



管理和監控 **Astra Trident**

Astra Trident

NetApp
April 18, 2024

目錄

管理和監控 Astra Trident	1
升級Astra Trident	1
使用 Tridentctl 管理 Astra Trident	7
監控Astra Trident	12
解除安裝Astra Trident	16

管理和監控 Astra Trident

升級Astra Trident

升級Astra Trident

Astra Trident依照每季發行時段進行、每一日曆年發行四個主要版本。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。我們鼓勵您每年至少升級一次、以善用Astra Trident的新功能。

升級前的考量

升級至最新版Astra Trident時、請考慮下列事項：

- 在指定 Kubernetes 叢集中的所有命名空間中、應該只安裝一個 Astra Trident 執行個體。
- Astra 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](#)"、您必須更新後端組態才能使用 `standardsw` 或 `zoneredundantstandardsw` 從 Astra Trident 23.01 升級時的服務層級。無法更新 `serviceLevel` 在後端中、可能會導致磁碟區故障。請參閱 "[CVS 服務類型範例](#)" 以取得詳細資料。
- 升級時、請務必提供 `parameter.fsType` 在中 `StorageClasses` 由Astra Trident使用。您可以刪除並重新建立 `StorageClasses` 無需中斷既有的磁碟區。
 - 這是強制實施的一項**要求 "[安全性內容](#)" 適用於SAN磁碟區。
 - `sample INPUT` 目錄包含 <https://github.com/NetApp/trident/blob/master/trident-installer/sample-input/storage-class-samples/storage-class-basic.yaml.template> 等範例[`storage-class-basic.yaml.template`] 和連結：`storage-class-bronze-default.yaml`。
 - 如需詳細資訊、請參閱 "[已知問題](#)"。

步驟 1：選取版本

Astra Trident版本遵循日期型 `YY.MM` 命名慣例、其中「是」是一年的最後兩位數、「公釐」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 23.01（或任何 23.01 點版本）升級至 24.02。
- 如果您要從四個版本的外部版本升級、請執行多步驟升級。請使用的升級指示 "[舊版](#)" 您要從升級至最新版本、以符合四個版本的時間範圍。例如、如果您執行 22.01 且想要升級至 24.02：
 - a. 第一次從 22.01 升級至 23.01。
 - b. 然後從 23.01 升級至 24.02。



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、請參閱 "[GitHub問題詳細資料](#)"。

步驟 2：確定原始安裝方法

若要判斷您原本用來安裝 Astra Trident 的版本：

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

步驟 3：選擇升級方法

一般而言、您應該使用與初始安裝相同的方法進行升級、不過您可以 ["在安裝方法之間移動"](#)。有兩種方法可以升級 Astra Trident。

- ["使用Trident營運者進行升級"](#)



我們建議您檢閱 ["瞭解營運商升級工作流程"](#) 與操作員一起升級之前。

*

與營運者一起升級

瞭解營運商升級工作流程

在使用 Trident 運算子升級 Astra Trident 之前、您應該先瞭解升級期間發生的背景程序。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

其中一項 ["使用 Trident 運算子的優點"](#) 安裝和升級 Astra Trident 是自動處理 Astra Trident 和 Kubernetes 物件、而不會中斷現有的掛載磁碟區。如此一來、Astra 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。



如需 Kubernetes 叢集上 Astra Trident 架構的詳細資訊、請參閱 ["Astra Trident 架構"](#)。

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 Astra 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 。

升級 Trident 營運商安裝

您可以使用 Trident 運算子手動或使用 Helm 來升級 Astra Trident 。您可以從 Trident 運算子安裝升級至其他 Trident 運算子安裝、或從升級 tridentctl 安裝至 Trident 運算子版本。檢閱 ["選擇升級方法"](#) 在升級 Trident 操作員安裝之前。

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 Astra 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 Astra Trident 進行升級、請使用的升級指示 ["您已安裝的版本"](#) Astra Trident 的

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。

- 對於執行 Kubernetes 1.24 或更早版本的叢集、請使用 ["bunder_pre_1_25.yaml"](#) 。
- 對於執行 Kubernetes 1.25 或更新版本的叢集、請使用 ["bunder_POST_1_25.yaml"](#) 。

開始之前

確保您使用的是執行中的 Kubernetes 叢集 ["支援的Kubernetes版本"](#) 。

步驟

1. 驗證 Astra Trident 版本：

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

2. 刪除用來安裝目前 Astra 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. 如果您使用自訂初始安裝 `TridentOrchestrator` 屬性、您可以編輯 `TridentOrchestrator` 物件以修改安裝參數。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（其中包含 `<bundle.yaml>`）來安裝 Astra Trident `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 Astra Trident 24.02、請執行下列命令：

```
kubectl create -f 24.02.0/trident-installer/deploy/<bundle.yaml> -n
trident
```

