使用 SnapMirror 複寫磁碟區

Astra Trident

NetApp
July 18, 2025

目錄

使用 SnapMirror 複寫磁碟區

使用 Astra Control Provisioner 、您可以在一個叢集上的來源磁碟區和對等叢集上的目的地磁碟區之間建立鏡射關係、以便複寫資料以進行災難恢復。您可以使用命名的自訂資源定義（ CRD ）來執行下列作業：

- 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**

- * Astra Control Provisioner* ： Astra Control Provisioner 版本 23.10 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" 。

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。

- **Astra Control Provisioner** 和 **SVM** ：對等的遠端 SVM 必須可供目的地叢集上的 Astra Control Provisioner 使用。

支援的驅動程式

- ONTAP NAS 和 ONTAP SAN 驅動程式支援 Volume 複寫。

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
```

Astra Control Provisioner 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"
```

Astra Control Provisioner 將使用設定的關係原則名稱（或 ONTAP 的預設名稱）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
```

Astra Control Provisioner 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、 Astra Control Provisioner 將確保新的 FlexVol 磁碟區放置在與 MirrorRelationship 中定義的遠端 SVM 對等的 SVM 上。

Volume 複寫狀態

Trident 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 Astra Control Provisioner 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 *TridentMirrorRelationationship _ 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

Astra Control Provisioner 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）

Astra Control Provisioner 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**

在鏡射關係的一側刪除 TMR 會導致其餘 TMR 在 Astra Control Provisioner 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 _ 升級 _ 狀態、則沒有現有的鏡射關係、 *TMR* 將會移除、 *Astra Control Provisioner* 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 Astra Control 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。

