Astra Trident をアップグレード

Astra Trident

NetApp
November 14, 2025

目次

Astra Trident をアップグレード

Astra Trident をアップグレード

Astra Trident は四半期ごとにリリースサイクルを実施し、毎年 4 つのメジャーリリースをリリースしています。各新しいリリースは、以前のリリースに基づいてビルドされ、新機能とパフォーマンスの強化に加え、バグの修正や改善点が追加されています。ネットアップでは、Astra Tridentの新機能を活用するために、1年に1回以上アップグレードすることを推奨しています。

バージョンを選択します

Astra Tridentバージョンは日付ベースです `YY.MM` 命名規則。「YY」は年の最後の2桁、「MM」は月です。ドットリリースは、の後に続きます `YY.MM.X` 条約。ここで、「X」はパッチレベルです。アップグレード前のバージョンに基づいて、アップグレード後のバージョンを選択します。

- インストールされているバージョンの4リリースウィンドウ内にある任意のターゲットリリースに直接アップグレードできます。たとえば、22.01から23.01に直接アップグレードできます(22.01.1などのドットリリースを含む)。
- 以前のリリースを使用している場合は、具体的な手順について、該当するリリースのドキュメントを参照してアップグレードを実行してください。そのためには、最初に 4 つのリリースウィンドウに対応する最新リリースにアップグレードする必要があります。たとえば'18.07を実行していて'20.07リリースにアップグレードする場合は'次のように複数ステップのアップグレードプロセスを実行します
  a. 最初のアップグレードは 18.07 から 19.07 へ。
  b. その後 '19.07 から 20.07 にアップグレードします

- バージョン19.04以前のアップグレードでは、Astra TridentメタデータをIT所有から移行する必要があります `etcd` をCRDオブジェクトに追加します。リリースのマニュアルを参照して、アップグレードの仕組みを確認してください。
- アップグレードするときは、この作業を行うことが重要です `parameter.fsType` インチ `StorageClasses` Astra Tridentが使用。削除して再作成することができます `StorageClasses` 実行前のボリュームの中断はなし。これは、SANボリュームに対してhttps://kubernetes.io/docs/tasks/configure-pod-container/security-context/[securityコンテキスト^]を適用するための要件です。https://github.com/NetApp/trident/tree/master/trident-installer/sample-input[sample input^]ディレクトリには、https://github.com/NetApp/trident/blob/master/trident-installer/sample-input/storage-class-samples/storage-class-basic.yaml.templなどの例が含まれています[`storage-class-basic.yaml.templ`]）とリンク：https://github.com/NetApp/trident/blob/master/trident-installer/sample-input/storage-class-samples/storage-class-bronze-default.yaml[`storage-class-bronze-default.yaml`^]をクリックします。詳細については、を参照してください "既知の問題"。

アップグレードオプションを選択します

Tridentをアップグレードする方法は2つあります。通常は、初期インストールに使用したものと同じオプションを使用しますが、使用することもできます "インストール方法を切り替えます"。

- "Tridentオペレータを使用してアップグレード"
  *

CSI のボリュームスナップショットは、 Kubernetes 1.20 以降の GA 機能になりました。Astra Trident をアップグレードする場合、アップグレードを実行する前に、以前のスナップショット CRS と CRD （ボリューム Snapshot クラス、ボリューム Snapshot 、ボリューム Snapshot コンテンツ）をすべて削除する必要があります。を参照してください "この blog" アルファスナップショットを beta/GA 仕様に移行する手順を理解する。

