Tridentオペレータを使用してインストール

Astra Trident

NetApp
March 05, 2026

This PDF was generated from https://docs.netapp.com/ja-jp/trident-2304/trident-get-started/kubernetes-deploy-operator.html on March 05, 2026. Always check docs.netapp.com for the latest.

目次

Tridentオペレータを使用してインストール . . . 1
- Tridentオペレータを手動で導入（標準モード） . . . 1
  - Astra Tridentに関する重要な情報23.04 . . . 1
  - Tridentオペレータを手動で導入し、Tridentをインストール . . . 1
  - インストールを確認します。 . . . 4
  - 次のステップ . . . 6
- Tridentオペレータを手動で導入（オフラインモード） . . . 6
  - Astra Tridentに関する重要な情報23.04 . . . 6
  - Tridentオペレータを手動で導入し、Tridentをインストール . . . 6
  - インストールを確認します。 . . . 11
  - 次のステップ . . . 12
- Helm（標準モード）を使用してTridentを導入 . . . 12
  - Astra Tridentに関する重要な情報23.04 . . . 12
  - Tridentオペレータを導入し、Helmを使用してAstra Tridentをインストール . . . 13
  - インストール中に設定データを渡す . . . 13
  - 設定オプション . . . 14
  - 次のステップ . . . 16
- Helm（オフラインモード）を使用したTridentのオペレータの導入 . . . 17
  - Astra Tridentに関する重要な情報23.04 . . . 17
  - Tridentオペレータを導入し、Helmを使用してAstra Tridentをインストール . . . 17
  - インストール中に設定データを渡す . . . 18
  - 設定オプション . . . 19
  - 次のステップ . . . 21
- Tridentオペレータのインストールをカスタマイズ . . . 21
  - コントローラポッドとノードポッドについて . . . 21
  - 設定オプション . . . 22
  - 構成例 . . . 24

Tridentオペレータを使用してインストール

Tridentオペレータを手動で導入（標準モード）

Tridentオペレータが手動で導入してAstra Tridentをインストールできます。このプロセスでは、環境 をインストールする際に、Astra Tridentで必要なコンテナイメージがプライベートレジストリに格納されません。プライベートイメージレジストリがある場合は、を使用します "オフライン導入のプロセス"。

Astra Tridentに関する重要な情報23.04

- Astra Tridentに関する次の重要な情報をお読みください。*

** ：Trident ** に関する重要な情報

- TridentでKubernetes 1.27がサポートされるようになりました。Kubernetesをアップグレードする前にTridentをアップグレード
- Astra Tridentは、SAN環境でマルチパス構成を厳密に使用し、推奨される値をに設定します `find_multipaths: no` multipath.confファイル内。

  非マルチパス構成またはを使用 `find_multipaths: yes` または `find_multipaths: smart` multipath.confファイルの値が原因でマウントが失敗します。Tridentはの使用を推奨しています `find_multipaths: no` 21.07リリース以降

Tridentオペレータを手動で導入し、Tridentをインストール

レビュー "インストールの概要" インストールの前提条件を満たし、環境に適したインストールオプションを選択していることを確認します。

作業を開始する前に

インストールを開始する前に、Linuxホストにログインして、管理が機能していることを確認します。 "サポートされる Kubernetes クラスタ" 必要な権限があることを確認します。

> OpenShiftでは、を使用します `oc` ではなく `kubectl` 以降のすべての例では、を実行して、最初に* system:admin *としてログインします `oc login -u system:admin` または `oc login -u kube-admin`。

1. Kubernetesのバージョンを確認します。

```
kubectl version
```

2. クラスタ管理者の権限を確認します。

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

3. Docker Hubのイメージを使用してポッドを起動し、ポッドネットワーク経由でストレージシステムにアクセスできることを確認します。

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

手順**1**：**Trident**のインストーラパッケージをダウンロード

Astra Tridentインストーラパッケージには、Tridentオペレータの導入とAstra Tridentのインストールに必要なものがすべて含まれています。から最新バージョンのTridentインストーラをダウンロードして展開します "GitHubの_Assets_sectionを参照してください"。

```
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
```

手順**2**：を作成します `TridentOrchestrator` **CRD**

を作成します `TridentOrchestrator` カスタムリソース定義（CRD）。を作成します `TridentOrchestrator` カスタムリソース。で適切なCRD YAMLバージョンを使用します `deploy/crds` を作成します `TridentOrchestrator` CRD。

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

手順**3**：**Trident**のオペレータを導入する

Astra Tridentインストーラには、オペレータのインストールや関連オブジェクトの作成に使用できるバンドルファイルが用意されています。このバンドルファイルを使用すると、オペレータを簡単に導入し、デフォルトの設定でAstra Tridentをインストールできます。

- クラスタでKubernetes 1.24以前を実行している場合は、を使用します `bundle_pre_1_25.yaml`。

- クラスタでKubernetes 1.25以降を実行している場合は、を使用します `bundle_post_1_25.yaml`。

作業を開始する前に

- デフォルトでは、Tridentのインストーラによって `trident` ネームスペース：状況に応じて `trident` ネームスペースが存在しません。次を使用して作成してください：

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

- オペレータを以外のネームスペースに配置する場合 `trident` 名前空間、更新 `serviceaccount.yaml`、 `clusterrolebinding.yaml` および `operator.yaml` を使用してバンドルファイルを生成します `kustomization.yaml`。

  a. を作成します `kustomization.yaml` 次のコマンドを使用して、*<bundle>* is `bundle_pre_1_25` または `bundle_post_1_25` 使用しているKubernetesのバージョンに基づきます。

  ```
  cp kustomization_<bundle>.yaml kustomization.yaml
  ```

  b. 次のコマンドを使用してバンドルをコンパイルします。WHERE_STORE_IS <bundle> `bundle_pre_1_25` または `bundle_post_1_25` 使用しているKubernetesのバージョンに基づきます。