啟用 Astra Control Provisioner

Trident 版本 23.10 及更新版本包含使用 Astra Control Provisioner 的選項，可讓獲授權的 Astra Control 使用者存取進階儲存資源配置功能。Astra Control Provisioner 除了提供標準 Astra Trident CSI 型功能之外、還提供這項延伸功能。您可以使用此程序來啟用和安裝 Astra Control Provisioner 。

您的 Astra Control Service 訂閱會自動包含 Astra Control Provisioner 使用授權。

Astra Control 的更新將取代 Astra Trident 做為儲存資源配置程式和 Orchestrator 、並成為 Astra Control 使用的必要項目。因此、強烈建議 Astra Control 使用者啟用 Astra Control Provisioner 。Astra Trident 將繼續保持開放原始碼、並以 NetApp 的新 CSI 和其他功能來發行、維護、支援及更新。

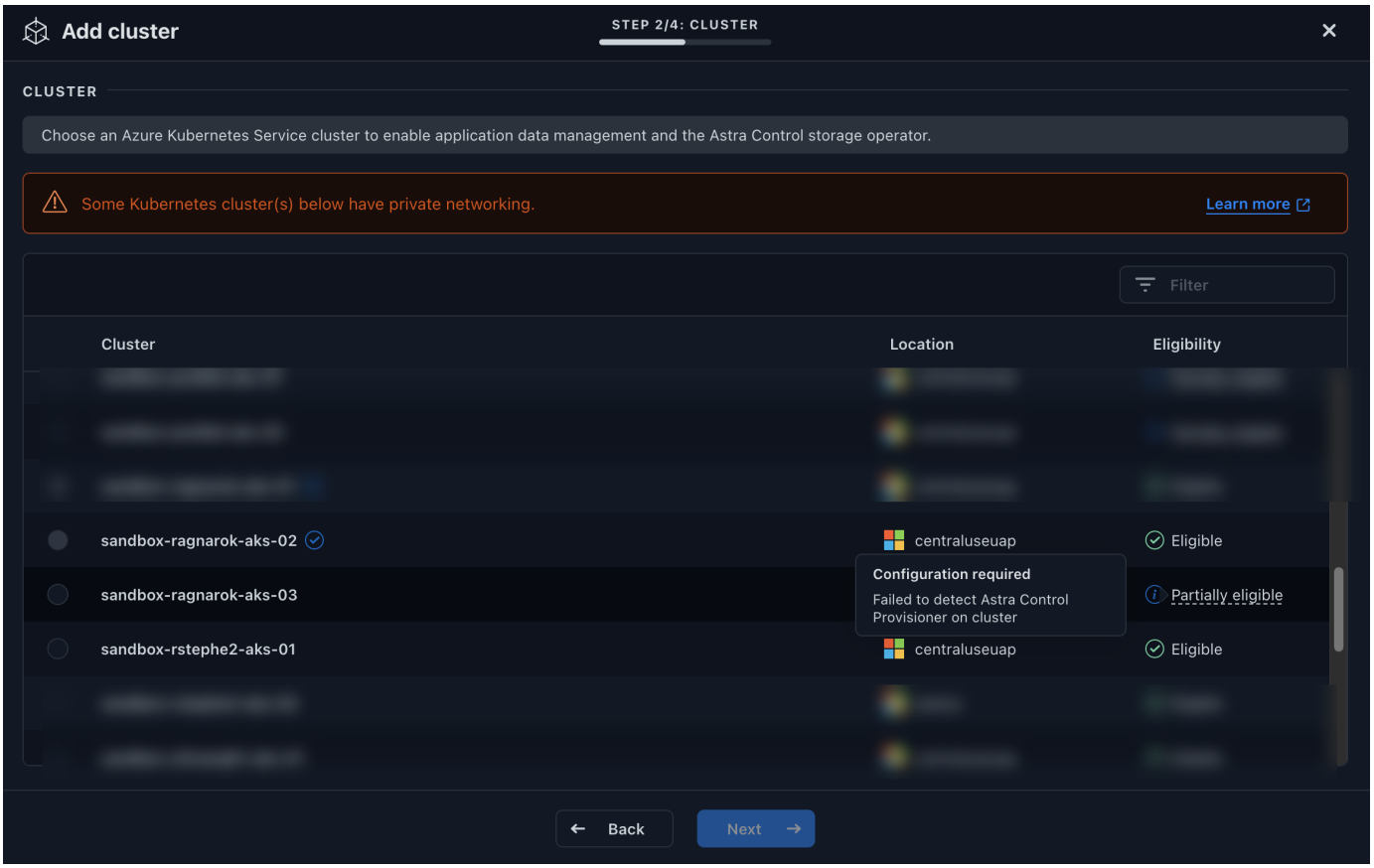
如何知道我是否需要啟用 **Astra Control Provisioner** ？

如果您將叢集新增至先前未安裝 Astra Trident 的 Astra Control Service 、叢集將會標示為 `Eligible`。您之後 "將叢集新增至 Astra Control"、 Astra Control Provisioner 將會自動啟用。

如果您的叢集未標記、則 `Eligible`會標記為 `Partially eligible` 下列其中一項：

- 它使用的是舊版 Astra Trident
- 它使用的 Astra Trident 23.10 尚未啟用置備程式選項
- 這是一種不允許自動啟用的叢集類型

在 `Partially eligible` 案例中、請使用這些指示來手動為叢集啟用 Astra Control Provisioner 。



啟用 **Astra Control Provisioner** 之前

如果您現有的 Astra Trident 不含 Astra Control Provisioner 、而且想要啟用 Astra Control Provisioner 、請先執行下列步驟：

- * 如果您已安裝 Astra Trident 、請確認其版本位於四個版本的視窗 * ：如果 Astra Trident 位於 24.02 版的四個版本視窗內、您可以使用 Astra Control Provisioner 直接升級至 Astra Trident 24.02 。例如、您可以直接從 Astra Trident 23.04 升級至 24.02 。
- * 確認您的叢集具有 AMD64 系統架構 * ： Astra Control Provisioner 映像同時在 AMD64 和 ARM64 CPU 架構中提供、但 Astra Control 僅支援 AMD64 。

步驟

1. 存取 NetApp Astra Control 影像登錄：
   a. 登入 Astra Control Service UI 並記錄 Astra Control 帳戶 ID 。
      i. 選取頁面右上角的圖示。
      ii. 選擇* API存取*。
      iii. 記下您的帳戶 ID 。
   b. 從同一頁面選取 * 產生 API 權杖 * 、然後將 API 權杖字串複製到剪貼簿、並將其儲存在編輯器中。
   c. 使用您偏好的方法登入 Astra Control 登錄：