演算子に変更があります

Astra Trident の 21.01 リリースでは、アーキテクチャに関する次のような重要な変更がオペレータに導入されています。

- 演算子は * cluster を対象とした * になりました。Trident 演算子の以前のインスタンス（バージョン 20.04 ～ 20.10 ）は、 * 名前空間スコープ * でした。クラスタを対象としたオペレータが有利な理由は次のとおりです。
  - リソースのアカウンタビリティ：オペレータは、 Astra Trident インストールに関連付けられたリソースをクラスタレベルで管理するようになりました。Astra Tridentのインストールの一環として、オペレータはを使用して複数のリソースを作成し、管理します `ownerReferences`。メンテナンス `ownerReferences` クラスタを対象としたリソースでは、OpenShiftなどの特定のKubernetesディストリビュータでエラーが発生する可能性があります。これは、クラスタを対象としたオペレータによって緩和されます。Trident リソースの自動修復とパッチ適用には、この要件が不可欠です。
  - アンインストール中のクリーンアップ： Astra Trident を完全に削除するには、関連するリソースをすべて削除する必要があります。ネームスペースを対象としたオペレータが、クラスタを対象としたリソース（ clusterRole 、 ClusterRoleBinding 、 PodSecurityPolicy など）の削除で問題が発生し、クリーンアップが完了しない場合があります。クラスタを対象としたオペレータがこの問題を排除し、必要に応じて、 Astra Trident を完全にアンインストールし、 Aresh をインストールできます。
- `TridentProvisioner` がに置き換えられました `TridentOrchestrator` Astra Tridentのインストールと管理に使用したカスタムリソース。また、に新しいフィールドが導入されます `TridentOrchestrator` 仕様Tridentのネームスペースは、を使用してからインストールまたはアップグレードするように指定できます `spec.namespace` フィールド。例を見てみましょう "こちらをご覧ください"。

オペレータにアップグレードしてください

既存の Astra Trident インストールは、オペレータが簡単にアップグレードできます。

作業を開始する前に

オペレータを使用してアップグレードするには、次の条件を満たしている必要があります。

- CSIベースのAstra Tridentがインストールされている必要があります。の19.07以降のすべてのリリースはCSIベースです。Tridentネームスペース内のポッドを調べて確認できます。
  - 23.01より前のバージョンのポッドの命名は、の後に続きます `trident-csi-*` 表記規則
  - 23.01以降でポッドの命名には次のものが使用されます。 `trident-controller-<generated id>` コントローラポッド用 `trident-node-<operating system>-<generated id>` ノードポッド用 `trident-operator-<generated id>` オペレータポッド用。
- CSI Trident をアンインストールしても、インストールからのメタデータが保持されている場合は、オペレータを使用してアップグレードできます。
- 特定の Kubernetes クラスタ内のすべてのネームスペースに存在する Trident のは、 1 つの Astra だけで

す。

- Kubernetesクラスタを使用して実行する必要があります "サポートされるKubernetesバージョン"。
- アルファスナップショットのCRDが存在する場合は、で削除する必要があります `tridentctl obliviate alpha-snapshot-crd`。これにより、アルファスナップショット仕様の CRD が削除されます。削除または移行が必要な既存のスナップショットについては、を参照してください "この blog"。

> - OpenShift Container Platformで演算子を使用してTridentをアップグレードする場合は、Trident 21.01.1以降にアップグレードする必要があります。21.01.0 でリリースされた Trident オペレータには、 21.01.1 で修正された既知の問題が含まれています。詳細については、を参照してください "GitHub の問題の詳細"。
> - を使用している場合は、Tridentのアップグレードにオペレータを使用しないでください `etcd`- Tridentリリース（19.04以前）。

クラスタを対象とした**Trident**オペレータ環境をアップグレード

クラスタを対象としたTridentのオペレータ環境をアップグレードする手順は、次のとおりです。すべてのAstra Tridentバージョン21.01以降では、クラスタを対象とした演算子を使用します。

手順

1. Astra Tridentのバージョンを確認します。

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

2. 現在の Astra Trident インスタンスのインストールに使用した Trident オペレータを削除たとえば、22.01からアップグレードする場合は、次のコマンドを実行します。

```
kubectl delete -f 22.01/trident-installer/deploy/bundle.yaml -n trident
```

3. を使用して初期インストールをカスタマイズした場合 `TridentOrchestrator` 属性を編集できます `TridentOrchestrator` インストールパラメータを変更するオブジェクト。これには、ミラーリングされたTridentおよびCSIイメージレジストリをオフラインモードに指定したり、デバッグログを有効にしたり、イメージプルシークレットを指定したりするための変更が含まれます。
4. 環境に適したバンドルYAMLファイルとAstra Tridentバージョンを使用してAstra Tridentをインストールします。たとえば、Kubernetes 1.26用にAstra Trident 23.01をインストールする場合は、次のコマンドを実行します。

```
kubectl create -f 23.01.1/trident-installer/deploy/bundle_post_1_25.yaml
-n trident
```