升級 Helm 安裝

您可以升級 Astra Trident Helm 安裝。



將Kubernetes叢集從1.24升級至1.25或更新版本、且已安裝Astra Trident時、您必須更新`vales.yaml`才能設定 `excludePodSecurityPolicy` 至 `true` 或新增 `--set excludePodSecurityPolicy=true` 至 `helm upgrade` 命令、然後才能升級叢集。

步驟

1. 如果您 "使用 Helm 安裝 Astra Trident"、您可以使用 `helm upgrade trident netapp-trident/trident-operator --version 100.2402.0` 只需一步即可升級。如果您未新增 Helm repo 或無法使用它來升級：
 - a. 從下載最新的 Astra Trident 版本 "[GitHub的_Assets區段](#)"。
 - b. 使用 `helm upgrade` 命令位置 `trident-operator-24.02.0.tgz` 反映您要升級的版本。

```
helm upgrade <name> trident-operator-24.02.0.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）、請附加 `helm upgrade` 命令使用 `--set` 為了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。

從升級 `tridentctl` 安裝至 **Trident** 操作員

您可以從升級至最新版的Trident運算子 `tridentctl` 安裝：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Astra Trident版本。

```
# Download the release required [24.020.0]
mkdir 24.02.0
cd 24.02.0
wget
https://github.com/NetApp/trident/releases/download/v24.02.0/trident-
installer-24.02.0.tar.gz
tar -xf trident-installer-24.02.0.tar.gz
cd trident-installer
```

2. 從資訊清單建立「TridentOrchestrator」CRD。

```
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」、以安裝Astra 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:          v24.02.0
```

使用tridentctl進行升級

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

關於這項工作

解除安裝和重新安裝Astra Trident可做為升級。當您解除安裝Trident時、不會刪除由Astra Trident部署所使用的持續磁碟區宣告 (PVC) 和持續磁碟區 (PV)。當Astra Trident離線時、已配置的PV仍可繼續使用、而Astra Trident會在任何建立於過渡期間的永久虛電路恢復上線後、為其配置磁碟區。

開始之前

檢閱 ["選擇升級方法"](#) 使用升級之前 tridentctl。

步驟

1. 在中執行解除安裝命令 tridentctl 移除與 Astra Trident 相關的所有資源、但 CRD 和相關物件除外。

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


2. 重新安裝 Astra Trident 。請參閱 ["使用tridentctl安裝Astra Trident"](#) 。



請勿中斷升級程序。確保安裝程式執行完成。

使用 **Tridentctl** 管理 **Astra Trident**

。 ["Trident安裝程式套裝組合"](#) 包括 `tridentctl` 命令列公用程式、提供 Astra Trident 的簡單存取。擁有足夠權限的 Kubernetes 使用者可以使用它來安裝 Astra Trident 或管理內含 Astra Trident Pod 的命名空間。

命令和全域旗標

您可以執行 `tridentctl help` 取得的可用命令清單 `tridentctl` 或附加 `--help` 標記至任何命令、以取得該特定命令的選項和旗標清單。

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

Astra Trident `tridentctl` 公用程式支援下列命令和全域旗標。

create

新增資源至 Astra Trident 。

delete

從 Astra Trident 移除一或多個資源。

get

從 Astra Trident 取得一或多個資源。

help

任何命令的相關說明。

images

列印 Astra Trident 所需的容器影像表格。

import

將現有資源匯入 Astra Trident 。

install

安裝Astra Trident 。

logs

列印 Astra Trident 的記錄。

send

從 Astra Trident 傳送資源。

解除安裝

解除安裝 Astra Trident 。

update

修改 Astra Trident 中的資源。

update backend state

暫時暫停後端作業。

upgrade

升級 Astra Trident 中的資源。

「分度」

列印 Astra Trident 的版本。

-d、--debug

除錯輸出。

-h、--help

的說明 tridentctl。

-k、--kubeconfig string

指定 KUBECONFIG 從本機或從一個 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 可指向特定 Kubernetes 叢集和問題的變數 tridentctl 命令到該叢集。

-n、--namespace string

Astra Trident 部署的命名空間。

-o、--output string

輸出格式。json之一|yaml|name|w|ps（預設）。

-s、--server string

Astra 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 命令以新增資源至Astra Trident。