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

手順

1. リソースを作成し、オペレータを配置します。

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

2. operator、deployment、およびReplicaSetsが作成されたことを確認します。

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

> Kubernetes クラスタには、オペレータのインスタンスが * 1 つしか存在しないようにしてください。Trident のオペレータが複数の環境を構築することは避けてください。

手順**4**： `TridentOrchestrator` **Trident**をインストール

これで、を作成できます `TridentOrchestrator` Astra Tridentを導入必要に応じて、を実行できます "Tridentのインストールをカスタマイズ" で属性を使用する `TridentOrchestrator` 仕様

```
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:23.04
    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:23.04.0
  Message:                  Trident installed  Namespace:
trident
  Status:                   Installed
  Version:                  v23.04.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

インストールを確認します。

インストールを確認するには、いくつかの方法があります。

を使用します TridentOrchestrator ステータス

のステータス TridentOrchestrator インストールが正常に完了したかどうかを示し、インストールされているTridentのバージョンが表示されます。インストール中、のステータス TridentOrchestrator からの変更 Installing 終了： Installed。を確認した場合は Failed ステータスとオペレータは単独で回復できません。 "ログをチェックしてください"。

ステータス	説明
インストール中です	このツールを使用してAstra Tridentをインストールしている TridentOrchestrator CR。
インストール済み	Astra Trident のインストールが完了しました。
アンインストール中です	OperatorはAstra Tridentをアンインストールしています。理由はです spec.uninstall=true。
アンインストール済み	Astra Trident がアンインストールされました。
失敗しました	オペレータがインストール、パッチ適用、アップデート、またはアンインストールできませんでした Astra Trident。オペレータはこの状態からのリカバリを自動的に試行します。この状態が解消されない場合は、トラブルシューティングが必要です。
更新中です	オペレータが既存のインストールを更新しています。
エラー	。 TridentOrchestrator は使用されません。もう一つ 存在します。

ポッドの作成ステータスを使用する

作成したポッドのステータスを確認することで、Astra Tridentのインストールが完了したかどうかを確認できます。

```
kubectl get pods -n trident

NAME                                  READY   STATUS    RESTARTS
AGE
trident-controller-7d466bf5c7-v4cpw   6/6     Running   0
1m
trident-node-linux-mr6zc              2/2     Running   0
1m
trident-node-linux-xrp7w              2/2     Running   0
1m
trident-node-linux-zh2jt              2/2     Running   0
1m
trident-operator-766f7b8658-ldzsv     1/1     Running   0
3m
```

を使用します `tridentctl`

を使用できます `tridentctl` インストールされているAstra Tridentのバージョンを確認します。

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 23.04.0        | 23.04.0        |
+----------------+----------------+
```

次のステップ

できるようになりました。 "バックエンドとストレージクラスを作成し、ボリュームをプロビジョニングして、ポッドにボリュームをマウントします"。

Tridentオペレータを手動で導入（オフラインモード）

Tridentオペレータが手動で導入してAstra Tridentをインストールできます。このプロセスでは、環境 をインストールする際に、Astra Tridentで必要なコンテナイメージがプライベートレジストリに格納されます。プライベートイメージレジストリがない場合は、を使用します "標準的な導入のプロセス"。

Astra Tridentに関する重要な情報23.04

- Astra Tridentに関する次の重要な情報をお読みください。*

** ：Trident ** に関する重要な情報

- TridentでKubernetes 1.27がサポートされるようになりました。Kubernetesをアップグレードする前にTridentをアップグレード
- Astra Tridentは、SAN環境でマルチパス構成を厳密に使用し、推奨される値をに設定します `find_multipaths: no` multipath.confファイル内。

  非マルチパス構成またはを使用 `find_multipaths: yes` または `find_multipaths: smart` multipath.confファイルの値が原因でマウントが失敗します。Tridentはの使用を推奨しています `find_multipaths: no` 21.07リリース以降

Tridentオペレータを手動で導入し、Tridentをインストール

レビュー "インストールの概要" インストールの前提条件を満たし、環境に適したインストールオプションを選択していることを確認します。

作業を開始する前に

Linuxホストにログインして、管理が機能していることとを確認します "サポートされる Kubernetes クラスタ" 必要な権限があることを確認します。

> OpenShiftでは、を使用します `oc` ではなく `kubectl` 以降のすべての例では、を実行して、最初に* system:admin *としてログインします `oc login -u system:admin` または `oc login -u kube-admin`。

1. Kubernetesのバージョンを確認します。

```
kubectl version
```

2. クラスタ管理者の権限を確認します。

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

3. Docker Hubのイメージを使用してポッドを起動し、ポッドネットワーク経由でストレージシステムにアクセスできることを確認します。

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

手順**1**：**Trident**のインストーラパッケージをダウンロード

Astra Tridentインストーラパッケージには、Tridentオペレータの導入とAstra Tridentのインストールに必要なものがすべて含まれています。から最新バージョンのTridentインストーラをダウンロードして展開します "GitHubの_Assets_sectionを参照してください"。

```
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
```

手順**2**：を作成します `TridentOrchestrator` **CRD**

を作成します `TridentOrchestrator` カスタムリソース定義（CRD）。を作成します `TridentOrchestrator` カスタムリソース。で適切なCRD YAMLバージョンを使用します `deploy/crds` を作成します `TridentOrchestrator` CRD：

```
kubectl create -f deploy/crds/<VERSION>.yaml
```