Tridentでは、オペレータのインストールやKubernetesバージョンに関連するオブジェクトの作成に使用できるバンドルファイルが提供されています。

- Kubernetes 1.24以前を実行しているクラスタの場合は、を使用します "Bundle_pre_1_25.yaml"。
- Kubernetes 1.25以上を実行するクラスタの場合は、を使用します "bundle_post_1_25.yaml"。

結果

Tridentのオペレータが、既存のAstra Tridentインストールを特定し、オペレータと同じバージョンにアップグレードします。

名前空間を対象としたオペレータインストールをアップグレードします

名前空間を対象とした演算子（バージョン20.07～20.10）を使用してインストールされたAstra Tridentのインスタンスからアップグレードするには、次の手順を実行します。

手順

1. 既存の Trident インストールのステータスを確認そのためには、の*ステータス*を確認してください `TridentProvisioner`。ステータスがになっている必要があります `Installed`。

```
kubectl describe tprov trident -n trident | grep Message: -A 3
Message:  Trident installed
Status:   Installed
Version:  v20.10.1
```

ステータスがになっている場合 `Updating`をクリックし、続行する前に解決してください。可能なステータス値のリストについては、を参照してください "こちらをご覧ください"。

2. を作成します `TridentOrchestrator` Tridentインストーラに付属のマニフェストを使用したCRD。

```
# Download the release required [23.01.1]
mkdir 23.01.1
cd 23.01.1
wget
https://github.com/NetApp/trident/releases/download/v23.01.1/trident-
installer-23.01.1.tar.gz
tar -xf trident-installer-23.01.1.tar.gz
cd trident-installer
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
```

3. マニフェストを使用して、名前空間を対象とした演算子を削除します。この手順を完了するには、名前空

間を対象とした演算子をから配備するために使用するバンドルYAMLファイルが必要です `https://github.com/NetApp/trident/tree/stable/vXX.XX/deploy/BUNDLE.YAML` ここで、 `vXX.XX` は、バージョン番号およびです `BUNDLE.YAML` はバンドルYAMLファイル名です。

Tridentのインストールパラメータに必要な変更を加えます（の値の変更など） `tridentImage`、 `autosupportImage`、プライベートイメージリポジトリ、および提供 `imagePullSecrets`)名前空間を対象とした演算子を削除した後、クラスタを対象とした演算子をインストールする前。更新可能なパラメータの一覧については、を参照してください "設定オプション"。

```
#Ensure you are in the right directory
pwd
/root/20.10.1/trident-installer

#Delete the namespace-scoped operator
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

#Confirm the Trident operator was removed
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
```

この段階では、を実行します `trident-operator-xxxxxxxxxx-xxxxx` ポッドが削除されました。

4. （オプション）インストールパラメータを変更する必要がある場合は、を更新します `TridentProvisioner` 仕様これらの変更には、コンテナイメージをからプルするためのプライベートイメージレジストリの変更、デバッグログの有効化、イメージプルシークレットの指定などがあります。

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

5. Tridentオペレータをインストール

クラスタを対象としたオペレータをインストールすると、の移行が開始されます `TridentProvisioner` オブジェクトの移動先 `TridentOrchestrator` オブジェクトを削除します `TridentProvisioner` オブジェクトと `tridentprovisioner` CRD、およびAstra Tridentを、使用しているクラスタ対象オペレータのバージョンにアップグレードします。次の例では、Tridentが23.01.1にアップグレードされています。

Tridentオペレータを使用してAstra Tridentをアップグレードすると、が移行されます `tridentProvisioner` をに追加します `tridentOrchestrator` 同じ名前のオブジェクト。これは、オペレータによって自動的に処理されます。アップグレードの際には、 Astra Trident が以前と同じネームスペースにインストールされる予定です。

```
#Ensure you are in the correct directory
pwd
/root/23.01.1/trident-installer

#Install the cluster-scoped operator in the **same namespace**
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

#Examine Trident pods in the namespace
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

#Confirm Trident has been updated to the desired version
kubectl describe torc trident | grep Message -A 3
Message:                Trident installed
Namespace:              trident
Status:                 Installed
Version:                v23.01.1
```

