# 與Trident營運者一起部署

Astra Trident

NetApp
April 16, 2024

This PDF was generated from https://docs.netapp.com/zh-tw/trident-2207/trident-get-started/kubernetes-customize-deploy.html on April 16, 2024. Always check docs.netapp.com for the latest.

# 目錄

與Trident營運者一起部署 . . . . . 1
部署Trident操作員、並使用Helm安裝Astra Trident . . . . . 1
手動部署Trident運算子 . . . . . 2
自訂Trident營運者部署 . . . . . 7

# 與Trident營運者一起部署

您可以使用Trident運算子來部署Astra Trident。您可以使用下列兩種方式之一來部署Trident運算子：

- 使用Trident "掌舵表"：Helm圖表可部署Trident運算子、並在單一步驟中安裝Trident。
- 手動：Trident提供 "bunder.yaml" 可用於安裝運算子及建立關聯物件的檔案。

> 如果您尚未熟悉 "基本概念"現在正是這麼做的好時機。

您需要的產品

若要部署Astra Trident、必須符合下列先決條件：

- 您擁有執行Kubernetes 1.19 - 1.24之受支援Kubernetes叢集的完整權限。
- 您可以存取支援的NetApp儲存系統。
- 您可以從所有Kubernetes工作節點掛載磁碟區。
- 您有一部Linux主機 `kubectl` （或 `oc`（如果您使用OpenShift）已安裝並設定為管理您要使用的Kubernetes叢集。
- 您已設定 `KUBECONFIG` 指向Kubernetes叢集組態的環境變數。
- 您已啟用 "具備Astra Trident所需的閘道"。
- 如果您使用Kubernetes搭配Docker Enterprise、 "請依照他們的步驟啟用CLI存取"。

您知道嗎？太棒了！讓我們開始吧。

## 部署Trident操作員、並使用Helm安裝Astra Trident

使用Helm執行列出的步驟來部署Trident運算子。

您需要的產品

除了上述先決條件之外、若要使用Helm部署Trident運算子、您還需要下列項目：

- Kubernetes 1.19 - 1.24
- Helm版本3

步驟

1. 新增Trident的Helm儲存庫：

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

2. 使用 `helm install` 命令並指定部署名稱。請參閱下列範例：

```
helm install <release-name> netapp-trident/trident-operator --version
22.4.0 --create-namespace <trident-namespace>
```

> 如果您已經為Trident建立命名空間 `--create-namespace` 參數不會建立額外的命名空間。

安裝期間有兩種傳遞組態資料的方法：

- `--values` （或 `-f`）：指定具有覆寫的Yaml檔案。這可以多次指定、最右邊的檔案會優先。
- `--set`：在命令行中指定覆蓋。

例如、變更的預設值 `debug`請執行下列步驟 `--set` 命令：

```
helm install <name> netapp-trident/trident-operator --version 22.7.0 --set
tridentDebug=true
```

。 `values.yaml` 檔案是Helm圖表的一部分、提供金鑰清單及其預設值。

`helm list` 顯示安裝的詳細資料、例如名稱、命名空間、圖表、狀態、 應用程式版本、修訂編號等。

# 手動部署Trident運算子

執行所列步驟、手動部署Trident運算子。

## 步驟1：判斷Kubernetes叢集的資格

首先您需要登入Linux主機、然後確認它正在管理_運作_、 "支援的Kubernetes叢集" 您擁有必要的權限。

> 使用OpenShift、使用 `oc` 而非 `kubectl` 在以下所有範例中、請先執行*系統：admin*登入 `oc login -u system:admin` 或 `oc login -u kube-admin`。

若要驗證Kubernetes版本、請執行下列命令：

```
kubectl version
```

若要查看您是否具有Kubernetes叢集管理員權限、請執行下列命令：

```
kubectl auth can-i '*' '*' --all-namespaces
```

若要驗證是否可以從Docker Hub啟動使用映像的Pod、並透過Pod網路連線至儲存系統、請執行下列命令：

```
kubectl run -i --tty ping --image=busybox --restart=Never --rm -- \
  ping <management IP>
```

## 步驟2：下載並設定營運者

> 從21.01開始、Trident運算子就是叢集範圍。若要使用Trident運算子來安裝Trident、必須建立 `TridentOrchestrator` 自訂資源定義（CRD）和定義其他資源。在安裝Astra Trident之前、您應該先執行這些步驟來設定操作員。