手順**3**：オペレータのレジストリの場所を更新します

インチ `/deploy/operator.yaml`、を更新します `image: docker.io/netapp/trident-operator:23.04.0` イメージレジストリの場所を反映します。。 "TridentとCSIの画像" 1つのレジストリまたは別のレジストリに配置できますが、すべてのCSIイメージは同じレジストリに配置する必要があります。例：

- `image: <your-registry>/trident-operator:23.04.0` すべての画像が1つのレジストリにある場合。
- `image: <your-registry>/netapp/trident-operator:23.04.0` TridentイメージがCSIイメージとは別のレジストリにある場合。

ステップ**4**：**Trident**オペレータを導入

Astra Tridentインストーラには、オペレータのインストールや関連オブジェクトの作成に使用できるバンドルファイルが用意されています。このバンドルファイルを使用すると、オペレータを簡単に導入し、デフォルトの設定でAstra Tridentをインストールできます。

- クラスタでKubernetes 1.24以前を実行している場合は、を使用します `bundle_pre_1_25.yaml`。
- クラスタでKubernetes 1.25以降を実行している場合は、を使用します `bundle_post_1_25.yaml`。

作業を開始する前に

- デフォルトでは、Tridentのインストーラによって `trident` ネームスペース：状況に応じて `trident` ネームスペースが存在しません。次を使用して作成してください：

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

- オペレータを以外のネームスペースに配置する場合 `trident` 名前空間、更新 `serviceaccount.yaml`、 `clusterrolebinding.yaml` および `operator.yaml` を使用してバンドルファイルを生成します `kustomization.yaml`。

  a. を作成します `kustomization.yaml` 次のコマンドを使用して、*<bundle>* is `bundle_pre_1_25` または `bundle_post_1_25` 使用しているKubernetesのバージョンに基づきます。

  ```
  cp kustomization_<bundle>.yaml kustomization.yaml
  ```

  b. 次のコマンドを使用してバンドルをコンパイルします。WHERE_STORE_IS <bundle> `bundle_pre_1_25` または `bundle_post_1_25` 使用しているKubernetesのバージョンに基づきます。

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

手順

1. リソースを作成し、オペレータを配置します。

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

2. operator、deployment、およびReplicaSetsが作成されたことを確認します。

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

Kubernetes クラスタには、オペレータのインスタンスが * 1 つしか存在しないようにしてください。Trident のオペレータが複数の環境を構築することは避けてください。

手順**5:**でイメージレジストリの場所を更新します `TridentOrchestrator`

。 "TridentとCSIの画像" 1つのレジストリまたは別のレジストリに配置できますが、すべてのCSIイメージは同じレジストリに配置する必要があります。更新 `deploy/crds/tridentorchestrator_cr.yaml` レジストリ設定に基づいて追加の場所の仕様を追加します。

1つのレジストリ内のイメージ

```
imageRegistry: "<your-registry>"
autosupportImage: "<your-registry>/trident-autosupport:23.04"
tridentImage: "<your-registry>/trident:23.04.0"
```

異なるレジストリ内の画像

を追加する必要があります `sig-storage` に移動します `imageRegistry` 別のレジストリの場所を使用します。

```
imageRegistry: "<your-registry>/sig-storage"
autosupportImage: "<your-registry>/netapp/trident-autosupport:23.04"
tridentImage: "<your-registry>/netapp/trident:23.04.0"
```

手順**6**： `TridentOrchestrator` **Trident**をインストール

これで、を作成できます `TridentOrchestrator` Astra Tridentを導入必要に応じて、さらに行うことができます "Tridentのインストールをカスタマイズ" で属性を使用する `TridentOrchestrator` 仕様次の例は、TridentイメージとCSIイメージが異なるレジストリにあるインストールを示しています。

```
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:
  Autosupport Image:  <your-registry>/netapp/trident-autosupport:23.04
  Debug:              true
  Image Registry:     <your-registry>/sig-storage
  Namespace:          trident
  Trident Image:      <your-registry>/netapp/trident:23.04.0
Status:
  Current Installation Params:
    IPv6:                       false
    Autosupport Hostname:
    Autosupport Image:          <your-registry>/netapp/trident-
autosupport:23.04
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:                      true
    Http Request Timeout:       90s
    Image Pull Secrets:
    Image Registry:       <your-registry>/sig-storage
    k8sTimeout:           30
    Kubelet Dir:          /var/lib/kubelet
    Log Format:           text
    Probe Port:           17546
    Silence Autosupport:  false
    Trident Image:        <your-registry>/netapp/trident:23.04.0
  Message:                Trident installed
  Namespace:              trident
  Status:                 Installed
  Version:                v23.04.0
Events:
    Type Reason Age From Message ---- ------ ---- ---- -------Normal
    Installing 74s trident-operator.netapp.io Installing Trident Normal
    Installed 67s trident-operator.netapp.io Trident installed
```

インストールを確認します。

インストールを確認するには、いくつかの方法があります。

を使用します `TridentOrchestrator` ステータス

のステータス `TridentOrchestrator` インストールが正常に完了したかどうかを示し、インストールされているTridentのバージョンが表示されます。インストール中、のステータス `TridentOrchestrator` からの変更 `Installing` 終了： `Installed`。を確認した場合は `Failed` ステータスとオペレータは単独で回復できません。 "ログをチェックしてください"。

