升級Astra Trident

Astra Trident

NetApp
April 04, 2024

目錄

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、提供新功能與效能增強功能、以及錯誤修正與改善功能。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。
- 從Trident 20.01開始、只有的試用版 "Volume快照" 支援。Kubernetes系統管理員應謹慎地將Alpha快照物件安全備份或轉換成試用版、以保留舊版Alpha快照。
  - 現在、自Kubernetes 1.20開始、「csi Volume Snapshot」就是GA的一項功能。在升級之前、您應該使用移除 Alpha Snapshot CRD `tridentctl obliviate alpha-snapshot-crd` 刪除 Alpha 快照規格的 CRD 。
  - Volume 快照的 Beta 版本引進一組修改過的 CRD 和快照控制器、這兩個控制器都應該在升級 Astra Trident 之前設定。
  - 如需詳細資訊、請參閱 "升級 Kubernetes 叢集之前必須瞭解的事項"。
- 所有版本 19.04 及更早版本的升級都需要從其本身移轉 Astra Trident 中繼資料 `etcd` 至CRD物件。請務必檢查 "Astra Trident 版本專屬的文件" 瞭解升級的運作方式。
- 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.templ 等範例[`storage-class-basic.yaml.templ`^] 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A `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 22.04 升級至 23.04 （包括任何點版本、例如 22.04.1 ）。
- 如果您有較早的版本、則應使用個別版本的文件來執行多步驟升級、以取得特定指示。這需要您先升級至最新版本、以符合您的四個版本。例如、如果您執行的是18.07、想要升級至20.07版本、請依照下列多步驟升級程序進行：
  a. 第一次從18.07升級至19.07。
  b. 然後從19.07升級至20.07。

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

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"。

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 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 。

步驟 3 ：選擇升級方法

升級 Astra Trident 有兩種方法。

何時使用營運商進行升級

您可以 "使用 Trident 運算子進行升級" 如果：

- 您最初是使用運算子或使用來安裝 Astra Trident `tridentctl`。
- 您已解除安裝 CSI Trident 、但安裝中的中繼資料仍會持續存在。
- 您有 CSI 型 Astra Trident 安裝。從19.07版開始的所有版本均採用基於SCSI的。您可以檢查 Trident 命名空間中的 Pod 、以驗證您的版本。
  - 23.01 之前版本的 Pod 命名使用： `trident-csi-*`
  - 23.01及更新版本中的Pod命名使用：
    - `trident-controller-<generated id>` 適用於控制器 Pod
    - `trident-node-<operating system>-<generated id>` 適用於節點 Pod
    - `trident-operator-<generated id>` 適用於操作人員 Pod

如果您使用的是、請勿使用運算子來升級Trident `etcd`Trident版本（19.04或更早版本）。

何時使用升級 `tridentctl`

您可以 如果您最初使用「 tridenctl 」安裝 Astra Trident 。

`tridentctl` 是安裝 Astra Trident 的傳統方法、為需要複雜自訂的客戶提供最多選項。如需詳細資料、請參閱 "選擇您的安裝方法"。

操作員變更

21.01 版的 Astra Trident 為駕駛員帶來架構變更：

- 運算子現在*叢集範圍*。Trident運算子先前的執行個體（20.04到20.10版）為*命名空間範圍*。叢集範圍內的運算子具有下列優點：
  - 資源責任：營運者現在可在叢集層級管理與Astra Trident安裝相關的資源。在安裝Astra Trident的過程中、營運者會使用來建立及維護多項資源 `ownerReferences`。維護 `ownerReferences` 在叢集範圍內的資源上、某些Kubernetes經銷商（例如OpenShift）可能會發生錯誤。叢集範圍的運算子可減輕此問題。對於自動修復和修補Trident資源、這是必要的需求。
  - 卸載期間清理：完整移除Astra Trident將需要刪除所有相關資源。命名空間範圍的運算子可能會在移除叢集範圍的資源（例如叢集角色、叢集角色繫結和Podcast安全性原則）時遇到問題、並導致不完整的清理。叢集範圍的運算子可消除此問題。使用者可以完全解除安裝Astra Trident、並視需要重新安裝。
- `TridentProvisioner` 現已取代為 `TridentOrchestrator` 作為用於安裝及管理Astra Trident的自訂資源。此外、也會在中引進新的欄位 `TridentOrchestrator` 規格使用者可以指定命名空間Trident必須使用安裝/升級 `spec.namespace` 欄位。您可以參考範例 "請按這裡"。