1. 請從下載並擷取最新版本的Trident安裝程式套件 "GitHub的_Assets區段"。

   ```
   wget
   https://github.com/NetApp/trident/releases/download/v22.04.0/trident-
   installer-22.04.0.tar.gz
   tar -xf trident-installer-22.04.0.tar.gz
   cd trident-installer
   ```

2. 使用適當的CRD資訊清單來建立 `TridentOrchestrator` 客戶需求日然後建立 `TridentOrchestrator` 稍後再自訂資源以產生操作者的安裝。

   執行下列命令：

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

3. 之後 `TridentOrchestrator` 建立客戶需求日之後、請建立下列操作員部署所需的資源：

   - 營運者的服務帳戶
   - 叢集角色和叢集角色繫結至服務帳戶
   - 專屬的PodSecurity原則
   - 營運者本身

     Trident安裝程式包含定義這些資源的資訊清單。根據預設、操作員會部署在中 `trident` 命名空間。如果是 `trident` 命名空間不存在、請使用下列資訊清單來建立命名空間。

   ```
   kubectl apply -f deploy/namespace.yaml
   ```

4. 可在非預設名稱空間中部署運算子 `trident` 命名空間、您應該更新 `serviceaccount.yaml`、`clusterrolebinding.yaml` 和 `operator.yaml` 提供清單並產生您的 `bundle.yaml`。

   執行下列命令以更新Yaml清單並產生您的 `bundle.yaml` 使用 `kustomization.yaml`：

```
kubectl kustomize deploy/ > deploy/bundle.yaml
```

執行下列命令以建立資源並部署營運者：

```
kubectl create -f deploy/bundle.yaml
```

5. 若要在部署後驗證操作員的狀態、請執行下列步驟：

```
kubectl get deployment -n <operator-namespace>

NAME                READY   UP-TO-DATE   AVAILABLE   AGE
trident-operator    1/1     1            1           3m
```

```
kubectl get pods -n <operator-namespace>

NAME                                READY   STATUS             RESTARTS
AGE
trident-operator-54cb664d-lnjxh     1/1     Running            0
3m
```

營運者部署成功建立一個在叢集中其中一個工作節點上執行的Pod。

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

## 步驟3：建立 `TridentOrchestrator` 並安裝Trident

您現在可以使用運算子來安裝Astra Trident！這需要建立 `TridentOrchestrator`。Trident安裝程式隨附建立的範例定義 `TridentOrchestrator`。這會啟動中的安裝 `trident` 命名空間。

```
kubectl create -f deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created

kubectl describe torc trident
Name:         trident
Namespace:
Labels:       <none>
Annotations: <none>
API Version: trident.netapp.io/v1
Kind:         TridentOrchestrator
...
Spec:
  Debug:     true
  Namespace: trident
Status:
  Current Installation Params:
    IPv6:                      false
    Autosupport Hostname:
    Autosupport Image:         netapp/trident-autosupport:21.04
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                     true
    Image Pull Secrets:
    Image Registry:
    k8sTimeout:           30
    Kubelet Dir:          /var/lib/kubelet
    Log Format:           text
    Silence Autosupport:  false
    Trident Image:        netapp/trident:21.04.0
  Message:                  Trident installed  Namespace:
trident
  Status:                   Installed
  Version:                  v21.04.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

Trident運算子可讓您使用中的屬性、自訂Astra Trident的安裝方式 `TridentOrchestrator` 規格請參閱 "自訂您的Trident部署"。

的狀態 `TridentOrchestrator` 指出安裝是否成功、並顯示安裝的Trident版本。

| 狀態 | 說明 |
|---|---|
| 安裝 | 操作員正在使用此工具安裝Astra Trident `TridentOrchestrator` CR. |
| 已安裝 | Astra Trident已成功安裝。 |
| 正在解除安裝 | 因為、操作者正在解除安裝Astra Trident `spec.uninstall=true`。 |
| 已解除安裝 | Astra Trident已解除安裝。 |
| 失敗 | 營運者無法安裝、修補、更新或解除安裝Astra Trident；營運者將自動嘗試從此狀態恢復。如果此狀態持續存在、您將需要疑難排解。 |
| 正在更新 | 營運者正在更新現有的安裝。 |
| 錯誤 | 。 `TridentOrchestrator` 未使用。另一個已經存在。 |

安裝期間的狀態 `TridentOrchestrator` 變更來源 `Installing` 至 `Installed`。如果您觀察到 `Failed` 狀態且操作員無法自行恢復、您應該檢查操作員的記錄。請參閱 "疑難排解" 區段。

您可以查看已建立的Pod、確認Astra Trident安裝是否已完成：