ステータス	説明
インストール中です	このツールを使用してAstra Tridentをインストールしている `TridentOrchestrator` CR。
インストール済み	Astra Trident のインストールが完了しました。
アンインストール中です	OperatorはAstra Tridentをアンインストールしています。理由はです `spec.uninstall=true`。
アンインストール済み	Astra Trident がアンインストールされました。
失敗しました	オペレータがインストール、パッチ適用、アップデート、またはアンインストールできませんでした Astra Trident。オペレータはこの状態からのリカバリを自動的に試行します。この状態が解消されない場合は、トラブルシューティングが必要です。
更新中です	オペレータが既存のインストールを更新しています。
エラー	。 `TridentOrchestrator` は使用されません。もう一つ 存在します。

ポッドの作成ステータスを使用する

作成したポッドのステータスを確認することで、Astra Tridentのインストールが完了したかどうかを確認できます。

```
kubectl get pods -n trident

NAME                                  READY   STATUS    RESTARTS
AGE
trident-controller-7d466bf5c7-v4cpw   6/6     Running   0
1m
trident-node-linux-mr6zc              2/2     Running   0
1m
trident-node-linux-xrp7w              2/2     Running   0
1m
trident-node-linux-zh2jt              2/2     Running   0
1m
trident-operator-766f7b8658-ldzsv     1/1     Running   0
3m
```

を使用します `tridentctl`

を使用できます `tridentctl` インストールされているAstra Tridentのバージョンを確認します。

```
./tridentctl -n trident version

+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 23.04.0        | 23.04.0        |
+----------------+----------------+
```

次のステップ

できるようになりました。 "バックエンドとストレージクラスを作成し、ボリュームをプロビジョニングして、ポッドにボリュームをマウントします"。

Helm（標準モード）を使用してTridentを導入

Tridentオペレータを導入し、Helmを使用してAstra Tridentをインストールできます。このプロセスでは、環境 をインストールする際に、Astra Tridentで必要なコンテナイメージがプライベートレジストリに格納されません。プライベートイメージレジストリがある場合は、を使用します "オフライン導入のプロセス"。

Astra Tridentに関する重要な情報23.04

- Astra Tridentに関する次の重要な情報をお読みください。*

** ：Trident ** に関する重要な情報

- TridentでKubernetes 1.27がサポートされるようになりました。Kubernetesをアップグレードする前にTridentをアップグレード
- Astra Tridentは、SAN環境でマルチパス構成を厳密に使用し、推奨される値をに設定します `find_multipaths: no` multipath.confファイル内。

  非マルチパス構成またはを使用 `find_multipaths: yes` または `find_multipaths: smart` multipath.confファイルの値が原因でマウントが失敗します。Tridentはの使用を推奨しています `find_multipaths: no` 21.07リリース以降

Tridentオペレータを導入し、Helmを使用してAstra Tridentをインストール

Tridentの使用 "Helmチャート" Tridentオペレータを導入し、Tridentを一度にインストールできます。

レビュー "インストールの概要" インストールの前提条件を満たし、環境に適したインストールオプションを選択していることを確認します。

作業を開始する前に

に加えて "導入の前提条件" 必要です "Helm バージョン 3"。

手順

1. Astra Trident Helmリポジトリを追加します。

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

2. 使用 `helm install` をクリックし、次の例に示すように、導入環境の名前を指定します `23.04.0` は、インストールするAstra Tridentのバージョンです。

```
helm install <name> netapp-trident/trident-operator --version 23.04.0
--create-namespace --namespace <trident-namespace>
```

> Tridentのネームスペースを作成済みの場合は、を参照してください `--create-namespace` パラメータでネームスペースが追加で作成されることはありません。

を使用できます `helm list` 名前、ネームスペース、グラフ、ステータス、アプリケーションバージョンなどのインストールの詳細を確認するには、次の手順を実行します。とリビジョン番号。

インストール中に設定データを渡す

インストール中に設定データを渡すには、次の 2 つの方法があります。

オプション	説明
`--values` （または `-f`）	オーバーライドを使用してYAMLファイルを指定します。これは複数回指定でき、右端のファイルが優先されます。
`--set`	コマンドラインでオーバーライドを指定します。

たとえば、のデフォルト値を変更するには、のように指定します `debug`をクリックし、次のコマンドを実行します `--set` コマンドを入力します `23.04.0` は、インストールするAstra Tridentのバージョンです。

```
helm install <name> netapp-trident/trident-operator --version 23.04.0
--create-namespace --namespace --set tridentDebug=true
```

設定オプション

このテーブルと `values.yaml` Helmチャートの一部であるファイルには、キーとそのデフォルト値のリストが表示されます。

オプション	説明	デフォルト
`nodeSelector`	ポッド割り当てのノードラベル	
`podAnnotations`	ポッドの注釈	
`deploymentAnnotations`	配置のアノテーション	
`tolerations`	ポッド割り当ての許容値	
`affinity`	ポッド割り当てのアフィニティ	
`tridentControllerPluginNodeSelector`	ポッド用の追加のノードセレクタ。を参照してください [コントローラポッドとノードポッドについて] を参照してください。	
`tridentControllerPluginTolerations`	ポッドに対するKubernetesの許容範囲を上書きします。を参照してください [コントローラポッドとノードポッドについて] を参照してください。	
`tridentNodePluginNodeSelector`	ポッド用の追加のノードセレクタ。を参照してください [コントローラポッドとノードポッドについて] を参照してください。	
`tridentNodePluginTolerations`	ポッドに対するKubernetesの許容範囲を上書きします。を参照してください [コントローラポッドとノードポッドについて] を参照してください。	