```
docker login cr.astra.netapp.io -u <account-id> -p <api-token>
```

```
crane auth login cr.astra.netapp.io -u <account-id> -p <api-token>
```

2. （僅限自訂登錄）請依照下列步驟將映像移至自訂登錄。如果您不使用登錄，請遵循中的 Trident 運算子步驟 下一節。

> 您可以使用 Podman 而非 Docker 來執行下列命令。如果您使用的是 Windows 環境、建議您使用 PowerShell 。

Docker

a. 從登錄中拉出 Astra Control Provisioner 映像：

> 所擷取的映像不支援多個平台、而且僅支援與擷取映像的主機相同的平台、例如 Linux AMD64 。

```
docker pull cr.astra.netapp.io/astra/trident-acp:24.02.0
--platform <cluster platform>
```

範例：

```
docker pull cr.astra.netapp.io/astra/trident-acp:24.02.0
--platform linux/amd64
```

b. 標記影像：

```
docker tag cr.astra.netapp.io/astra/trident-acp:24.02.0
<my_custom_registry>/trident-acp:24.02.0
```

c. 將映像推送至自訂登錄：

```
docker push <my_custom_registry>/trident-acp:24.02.0
```

起重機

a. 將 Astra Control Provisioner 資訊清單複製到您的自訂登錄：

```
crane copy cr.astra.netapp.io/astra/trident-acp:24.02.0
<my_custom_registry>/trident-acp:24.02.0
```

3. 確定原始 Astra Trident 安裝方法是否使用。
4. 使用您最初使用的安裝方法、在 Astra Trident 中啟用 Astra Control Provisioner ：

Astra Trident 運算子

a. "下載 Astra Trident 安裝程式並將其解壓縮"。

b. 如果您尚未安裝 Astra Trident 、或是從原始 Astra Trident 部署中移除運算子、請完成下列步驟：

   i. 建立客戶需求日：

```
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.1
6.yaml
```

   ii. 創建 trident 命名空間 (`kubectl create namespace trident`）或確認 trident 命名空間仍然存在 (`kubectl get all -n trident`）。如果已移除命名空間、請重新建立。

c. 將 Astra Trident 更新為 24.02.0 ：

> 對於運行 Kubernetes 1.24 或更早版本的羣集，請使用 `bundle_pre_1_25.yaml`。對於運行 Kubernetes 1.25 或更高版本的羣集，請使用 `bundle_post_1_25.yaml`。

```
kubectl -n trident apply -f trident-installer/deploy/<bundle-
name.yaml>
```

d. 確認 Astra Trident 正在執行：

```
kubectl get torc -n trident
```

回應：

```
NAME      AGE
trident   21m
```

e. [[Pull 機密]] 如果您有使用機密的登錄、請建立秘密來拉出 Astra Control Provisioner 映像：

```
kubectl create secret docker-registry <secret_name> -n trident
--docker-server=<my_custom_registry> --docker-username=<username>
--docker-password=<token>
```

f. 編輯 TridentOrchestrator CR 並進行下列編輯：