```
kubectl get pod -n trident

NAME                                  READY   STATUS    RESTARTS   AGE
trident-csi-7d466bf5c7-v4cpw          5/5     Running   0           1m
trident-csi-mr6zc                     2/2     Running   0           1m
trident-csi-xrp7w                     2/2     Running   0           1m
trident-csi-zh2jt                     2/2     Running   0           1m
trident-operator-766f7b8658-ldzsv     1/1     Running   0           3m
```

您也可以使用 `tridentctl` 檢查安裝的Astra Trident版本。

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 21.04.0        | 21.04.0        |
+----------------+----------------+
```

現在您可以繼續建立後端。請參閱 "部署後工作"。

如需部署期間的疑難排解問題、請參閱 "疑難排解" 區段。

# 自訂Trident營運者部署

Trident運算子可讓您使用中的屬性、自訂Astra Trident的安裝方式 `TridentOrchestrator` 規格

請參閱下表以取得屬性清單：

| 參數 | 說明 | 預設 |
|---|---|---|
| `namespace` | 用於安裝Astra Trident的命名空間 | "預設" |
| `debug` | 啟用Astra Trident的偵錯功能 | 錯 |
| `IPv6` | 透過IPv6安裝Astra Trident | 錯 |
| `k8sTimeout` | Kubernetes作業逾時 | 30秒 |
| `silenceAutosupport` | 請勿AutoSupport 自動將此套裝組合傳送至NetApp | 錯 |
| `enableNodePrep` | 自動管理工作節點相依性（* BETA *） | 錯 |
| `autosupportImage` | 遙測的容器影像AutoSupport | 「NetApp/Trident自動支援：21.04.0」 |
| `autosupportProxy` | 代理伺服器的位址/連接埠、用於傳送AutoSupport 「遙測」功能 | "<a href="http://proxy.example.com:8888"" class="bare">http://proxy.example.com:8888"</a> |
| `uninstall` | 用來解除安裝Astra Trident的旗標 | 錯 |
| `logFormat` | 要使用的Astra Trident記錄格式[text、json] | "文字" |
| `tridentImage` | 要安裝的Astra Trident映像 | 「NetApp/Trident：21.04」 |
| `imageRegistry` | 內部登錄的路徑、格式 `<registry FQDN>[:port][/subpath]` | "k8s.gcr.io/sig-storage（k8s 1.19+）或kay.io/k8scsi" |
| `kubeletDir` | 主機上的kubelet目錄路徑 | "/var/lib/kubelet" |
| `wipeout` | 要刪除以執行完整移除Astra Trident的資源清單 | |
| `imagePullSecrets` | 從內部登錄擷取映像的機密 | |
| `controllerPluginNodeSelector` | 執行Trident控制器csi外掛程式之Pod的其他節點選取器。格式與pod.spec.nodeSelector相同。 | 無預設值；選用 |
| `controllerPluginTolerations` | 覆寫執行Trident控制器csi外掛程式之Pod的容錯能力。格式與po.spec.TBolerations相同。 | 無預設值；選用 |
| `nodePluginNodeSelector` | 執行Trident Node SCSI外掛程式之Pod的其他節點選取器。格式與pod.spec.nodeSelector相同。 | 無預設值；選用 |

| 參數 | 說明 | 預設 |
| --- | --- | --- |
| `nodePluginTolerations` | 覆寫執行Trident Node SCSI外掛程式之Pod的容錯能力。格式與po.spec.TBolerations相同。 | 無預設值；選用 |

`spec.namespace` 在中指定 `TridentOrchestrator` 以表示安裝在哪個命名空間Astra Trident。此參數*無法在安裝Astra Trident之後更新*。嘗試這麼做會導致的狀態 `TridentOrchestrator` 變更為 `Failed`。Astra Trident不打算跨命名空間移轉。

如需格式化Pod參數的詳細資訊、請參閱 "將Pod指派給節點"。

您可以在定義時使用上述屬性 `TridentOrchestrator` 以自訂安裝。範例如下：

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

以下是另一個範例、說明如何使用節點選取器來部署Trident：

```
apiVersion: trident.netapp.io/v1
kind: TridentOrchestrator
metadata:
  name: trident
spec:
  debug: true
  namespace: trident
  controllerPluginNodeSelector:
    nodetype: master
  nodePluginNodeSelector:
    storage: netapp
```

如果您想要自訂安裝內容以外的內容 `TridentOrchestrator` 參數允許、您應該考慮使用 `tridentctl` 產生可視需要修改的自訂Y反 洗錢清單。

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 保留隨時變更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。