オプション	説明	デフォルト
imageRegistry	のレジストリを指定します trident-operator、trident、およびその他の画像。デフォルトをそのまま使用する場合は、空のままにします。	""
imagePullPolicy	のイメージプルポリシーを設定します trident-operator。	IfNotPresent
imagePullSecrets	のイメージプルシークレットを設定します trident-operator、trident、およびその他の画像。	
kubeletDir	kubeletの内部状態のホスト位置を上書きできます。	'"/var/lib/kubeelet"'
operatorLogLevel	Tridentオペレータのログレベルを次のように設定できます。trace、 debug、 info、 warn、error`または `fatal。	"info"
operatorDebug	Tridentオペレータのログレベルをdebugに設定できます。	true
operatorImage	のイメージを完全に上書きできます trident-operator。	""
operatorImageTag	のタグを上書きできます trident-operator イメージ（Image）：	""
tridentIPv6	IPv6クラスタでAstra Tridentを動作させることができます。	false
tridentK8sTimeout	ほとんどのKubernetes API処理でデフォルトの30秒タイムアウトを上書きします（0以外の場合は秒単位）。	0
tridentHttpRequestTimeout	HTTP要求のデフォルトの90秒タイムアウトをで上書きします 0s タイムアウトの期間は無限です。負の値は使用できません。	"90s"
tridentSilenceAutosupport	Astra Tridentの定期的なAutoSupport レポートを無効にできます。	false
tridentAutosupportImageTag	Astra Trident AutoSupport コンテナのイメージのタグを上書きできます。	<version>
tridentAutosupportProxy	Astra TridentのAutoSupport コンテナがHTTPプロキシ経由で自宅に通信できるようになります。	""
tridentLogFormat	Astra Tridentのログ形式を設定します (text または json）。	"text"

オプション	説明	デフォルト
`tridentDisableAuditLog`	Astra Trident監査ロガーを無効にします。	`true`
`tridentLogLevel`	Astra Tridentのログレベルを次のように設定できます。 `trace`、`debug`、 `info`、 `warn`、 `error`` または `fatal`。	`"info"`
`tridentDebug`	Astra Tridentのログレベルをに設定できます `debug`。	`false`
`tridentLogWorkflows`	特定のAstra Tridentワークフローを有効にして、トレースロギングやログ抑制を実行できます。	""
`tridentLogLayers`	特定のAstra Tridentレイヤでトレースロギングやログ抑制を有効にできます。	""
`tridentImage`	Astra Tridentのイメージを完全に上書きできます。	""
`tridentImageTag`	Astra Tridentのイメージのタグを上書きできます。	""
`tridentProbePort`	Kubernetesの活性/準備プローブに使用されるデフォルトポートを上書きできます。	""
`windows`	WindowsワーカーノードにAstra Tridentをインストールできます。	`false`
`enableForceDetach`	強制切り離し機能を有効にできます。	`false`
`excludePodSecurityPolicy`	オペレータポッドのセキュリティポリシーを作成から除外します。	`false`

コントローラポッドとノードポッドについて

Astra Tridentは、単一のコントローラポッドと、クラスタ内の各ワーカーノード上のノードポッドとして実行されます。Astra Tridentボリュームをマウントするすべてのホストでノードポッドが実行されている必要があります。

Kubernetes "ノードセレクタ" および "寛容さと汚れ" は、特定のノードまたは優先ノードで実行されるようにポッドを制限するために使用されます。「ControllerPlugin」およびを使用します `NodePlugin`を使用すると、拘束とオーバーライドを指定できます。

- コントローラプラグインは、Snapshotやサイズ変更などのボリュームのプロビジョニングと管理を処理します。
- ノードプラグインによって、ノードへのストレージの接続が処理されます。

次のステップ

できるようになりました。 "バックエンドとストレージクラスを作成し、ボリュームをプロビジョニングし

て、ポッドにボリュームをマウントします"。

Helm（オフラインモード）を使用したTridentのオペレータの導入

Tridentオペレータを導入し、Helmを使用してAstra Tridentをインストールできます。このプロセスでは、環境 をインストールする際に、Astra Tridentで必要なコンテナイメージがプライベートレジストリに格納されます。プライベートイメージレジストリがない場合は、を使用します "標準的な導入のプロセス"。

Astra Tridentに関する重要な情報23.04

- Astra Tridentに関する次の重要な情報をお読みください。*

** ：Trident ** に関する重要な情報

- TridentでKubernetes 1.27がサポートされるようになりました。Kubernetesをアップグレードする前にTridentをアップグレード
- Astra Tridentは、SAN環境でマルチパス構成を厳密に使用し、推奨される値をに設定します `find_multipaths: no` multipath.confファイル内。

  非マルチパス構成またはを使用 `find_multipaths: yes` または `find_multipaths: smart` multipath.confファイルの値が原因でマウントが失敗します。Tridentはの使用を推奨しています `find_multipaths: no` 21.07リリース以降

Tridentオペレータを導入し、Helmを使用してAstra Tridentをインストール

Tridentの使用 "Helmチャート" Tridentオペレータを導入し、Tridentを一度にインストールできます。

レビュー "インストールの概要" インストールの前提条件を満たし、環境に適したインストールオプションを選択していることを確認します。

作業を開始する前に

に加えて "導入の前提条件" 必要です "Helm バージョン 3"。

手順

1. Astra Trident Helmリポジトリを追加します。

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

2. 使用 `helm install` 展開およびイメージレジストリの場所の名前を指定します。。 "TridentとCSIの画像" 1つのレジストリまたは別のレジストリに配置できますが、すべてのCSIイメージは同じレジストリに配置する必要があります。例では、 `23.04.0` は、インストールするAstra Tridentのバージョンです。