```
kubectl edit torc trident -n trident
```

i. 設定 Astra Trident 映像的自訂登錄位置、或從 Astra Control 登錄或中拉出 (`tridentImage: <my_custom_registry>/trident:24.02.0 tridentImage: netapp/trident:24.02.0`) 。

ii. 啓用 Astra Control Provisioner (`enableACP: true`) 。

iii. 設定 Astra Control Provisioner 映像的自訂登錄位置、或從 Astra Control 登錄或將其拉出 (`acpImage: <my_custom_registry>/trident-acp:24.02.0 acpImage: cr.astra.netapp.io/astra/trident-acp:24.02.0`) 。

iv. 如果您先前在此程序中建立 影像拉出秘密 、您可以在此設定 (`imagePullSecrets: - <secret_name>`) 。請使用您在先前步驟中建立的相同名稱密碼名稱。

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  tridentImage: <registry>/trident:24.02.0
  enableACP: true
  acpImage: <registry>/trident-acp:24.02.0
  imagePullSecrets:
  - <secret_name>
```

g. 儲存並結束檔案。部署程序將會自動開始。

h. 確認已建立運算子、部署和複本集。

```
kubectl get all -n trident
```

Kubernetes叢集中只應有*一個運算子執行個體*。請勿建立 Astra Trident 運算子的多個部署。

i. 驗證該 `trident-acp` 容器是否正在運行且 `acpVersion` `24.02.0` 狀態爲 `Installed`：

```
kubectl get torc -o yaml
```

回應：

```
status:
  acpVersion: 24.02.0
  currentInstallationParams:
    ...
    acpImage: <registry>/trident-acp:24.02.0
    enableACP: "true"
    ...
  ...
  status: Installed
```

試用

a. "下載 Astra Trident 安裝程式並將其解壓縮"。

b. "如果您有現有的 Astra Trident 、請將其從裝載它的叢集上解除安裝"。

c. 安裝 Astra Trident 並啟用 Astra Control Provisioner (`--enable-acp=true`) ：

```
./tridentctl -n trident install --enable-acp=true --acp
-image=mycustomregistry/trident-acp:24.02
```

d. 確認 Astra Control Provisioner 已啟用：

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

回應：

```
+----------------+----------------+-------------+ | SERVER
VERSION | CLIENT VERSION | ACP VERSION | +----------------
+----------------+-------------+ | 24.02.0 | 24.02.0 | 24.02.0. |
+----------------+----------------+-------------+
```

掌舵

a. 如果您已安裝 Astra Trident 23.07.1 或更早版本、則 "解除安裝" 為操作員和其他元件。

b. 如果 Kubernetes 叢集執行 1.24 或更早版本、請刪除 PSP ：

```
kubectl delete psp tridentoperatorpod
```

c. 新增Astra Trident Helm儲存庫：

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

d. 更新 Helm 圖表：

```
helm repo update netapp-trident
```

回應：

```
Hang tight while we grab the latest from your chart
repositories...
...Successfully got an update from the "netapp-trident" chart
repository
Update Complete. □Happy Helming!□
```

e. 列出影像：

```
./tridentctl images -n trident
```

回應：

```
| v1.28.0            | netapp/trident:24.02.0|
|                    | docker.io/netapp/trident-
autosupport:24.02|
|                    | registry.k8s.io/sig-storage/csi-
provisioner:v4.0.0|
|                    | registry.k8s.io/sig-storage/csi-
attacher:v4.5.0|
|                    | registry.k8s.io/sig-storage/csi-
resizer:v1.9.3|
|                    | registry.k8s.io/sig-storage/csi-
snapshotter:v6.3.3|
|                    | registry.k8s.io/sig-storage/csi-node-
driver-registrar:v2.10.0 |
|                    | netapp/trident-operator:24.02.0 (optional)
```

f. 確保 Trident 操作員 24.02.0 可用：

```
helm search repo netapp-trident/trident-operator --versions
```

回應：

```
NAME                                CHART VERSION   APP VERSION
DESCRIPTION
netapp-trident/trident-operator 100.2402.0      24.02.0         A
```

g. 使用 `helm install` 並執行下列其中一個選項、其中包括這些設定：

- 部署位置的名稱
- Astra Trident 版本
- Astra Control Provisioner 映像的名稱
- 啟用資源配置程式的旗標
- （選用）本機登錄路徑。如果您使用的是本機登錄，則 "Trident 影像" 可以位於一個登錄或不同的登錄中，但所有 CSI 映像都必須位於相同的登錄中。
- Trident 命名空間

