安裝 Trident 、則可以執行下列其中一項動作來解除安裝：

1. 編輯 **`TridentOrchestrator` CR** 並設定解除安裝旗標 ：

   ```
   kubectl patch torc <trident-orchestrator-name> --type=merge -p
   '{"spec":{"uninstall":true}}'
   ```

   當 `uninstall`旗標設定為 `true`時、 Trident 操作員會解除安裝 Trident 、但不會移除 TridentOrchestrator 本身。如果您想要再次安裝Trident、請清理TridentOrchestrator並建立新的Trident。

2. 刪除 **`TridentOrchestrator`** ：移除用於部署 Trident 的 CR 後 `TridentOrchestrator`、您會指示操作員解除安裝 Trident 。操作員會處理移除並繼續移除 Trident 部署和取消程式集、刪除其在安裝過程`TridentOrchestrator`中所建立的 Trident Pod 。

   ```
   kubectl delete -f deploy/<bundle.yaml> -n <namespace>
   ```

### 解除安裝 **Helm** 安裝

如果您使用 Helm 安裝 Trident 、可以使用解除安裝 `helm uninstall`。

```
#List the Helm release corresponding to the Trident install.
helm ls -n trident
NAME          NAMESPACE       REVISION        UPDATED
STATUS          CHART                           APP VERSION
trident       trident         1               2021-04-20
00:26:42.417764794 +0000 UTC deployed        trident-operator-21.07.1
21.07.1

#Uninstall Helm release to remove Trident
helm uninstall trident -n trident
release "trident" uninstalled
```

## 解除安裝 `tridentctl`

使用 `uninstall`中的命令 `tridentctl`移除與 Trident 相關的所有資源、但 CRD 和相關物件除外：

```
./tridentctl uninstall -n <namespace>
```