1つのレジストリ内のイメージ

```
helm install <name> netapp-trident/trident-operator --version
23.04.0 --set imageRegistry=<your-registry> --create-namespace
--namespace <trident-namespace>
```

異なるレジストリ内の画像

を追加する必要があります `sig-storage` に移動します `imageRegistry` 別のレジストリの場所を使用します。

```
helm install <name> netapp-trident/trident-operator --version
23.04.0 --set imageRegistry=<your-registry>/sig-storage --set
operatorImage=<your-registry>/netapp/trident-operator:23.04.0 --set
tridentAutosupportImage=<your-registry>/netapp/trident-
autosupport:23.04 --set tridentImage=<your-
registry>/netapp/trident:23.04.0 --create-namespace --namespace
<trident-namespace>
```

Tridentのネームスペースを作成済みの場合は、を参照してください `--create -namespace` パラメータでネームスペースが追加で作成されることはありません。

を使用できます `helm list` 名前、ネームスペース、グラフ、ステータス、アプリケーションバージョンなどのインストールの詳細を確認するには、次の手順を実行します。とリビジョン番号。

インストール中に設定データを渡す

インストール中に設定データを渡すには、次の 2 つの方法があります。

オプション	説明
`--values` （または `-f`)	オーバーライドを使用してYAMLファイルを指定します。これは複数回指定でき、右端のファイルが優先されます。
`--set`	コマンドラインでオーバーライドを指定します。

たとえば、のデフォルト値を変更するには、のように指定します `debug`をクリックし、次のコマンドを実行します `--set` コマンドを入力します `23.04.0` は、インストールするAstra Tridentのバージョンです。

```
helm install <name> netapp-trident/trident-operator --version 23.04.0
--create-namespace --namespace --set tridentDebug=true
```

設定オプション

このテーブルと `values.yaml` Helmチャートの一部であるファイルには、キーとそのデフォルト値のリストが表示されます。

オプション	説明	デフォルト
`nodeSelector`	ポッド割り当てのノードラベル	
`podAnnotations`	ポッドの注釈	
`deploymentAnnotations`	配置のアノテーション	
`tolerations`	ポッド割り当ての許容値	
`affinity`	ポッド割り当てのアフィニティ	
`tridentControllerPluginNodeSelector`	ポッド用の追加のノードセレクタ。を参照してください [コントローラポッドとノードポッドについて] を参照してください。	
`tridentControllerPluginTolerations`	ポッドに対するKubernetesの許容範囲を上書きします。を参照してください [コントローラポッドとノードポッドについて] を参照してください。	
`tridentNodePluginNodeSelector`	ポッド用の追加のノードセレクタ。を参照してください [コントローラポッドとノードポッドについて] を参照してください。	
`tridentNodePluginTolerations`	ポッドに対するKubernetesの許容範囲を上書きします。を参照してください [コントローラポッドとノードポッドについて] を参照してください。	
`imageRegistry`	のレジストリを指定します `trident-operator`、`trident`、およびその他の画像。デフォルトをそのまま使用する場合は、空のままにします。	""
`imagePullPolicy`	のイメージプルポリシーを設定します `trident-operator`。	`IfNotPresent`
`imagePullSecrets`	のイメージプルシークレットを設定します `trident-operator`、`trident`、およびその他の画像。	
`kubeletDir`	kubeletの内部状態のホスト位置を上書きできます。	"/var/lib/kubeelet"
`operatorLogLevel`	Tridentオペレータのログレベルを次のように設定できます。`trace`、`debug`、`info`、`warn`、`error`または `fatal`。	`"info"`

オプション	説明	デフォルト
operatorDebug	Tridentオペレータのログレベルをdebugに設定できます。	true
operatorImage	のイメージを完全に上書きできます trident-operator。	""
operatorImageTag	のタグを上書きできます trident-operator イメージ（Image）：	""
tridentIPv6	IPv6クラスタでAstra Tridentを動作させることができます。	false
tridentK8sTimeout	ほとんどのKubernetes API処理でデフォルトの30秒タイムアウトを上書きします（0以外の場合は秒単位）。	0
tridentHttpRequestTimeout	HTTP要求のデフォルトの90秒タイムアウトをで上書きします 0s タイムアウトの期間は無限です。負の値は使用できません。	"90s"
tridentSilenceAutosupport	Astra Tridentの定期的なAutoSupport レポートを無効にできます。	false
tridentAutosupportImageTag	Astra Trident AutoSupport コンテナのイメージのタグを上書きできます。	<version>
tridentAutosupportProxy	Astra TridentのAutoSupport コンテナがHTTPプロキシ経由で自宅に通信できるようになります。	""
tridentLogFormat	Astra Tridentのログ形式を設定します (text または json）。	"text"
tridentDisableAuditLog	Astra Trident監査ロガーを無効にします。	true
tridentLogLevel	Astra Tridentのログレベルを次のように設定できます。 trace、 debug、 info、 warn、 error` または `fatal。	"info"
tridentDebug	Astra Tridentのログレベルをに設定できます debug。	false
tridentLogWorkflows	特定のAstra Tridentワークフローを有効にして、トレースロギングやログ抑制を実行できます。	""
tridentLogLayers	特定のAstra Tridentレイヤでトレースロギングやログ抑制を有効にできます。	""

オプション	説明	デフォルト
`tridentImage`	Astra Tridentのイメージを完全に上書きできます。	""
`tridentImageTag`	Astra Tridentのイメージのタグを上書きできます。	""
`tridentProbePort`	Kubernetesの活性/準備プローブに使用されるデフォルトポートを上書きできます。	""
`windows`	WindowsワーカーノードにAstra Tridentをインストールできます。	`false`
`enableForceDetach`	強制切り離し機能を有効にできます。	`false`
`excludePodSecurityPolicy`	オペレータポッドのセキュリティポリシーを作成から除外します。	`false`