選項

。沒有登錄的映像

```
helm install trident netapp-trident/trident-operator --version
100.2402.0 --set acpImage=cr.astra.netapp.io/astra/trident-
acp:24.02.0 --set enableACP=true --set operatorImage=netapp/trident-
operator:24.02.0 --set
tridentAutosupportImage=docker.io/netapp/trident-autosupport:24.02
--set tridentImage=netapp/trident:24.02.0 --namespace trident
```

。一個或多個登錄中的影像

```
helm install trident netapp-trident/trident-operator --version
100.2402.0 --set acpImage=<your-registry>:<acp image> --set
enableACP=true --set imageRegistry=<your-registry>/sig-storage --set
operatorImage=netapp/trident-operator:24.02.0 --set
tridentAutosupportImage=docker.io/netapp/trident-autosupport:24.02
--set tridentImage=netapp/trident:24.02.0 --namespace trident
```

您可以使用 `helm list` 若要檢閱安裝詳細資料、例如名稱、命名空間、圖表、狀態、應用程式版本、和修訂編號。

如果您在使用 Helm 部署 Trident 時遇到任何問題、請執行此命令以完全解除安裝 Astra Trident ：

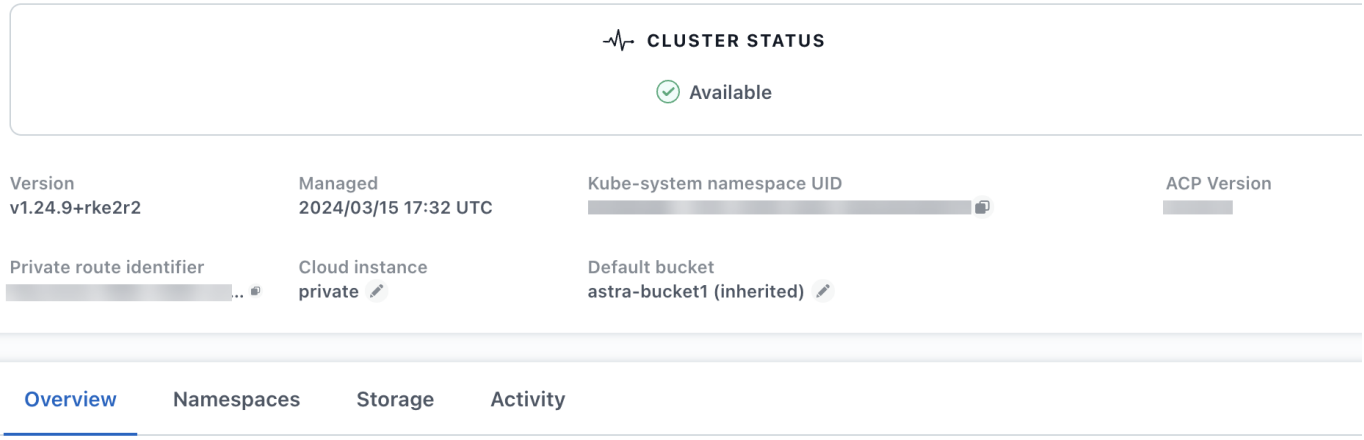
```
./tridentctl uninstall -n trident
```

- 在再次嘗試啟用 Astra Control Provisioner 之前、請勿 * "完全移除 Astra Trident 客戶需求日" 作為解除安裝的一部分。

結果

Astra Control Provisioner 功能已啟用、您可以使用任何適用於所執行版本的功能。

安裝 Astra Control Provisioner 之後、在 Astra Control UI 中裝載置備程式的叢集將會顯示 `ACP version` 而非 `Trident version` 欄位和目前安裝的版本號碼。



以取得更多資訊

- "Astra Trident 升級文件"

版權資訊

商標資訊