```
tridentctl create [option]
```

選項

backend：將後端新增至Astra Trident。

刪除

使用 delete 從Astra Trident移除一或多個資源的命令。

```
tridentctl delete [option]
```

選項

backend：從Astra Trident刪除一個或多個儲存後端。
snapshot：從Astra Trident刪除一個或多個Volume快照。

storageclass：從Astra Trident刪除一個或多個儲存類別。
volume：從Astra Trident刪除一個或多個儲存磁碟區。

取得

使用 `get` 從Astra Trident取得一或多個資源的命令。

```
tridentctl get [option]
```

選項

backend：從Astra Trident取得一或多個儲存後端。
snapshot：從Astra Trident取得一或多個快照。
storageclass：從Astra Trident取得一或多個儲存課程。
volume：從Astra Trident取得一或多個磁碟區。

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。

映像

使用 `images` 用於列印 Astra Trident 所需容器影像表格的旗標。

```
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` 將現有磁碟區匯入Astra Trident的命令。

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

別名

volume、v

旗標

-f、--filename string：Yaml或Json PVC檔案的路徑。
-h、--help：Volume的說明。
--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` 安裝Astra Trident的旗標。

```
tridentctl install [flags]
```

旗標

- autosupport-image string: AutoSupport 遙測的容器映像（預設為「NetApp/trident autosupport : <current-version>」）。
- autosupport-proxy string: 代理伺服器的位址/連接埠、用於傳送AutoSupport 「遙測」功能。
- enable-node-prep: 嘗試在節點上安裝所需的套件。
- generate-custom-yaml: 在不安裝任何內容的情況下生成Yaml文件。
- 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3個月）。
- kubelet-dir string: Kubelet內部狀態的主機位置（預設為「/var/lib/kubelet」）。
- log-format string: Astra Trident記錄格式（text、json）（預設「text」）。
- pv string: Astra Trident使用的舊PV名稱、確保不存在（預設為「Trident」）。
- pvc string: Astra Trident使用的舊版永久虛擬卷名稱、確保不存在（預設為「Trident」）。
- silence-autosupport: 請勿AutoSupport 自動將不實的套裝組合傳送至NetApp（預設為true）。
- silent: 安裝期間禁用大多數輸出。
- trident-image string: 要安裝的Astra Trident映像。
- use-custom-yaml: 使用安裝目錄中現有的任何Yaml檔案。
- use-ipv6: 使用IPv6進行Astra Trident的通訊。

記錄

使用 logs 用於列印Astra Trident記錄的旗標。

```
tridentctl logs [flags]
```

旗標

- a、--archive: 除非另有說明、否則請使用所有記錄建立支援歸檔。
- h、--help: 日誌幫助。
- l、--log string: 要顯示的Astra Trident記錄。其中一個trident | auto | trident運算子 | all（預設為「自動」）。
- node string: Kubernetes節點名稱、用於收集節點Pod記錄。
- p、--previous: 獲取先前容器實例的日誌（如果存在）。
- sidecars: 取得邊側邊容器的記錄。