コントローラポッドとノードポッドについて

Astra Tridentは、単一のコントローラポッドと、クラスタ内の各ワーカーノード上のノードポッドとして実行されます。Astra Tridentボリュームをマウントするすべてのホストでノードポッドが実行されている必要があります。

Kubernetes "ノードセレクタ" および "寛容さと汚れ" は、特定のノードまたは優先ノードで実行されるようにポッドを制限するために使用されます。「ControllerPlugin」およびを使用します `NodePlugin`を使用すると、拘束とオーバーライドを指定できます。

- コントローラプラグインは、Snapshotやサイズ変更などのボリュームのプロビジョニングと管理を処理します。
- ノードプラグインによって、ノードへのストレージの接続が処理されます。

次のステップ

できるようになりました。 "バックエンドとストレージクラスを作成し、ボリュームをプロビジョニングして、ポッドにボリュームをマウントします"。

Tridentオペレータのインストールをカスタマイズ

Tridentオペレータは、の属性を使用してAstra Tridentのインストールをカスタマイズできます `TridentOrchestrator` 仕様インストールをカスタマイズする場合は、それ以上のカスタマイズが必要です `TridentOrchestrator` 引数allow、使用を検討してください `tridentctl` 必要に応じて変更するカスタムYAMLマニフェストを生成します。

コントローラポッドとノードポッドについて

Astra Tridentは、単一のコントローラポッドと、クラスタ内の各ワーカーノード上のノードポッドとして実行されます。Astra Tridentボリュームをマウントするすべてのホストでノードポッドが実行されている必要があります。

Kubernetes "ノードセレクタ" および "寛容さと汚れ" は、特定のノードまたは優先ノードで実行されるようにポッドを制限するために使用されます。「ControllerPlugin」およびを使用します `NodePlugin`を使用すると、拘束とオーバーライドを指定できます。

- コントローラプラグインは、Snapshotやサイズ変更などのボリュームのプロビジョニングと管理を処理します。
- ノードプラグインによって、ノードへのストレージの接続が処理されます。

設定オプション

`spec.namespace` は、で指定します `TridentOrchestrator` Astra Tridentがインストールされているネームスペースを指定します。このパラメータ * は、 Astra Trident のインストール後に更新できません * 。これを実行すると、が実行されます `TridentOrchestrator` ステータスをに変更します `Failed`。Astra Tridentは、ネームスペース間での移行を意図していません。

このテーブルの詳細 `TridentOrchestrator` 属性。

パラメータ	説明	デフォルト
`namespace`	Astra Trident をインストールするネームスペース	デフォルト
`debug`	Astra Trident のデバッグを有効にします	いいえ
`enableForceDetach`	`ontap-san` および `ontap-san-economy` のみ。 KubernetesのNon-Graceful Node Shutdown（NGN）と連携して、ノードに障害が発生した場合に、マウントされたボリュームを含むワークロードを新しいノードに安全に移行する機能をクラスタ管理者に提供します。 *これは23.04の実験的な機能です。*を参照してください [フォースデタッチの詳細] を参照してください。	`false`
`windows`	をに設定します `true` Windowsワーカーノードへのインストールを有効にします。	いいえ
`useIPv6`	IPv6 経由の Astra Trident をインストール	いいえ
`k8sTimeout`	Kubernetes 処理のタイムアウト	30 秒
`silenceAutosupport`	AutoSupportバンドルをNetAppに送信しない 自動	いいえ

パラメータ	説明	デフォルト
autosupportImage	AutoSupport テレメトリのコンテナイメージ	"netapp/trident-autosupport：23.07"
autosupportProxy	AutoSupportを送信するためのプロキシのアドレス/ポート テレメータ	"http://proxy.example.com:8888"
uninstall	Astra Trident のアンインストールに使用するフラグ	いいえ
logFormat	Astra Trident のログ形式が使用 [text 、 JSON]	テキスト（ Text ）
tridentImage	インストールする Astra Trident イメージ	"NetApp/Trident：23.07"
imageRegistry	形式の内部レジストリへのパス <registry FQDN>[:port][/subpath]	「k8s.gcr.io/sig-storage」（k8s 1.19以降） または"quay.io/k8scsi"
kubeletDir	ホスト上の kubelet ディレクトリへのパス	"/var/lib/kubeelet"
wipeout	完全な削除を実行するために削除するリソースのリスト Astra Trident	
imagePullSecrets	内部レジストリからイメージをプルするシークレット	
imagePullPolicy	Tridentオペレータのイメージプルポリシーを設定します。有効な値は次のとおりです。 Always 常にイメージをプルする。 IfNotPresent ノード上にイメージが存在しない場合にのみ取得します。 Never 画像を絶対に引き出さないでください。	IfNotPresent
controllerPluginNodeSelector	ポッド用の追加のノードセレクタ。 の形式はと同じです pod.spec.nodeSelector。	デフォルトはありません。オプションです
controllerPluginTolerations	ポッドに対するKubernetesの許容範囲を上書きします。はと同じ形式です pod.spec.Tolerations。	デフォルトはありません。オプションです
nodePluginNodeSelector	ポッド用の追加のノードセレクタ。の形式はと同じです pod.spec.nodeSelector。	デフォルトはありません。オプションです

パラメータ	説明	デフォルト
nodePluginTolerations	ポッドに対するKubernetesの許容範囲を上書きします。はと同じ形式です pod.spec.Tolerations。	デフォルトはありません。オプションです