與營運者一起升級

您可以使用操作員手動或使用 Helm 輕鬆升級現有的 Astra Trident 安裝。

使用**Trident**營運者進行升級

一般而言、您應該使用原本用於安裝 Astra Trident 的相同方法來升級 Astra Trident 。檢閱 "選擇升級方法" 嘗試與 Trident 運算子一起升級之前。

> 從使用命名空間範圍運算子（ 20.07 至 20.10 版）安裝的 Astra Trident 執行個體升級時、Trident 運算子會自動：
>
> - 移轉 `tridentProvisioner` 至 `tridentOrchestrator` 具有相同名稱的物件、
> - 刪除 `TridentProvisioner` 物件和 `tridentprovisioner` 客戶需求日
> - 將 Astra Trident 升級至使用的叢集範圍運算子版本
> - 在最初安裝 Astra Trident 的相同命名空間中安裝 Astra Trident

升級叢集範圍的**Trident**操作員安裝

您可以升級叢集範圍的 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。

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、如果您要從 22.01 升級、請執行下列命令：

```
kubectl delete -f 22.01/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裝組合Yaml檔案和Astra Trident版本來安裝Astra Trident。例如、如果您要為 Kubernetes 1.27 安裝 Astra Trident 23.04 、請執行下列命令：

```
kubectl create -f 23.04.0/trident-installer/deploy/bundle_post_1_25.yaml
-n trident
```

Tridentt提供一個套裝組合檔案、可用來安裝運算子、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"。

結果

Trident營運者將識別現有的Astra Trident安裝、並將其升級至與營運者相同的版本。

升級命名空間範圍內的操作員安裝

您可以使用命名空間範圍運算子（ 20.07 至 20.10 版）、從安裝的 Astra Trident 執行個體升級到叢集範圍的運算子安裝。

開始之前

您需要用來部署命名空間範圍運算子的套件 YAML 檔案 `https://github.com/NetApp/trident/tree/stable/vXX.XX/deploy/BUNDLE.YAML` 其中 *`vXX.XX`* 為版本編號和 *`BUNDLE.YAML`* 為套裝組合Yaml檔案名稱。

步驟

1. 驗證 `TridentProvisioner` 現有 Trident 安裝的狀態為 `Installed`。

```
kubectl describe tprov trident -n trident | grep Message: -A 3

Message:  Trident installed
Status:   Installed
Version:  v20.10.1
```

如果狀態顯示 `Updating`、請務必先解決此問題、再繼續進行。如需可能狀態值的清單、請參閱 "請按這裡"。

2. 建立 `TridentOrchestrator` 請使用Trident安裝程式隨附的資訊清單來進行CRD。

```
# Download the release required [23.04.0]
mkdir 23.04.0
cd 23.04.0
wget
https://github.com/NetApp/trident/releases/download/v23.04.0/trident-
installer-23.04.0.tar.gz
tar -xf trident-installer-23.04.0.tar.gz
cd trident-installer
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
```

3. 使用命名空間範圍的運算子資訊清單來刪除。
   a. 確保您位於正確的目錄中。

```
pwd
/root/20.10.1/trident-installer
```

   b. 刪除命名空間範圍運算子。

```
kubectl delete -f deploy/<BUNDLE.YAML> -n trident

serviceaccount "trident-operator" deleted
clusterrole.rbac.authorization.k8s.io "trident-operator" deleted
clusterrolebinding.rbac.authorization.k8s.io "trident-operator"
deleted
deployment.apps "trident-operator" deleted
podsecuritypolicy.policy "tridentoperatorpods" deleted
```

   c. 確認 Trident 運算子已移除。

```
kubectl get all -n trident

NAME                                  READY   STATUS    RESTARTS   AGE
pod/trident-csi-68d979fb85-dsrmn   6/6     Running   12         99d
pod/trident-csi-8jfhf              2/2     Running   6          105d
pod/trident-csi-jtnjz              2/2     Running   6          105d
pod/trident-csi-lcxvh              2/2     Running   8          105d

NAME                   TYPE        CLUSTER-IP       EXTERNAL-IP
PORT(S)              AGE
service/trident-csi    ClusterIP   10.108.174.125   <none>
34571/TCP,9220/TCP   105d

NAME                           DESIRED   CURRENT   READY   UP-TO-DATE
AVAILABLE   NODE SELECTOR                                        AGE
daemonset.apps/trident-csi   3         3         3       3
3           kubernetes.io/arch=amd64,kubernetes.io/os=linux   105d

NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/trident-csi    1/1     1            1           105d