傳送

使用 send 從Astra Trident傳送資源的命令。

```
tridentctl send [option]
```

選項

- autosupport: 將AutoSupport 一份不適用的歸檔文件傳送給NetApp。

解除安裝

使用 uninstall 解除安裝Astra Trident的旗標。

```
tridentctl uninstall [flags]
```

旗標

- h, --help：解除安裝說明。
- silent：卸載期間禁用大多數輸出。

更新

使用 `update` 命令以修改 Astra Trident 中的資源。

```
tridentctl update [option]
```

選項

- backend：更新Astra Trident的後端。

更新後端狀態

使用 `update backend state` 暫停或恢復後端作業的命令。

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

旗標

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

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

版本

使用 `version` 用於列印版本的旗標 `tridentctl` 以及執行中的Trident服務。

```
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。

監控Astra Trident

Astra Trident 提供一組 Prometheus 指標端點、可用來監控 Astra Trident 的效能。

總覽

Astra Trident提供的指標可讓您執行下列作業：

- 隨時掌握Astra Trident的健全狀況與組態。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Astra Trident與後端及執行作業所需的時間。



根據預設、Trident的指標會顯示在「/度量」端點的目標連接埠「8001」上。安裝Trident時*預設會啟用這些度量。

您需要的產品

- 安裝Astra Trident的Kubernetes叢集。
- Prometheus執行個體。這可以是 "[容器化Prometheus部署](#)" 或者、您也可以選擇以執行Prometheus "[原生應用程式](#)"。

步驟1：定義Prometheus目標

您應該定義Prometheus目標、以收集指標並取得有關後端Astra Trident管理的資訊、以及其建立的磁碟區等資訊。這 "[部落格](#)" 說明如何使用Prometheus和Grafana搭配Astra Trident來擷取指標。部落格說明如何在Kubernetes 叢集中以運算符的形式執行 Prometheus 、以及如何建立 ServiceMonitor 來取得 Astra Trident 指標。

步驟2：建立Prometheus ServiceMonitor

若要使用Trident指標、您應該建立Prometheus ServiceMonitor、以監控「Trident - csi」服務、並在「metrics」連接埠上聆聽。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定義會擷取由「Trident -csi」服務傳回的度量、並特別尋找服務的「metrics」端點。因

此、Prometheus現在已設定為瞭解Astra Trident的指標。

除了直接從Astra Trident取得的指標之外、Kubelet也公開了許多指標 `kubelet_volume_*` 透過IT本身的指標端點來建立指標。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Astra 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
```

- 透過狀態代碼*來自Astra Trident的休息回應百分比

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

- 由Astra Trident執行的平均營運持續時間

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

取得Astra 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
```

深入瞭解Astra Trident AutoSupport 遙測技術

依預設、Astra Trident會每日傳送Prometheus指標和基本後端資訊給NetApp。

- 若要停止Astra Trident將Prometheus指標和基本後端資訊傳送給NetApp、請在Astra Trident安裝期間、傳遞「-Silence-autosupport」旗標。
- Astra Trident也可透過「tridentctl傳送AutoSupport 功能」、隨需將容器記錄傳送給NetApp Support。您需要觸發Astra Trident來上傳記錄。在您提交記錄之前、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、Astra Trident會從過去24小時擷取記錄。
- 您可以使用指定記錄保留時間範圍 --since 旗標。例如：tridentctl send autosupport --since=1h。此資訊會透過收集和傳送 trident-autosupport 安裝於Astra Trident旁的容器。您可以從取得Container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a "EULA" 這不適用於Trident Container映像本身。您可以深入瞭解NetApp對資料安全性與信任的承諾 ["請按這裡"](#)。

Astra 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:
    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、可以使用「-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「tridentctl install -generate-custom-yaml」來建立Yaml檔案、並在「trident部署.yaml」中新增「trident-autosupport」容器的「-proxy-URL」引數。

停用Astra Trident度量

要使指標不被報告，您應該生成自定義YAML（使用"-generate-custom-yaml"標誌）並進行編輯，以刪除對"trident-main"容器所調用的"-metrics"標誌。

解除安裝Astra Trident

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

關於這項工作

- 如果您需要修正升級、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 Astra Trident 並使用相關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 Astra Trident 並不會移除 Astra Trident 所建立的 CRD 或相關物件。如果您需要完全移除 Astra Trident 及其所有資料、請參閱 ["完全移除 Astra Trident 和 CRD"](#)。

開始之前

如果您要停用 Kubernetes 叢集、則必須先刪除所有使用 Astra Trident 所建立之 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 Astra Trident 相同的方法來解除安裝 Astra Trident 。在解除安裝之前、請先確認您原本用來安裝 Astra Trident 的版本。

1. 使用 `kubectl get pods -n trident` 檢查 Pod 。
- 如果沒有操作員 Pod 、則使用安裝 Astra Trident `tridentctl` 。
- 如果有操作員 Pod 、則使用 Trident 操作員手動或使用 Helm 來安裝 Astra Trident 。
2. 如果有操作員 Pod 、請使用 `kubectl describe tproc trident` 判斷 Astra Trident 是否使用 Helm 安裝。
- 如果有 Helm 標籤、則使用 Helm 安裝 Astra Trident 。
- 如果沒有 Helm 標籤、則使用 Trident 運算子手動安裝 Astra 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 Astra Trident 、您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`、Trident 運算子會卸載 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 TridentOrchestrator 用來部署 Astra Trident 的 CR、您可以指示操作員解除安裝 Trident。操作員會處理的移除作業 TridentOrchestrator 然後繼續移除 Astra Trident 部署和取消安裝、刪除它在安裝過程中建立的 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 Astra Trident、您可以使用「`helm uninstall`」來解除安裝。

```
#List the Helm release corresponding to the Astra 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 安裝

使用 `uninstall` 命令輸入 `tridentctl` 若要移除與 Astra Trident 相關的所有資源、但 CRD 和相關物件除外：

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

版權資訊

Copyright © 2024 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。