ポッドパラメータの書式設定の詳細については、を参照してください "ポッドをノードに割り当てます"。

フォースデタッチの詳細

では、[強制切り離し]を使用できます ontap-san および ontap-san-economy のみ。強制接続解除を有効にする前に、Kubernetesクラスタで非グレースフルノードシャットダウン（NGN）を有効にする必要があります。詳細については、を参照してください "Kubernetes：正常なノードシャットダウンではありません"。

Astra TridentはKubernetes NGNに依存しているため、削除しないでください out-of-service 許容できないすべてのワークロードが再スケジュールされるまで、正常でないノードから影響を受けます。汚染を無謀に適用または削除すると、バックエンドのデータ保護が危険にさらされる可能性があります。

Kubernetesクラスタ管理者がを適用したとき node.kubernetes.io/out-of-service=nodeshutdown:NoExecute ノードおよびに影響を与えます enableForceDetach がに設定されます `true`Astra Tridentがノードのステータスを判断し、次の処理を実行します。

1. そのノードにマウントされたボリュームのバックエンドI/Oアクセスを停止します。
2. Astra Tridentノードオブジェクトをにマークします dirty (新しい出版物には安全ではありません)。

Tridentコントローラは、（とマークされたあとに）ノードが再認定されるまで、新しいパブリッシュボリューム要求を拒否します dirty`をクリックします。マウントされたPVCでスケジュールされているワークロード（クラスタノードが正常で準備が完了したあとも含む）は、Astra Tridentがノードを検証できるまで受け入れられません `clean (新しい出版物のための安全)。

ノードの健常性が回復してtaintが削除されると、Astra Tridentは次の処理を実行します。

1. ノード上の古い公開パスを特定してクリーンアップします。
2. ノードがに含まれている場合 cleanable 状態（out-of-service taintが削除され、ノードがinになっています Ready 状態）。古い公開済みパスはすべてクリーンで、Astra Tridentはノードをとして再登録します clean 新しいボリュームのノードへの公開を許可します。

構成例

上記の属性は、を定義するときに使用できます TridentOrchestrator をクリックして、インストールをカスタマイズします。

例**1**：基本的なカスタム構成

次に、基本的なカスタム構成の例を示します。

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

例**2**：ノードセレクタを使用して導入します

次の例では、ノードセレクタを使用してTridentを導入する方法を示します。

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

例**3**：**Windows**ワーカーノードに導入する

この例は、Windowsワーカーノードへの導入を示しています。

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

著作権に関する情報

Copyright © 2026 NetApp, Inc. All Rights Reserved. Printed in the U.S.このドキュメントは著作権によって保護されています。著作権所有者の書面による事前承諾がある場合を除き、画像媒体、電子媒体、および写真複写、記録媒体、テープ媒体、電子検索システムへの組み込みを含む機械媒体など、いかなる形式および方法による複製も禁止します。

ネットアップの著作物から派生したソフトウェアは、次に示す使用許諾条項および免責条項の対象となります。

このソフトウェアは、ネットアップによって「現状のまま」提供されています。ネットアップは明示的な保証、または商品性および特定目的に対する適合性の暗示的保証を含み、かつこれに限定されないいかなる暗示的な保証も行いません。ネットアップは、代替品または代替サービスの調達、使用不能、データ損失、利益損失、業務中断を含み、かつこれに限定されない、このソフトウェアの使用により生じたすべての直接的損害、間接的損害、偶発的損害、特別損害、懲罰的損害、必然的損害の発生に対して、損失の発生の可能性が通知されていたとしても、その発生理由、根拠とする責任論、契約の有無、厳格責任、不法行為（過失またはそうでない場合を含む）にかかわらず、一切の責任を負いません。

ネットアップは、ここに記載されているすべての製品に対する変更を随時、予告なく行う権利を保有します。ネットアップによる明示的な書面による合意がある場合を除き、ここに記載されている製品の使用により生じる責任および義務に対して、ネットアップは責任を負いません。この製品の使用または購入は、ネットアップの特許権、商標権、または他の知的所有権に基づくライセンスの供与とはみなされません。

このマニュアルに記載されている製品は、1つ以上の米国特許、その他の国の特許、および出願中の特許によって保護されている場合があります。

権利の制限について：政府による使用、複製、開示は、DFARS 252.227-7013（2014年2月）およびFAR 5252.227-19（2007年12月）のRights in Technical Data -Noncommercial Items（技術データ - 非商用品目に関する諸権利）条項の(b)(3)項、に規定された制限が適用されます。

本書に含まれるデータは商用製品および / または商用サービス（FAR 2.101の定義に基づく）に関係し、データの所有権はNetApp, Inc.にあります。本契約に基づき提供されるすべてのネットアップの技術データおよびコンピュータ ソフトウェアは、商用目的であり、私費のみで開発されたものです。米国政府は本データに対し、非独占的かつ移転およびサブライセンス不可で、全世界を対象とする取り消し不能の制限付き使用権を有し、本データの提供の根拠となった米国政府契約に関連し、当該契約の裏付けとする場合にのみ本データを使用できます。前述の場合を除き、NetApp, Inc.の書面による許可を事前に得ることなく、本データを使用、開示、転載、改変するほか、上演または展示することはできません。国防総省にかかる米国政府のデータ使用権については、DFARS 252.227-7015(b)項（2014年2月）で定められた権利のみが認められます。

商標に関する情報

NetApp、NetAppのロゴ、http://www.netapp.com/TMに記載されているマークは、NetApp, Inc.の商標です。その他の会社名と製品名は、それを所有する各社の商標である場合があります。