NAME                                      DESIRED   CURRENT   READY
AGE
replicaset.apps/trident-csi-68d979fb85    1         1         1
105d
```

4. （選用）如果需要修改安裝參數、請更新 `TridentProvisioner` 規格這可能包括變更：的值 `tridentImage`、 `autosupportImage`、私有映像儲存庫、以及提供 `imagePullSecrets`）刪除命名空間範圍的運算子之後、安裝叢集範圍的運算子之前。如需可更新的完整參數清單、請參閱 "組態選項"。

```
kubectl patch tprov <trident-provisioner-name> -n <trident-namespace>
--type=merge -p '{"spec":{"debug":true}}'
```

5. 安裝 Trident 叢集範圍運算子。
   a. 確保您位於正確的目錄中。

```
pwd
/root/23.04.0/trident-installer
```

   b. 在同一個命名空間中安裝叢集範圍的運算子。

Trident提供一個套裝組合檔案、可用來安裝運算子、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"。

```
kubectl create -f deploy/<BUNDLE.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

#All tridentProvisioners will be removed, including the CRD itself
kubectl get tprov -n trident
Error from server (NotFound): Unable to list "trident.netapp.io/v1,
Resource=tridentprovisioners": the server could not find the
requested resource (get tridentprovisioners.trident.netapp.io)

#tridentProvisioners are replaced by tridentOrchestrator
kubectl get torc
NAME      AGE
trident   13s
```

c. 檢查命名空間中的 Trident Pod 。。 `trident-controller` 和Pod名稱反映了23.01中引入的命名慣例。

```
kubectl get pods -n trident

NAME                                  READY   STATUS    RESTARTS
AGE
trident-controller-79df798bdc-m79dc      6/6     Running   0
1m41s
trident-node-linux-xrst8                 2/2     Running   0
1m41s
trident-operator-5574dbbc68-nthjv        1/1     Running   0
1m52s
```

d. 確認 Trident 已更新至所需版本。

```
kubectl describe torc trident | grep Message -A 3
Message:                Trident installed
Namespace:              trident
Status:                 Installed
Version:                v23.04.0
```

升級Helm型的營運者安裝

請執行下列步驟、升級Helm型的操作員安裝。

> 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. 下載最新的Astra Trident版本。
2. 使用 `helm upgrade` 命令位置 `trident-operator-23.04.0.tgz` 反映您要升級的版本。

```
helm upgrade <name> trident-operator-23.04.0.tgz
```

> 如果您在初始安裝期間設定任何非預設選項（例如指定Trident和csi映像的私有、鏡射登錄）、請使用 `--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。
>
> 例如、變更的預設值 `tridentDebug`，執行下列命令：
>
> ```
> helm upgrade <name> trident-operator-23.04.0-custom.tgz --set tridentDebug=true
> ```

3. 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。

結果

Trident營運者將識別現有的Astra Trident安裝、並將其升級至與營運者相同的版本。

從非營運者安裝升級

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

步驟

1. 下載最新的Astra Trident版本。

```
# Download the release required [23.04.0]
mkdir 23.04.0
cd 23.04.0
wget
https://github.com/NetApp/trident/releases/download/v22.01.1/trident-
installer-23.04.0.tar.gz
tar -xf trident-installer-23.04.0.tar.gz
cd trident-installer
```

2. 建立 `tridentorchestrator` 資訊清單中的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.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` 用於安裝Astra Trident的CR。

```
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:                v23.04.0
```

結果

現有的後端和PVCS會自動提供使用。

使用tridentctl進行升級

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

使用升級 **Astra Trident** `tridentctl`

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。

使用升級磁碟區 `tridentctl`

升級後、您可以使用較新的 Trident 版本（例如隨選 Volume Snapshot ）所提供的豐富功能集、使用來升級磁碟區 `tridentctl upgrade` 命令。

如果有舊版磁碟區、您應該將它們從 NFS 或 iSCSI 類型升級至 CSI 類型、以使用 Astra Trident 中的完整新功能集。Trident提供的舊PV支援傳統功能集。

開始之前

在決定將磁碟區升級為 CSI 類型之前、請考量下列事項：

- 您可能不需要升級所有磁碟區。先前建立的磁碟區將繼續可供存取、並正常運作。
- 在升級時、PV可作為部署/狀態集的一部分掛載。不需要關閉部署/狀態集。
- 您*無法*在升級時將PV附加至獨立式Pod。在升級磁碟區之前、您應該先關閉Pod。
- 您只能升級綁定到PVc的磁碟區。在升級之前、應先移除和匯入未繫結至PVCS的磁碟區。

步驟

1. 執行 `kubectl get pv` 以列出PV。

```
kubectl get pv
NAME                        CAPACITY     ACCESS MODES   RECLAIM POLICY
STATUS   CLAIM                  STORAGECLASS    REASON   AGE
default-pvc-1-a8475         1073741824   RWO            Delete
Bound    default/pvc-1          standard                19h
default-pvc-2-a8486         1073741824   RWO            Delete
Bound    default/pvc-2          standard                19h
default-pvc-3-a849e         1073741824   RWO            Delete
Bound    default/pvc-3          standard                19h
default-pvc-4-a84de         1073741824   RWO            Delete
Bound    default/pvc-4          standard                19h
trident                     2Gi          RWO            Retain
Bound    trident/trident                                19h
```

目前有四個PV是由Trident 20.07使用所建立 `netapp.io/trident` 置備程式：

2. 執行 `kubectl describe pv` 以取得PV的詳細資料。

```
kubectl describe pv default-pvc-2-a8486

Name:            default-pvc-2-a8486
Labels:          <none>
Annotations:     pv.kubernetes.io/provisioned-by: netapp.io/trident
                 volume.beta.kubernetes.io/storage-class: standard
Finalizers:      [kubernetes.io/pv-protection]
StorageClass:    standard
Status:          Bound
Claim:           default/pvc-2
Reclaim Policy:  Delete
Access Modes:    RWO
VolumeMode:      Filesystem
Capacity:        1073741824
Node Affinity:   <none>
Message:
Source:
    Type:      NFS (an NFS mount that lasts the lifetime of a pod)
    Server:    10.xx.xx.xx
    Path:      /trid_1907_alpha_default_pvc_2_a8486
    ReadOnly:  false
```

PV是使用建立的 `netapp.io/trident` 資源配置程式、類型為NFS。為了支援Astra Trident提供的所有新功能、此PV應升級為「csi」類型。

3. 執行 `tridentctl upgrade volume <name-of-trident-volume>` 將舊Astra Trident Volume升級至csi規格的命令。

```
./tridentctl get volumes -n trident
+---------------------+---------+--------------+----------
+--------------------------------------+--------+---------+
|            NAME     |  SIZE   | STORAGE CLASS | PROTOCOL |
BACKEND UUID             | STATE  | MANAGED |
+---------------------+---------+--------------+----------
+--------------------------------------+--------+---------+
| default-pvc-2-a8486 | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true    |
| default-pvc-3-a849e | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true    |
| default-pvc-1-a8475 | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true    |
| default-pvc-4-a84de | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true    |
+---------------------+---------+--------------+----------
+--------------------------------------+--------+---------+

./tridentctl upgrade volume default-pvc-2-a8486 -n trident
+---------------------+---------+--------------+----------
+--------------------------------------+--------+---------+
|            NAME     |  SIZE   | STORAGE CLASS | PROTOCOL |
BACKEND UUID             | STATE  | MANAGED |
+---------------------+---------+--------------+----------
+--------------------------------------+--------+---------+
| default-pvc-2-a8486 | 1.0 GiB | standard      | file     | c5a6f6a4-
b052-423b-80d4-8fb491a14a22 | online | true    |
+---------------------+---------+--------------+----------
+--------------------------------------+--------+---------+
```

4. 執行 `kubectl describe pv` 以驗證Volume是否為「csi Volume」（SCSI Volume）。

```
kubectl describe pv default-pvc-2-a8486
Name:            default-pvc-2-a8486
Labels:          <none>
Annotations:     pv.kubernetes.io/provisioned-by: csi.trident.netapp.io
                 volume.beta.kubernetes.io/storage-class: standard
Finalizers:      [kubernetes.io/pv-protection]
StorageClass:    standard
Status:          Bound
Claim:           default/pvc-2
Reclaim Policy:  Delete
Access Modes:    RWO
VolumeMode:      Filesystem
Capacity:        1073741824
Node Affinity:   <none>
Message:
Source:
    Type:              CSI (a Container Storage Interface (CSI) volume
source)
    Driver:            csi.trident.netapp.io
    VolumeHandle:      default-pvc-2-a8486
    ReadOnly:          false
    VolumeAttributes:      backendUUID=c5a6f6a4-b052-423b-80d4-
8fb491a14a22

internalName=trid_1907_alpha_default_pvc_2_a8486
                           name=default-pvc-2-a8486
                           protocol=file
Events:                <none>
```

版權資訊

商標資訊