。 `trident-controller` ポッド名は、23.01で導入された命名規則を反映しています。

Helm ベースのオペレータインストレーションをアップグレードします

Helm ベースのオペレータインストレーションをアップグレードするには、次の手順を実行します。

> Astra Tridentがインストールされ ているKubernetesクラスタを1.24から1.25以降にアップグレードする場合は、value.yamlを更新して設定する必要があります `excludePodSecurityPolicy` 終了： `true` または、を追加します `--set excludePodSecurityPolicy=true` に移動します `helm upgrade` コマンドを実行してからクラスタをアップグレードしてください。

手順

1. 最新の Astra Trident リリースをダウンロード
2. を使用します `helm upgrade` コマンドを入力します `trident-operator-23.01.1.tgz` アップグレード後のバージョンが反映されます。

```
helm upgrade <name> trident-operator-23.01.1.tgz
```

> 初期インストール時にデフォルト以外のオプションを設定した場合（Tridentイメージおよびイメージのプライベートなミラーレジストリを指定するなど）は、を使用します `--set` これらのオプションがupgradeコマンドに含まれるようにするため、それらのオプションの値をdefaultにリセットします。
>
> 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 `tridentDebug`を使用して、次のコマンドを実行します。
>
> ```
> helm upgrade <name> trident-operator-23.01.1-custom.tgz --set tridentDebug=true
> ```

3. を実行します `helm list` グラフとアプリのバージョンが両方ともアップグレードされていることを確認します。を実行します `tridentctl logs` デバッグメッセージを確認します。

結果

Tridentのオペレータが、既存のAstra Tridentインストールを特定し、オペレータと同じバージョンにアップグレードします。

オペレータ以外のインストールからアップグレードします

からTridentの最新リリースにアップグレードできます `tridentctl` インストール：

手順

1. 最新の Astra Trident リリースをダウンロード

```
# Download the release required [23.01.1]
mkdir 23.01.1
cd 23.01.1
wget
https://github.com/NetApp/trident/releases/download/v22.01.1/trident-
installer-23.01.1.tar.gz
tar -xf trident-installer-23.01.1.tar.gz
cd trident-installer
```

2. を作成します `tridentorchestrator` マニフェストからのCRD。

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

3. オペレータを配備します。

```
#Install the cluster-scoped operator in the **same namespace**
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. を作成します `TridentOrchestrator` Astra Tridentのインストール用にCR。

```
#Create a tridentOrchestrator to initiate a Trident install
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

#Confirm Trident was upgraded to the desired version
kubectl describe torc trident | grep Message -A 3
Message:                Trident installed
Namespace:              trident
Status:                 Installed
Version:                v23.01.1
```

結果

既存のバックエンドと PVC は自動的に使用可能

tridentctl を使用してアップグレードします

を使用すると、既存のAstra Tridentインストールを簡単にアップグレードできます `tridentctl`。

アップグレード前の考慮事項

最新リリースの Astra Trident にアップグレードする際は、次の点を考慮してください。

- Trident 20.01 以降では、のベータ版のみが提供されます "ボリューム Snapshot" がサポートされます。Kubernetes 管理者は、従来のアルファスナップショットを保持するために、アルファスナップショットオブジェクトを安全にバックアップするか、ベータ版に変換するように注意する必要があります。
- ボリュームスナップショットのベータリリースでは、一連の新しい CRD とスナップショットコントローラが導入されています。どちらも Astra Trident をインストールする前にセットアップする必要があります。 "この blog" alpha ボリュームの Snapshot をベータ版に移行する手順について説明します。
- Astra Trident のアンインストールと再インストールはアップグレードとして機能します。Trident をアンイ

ンストールしても、 Astra Trident 環境で使用されている Persistent Volume Claim （ PVC ；永続的ボリューム要求）と Persistent Volume （ PV ；永続的ボリューム）は削除されません。Astra Trident がオフラインの間は、すでにプロビジョニング済みの PVS を引き続き使用でき、 Astra Trident は、オンラインに戻った時点で作成された PVC に対してボリュームをプロビジョニングします。

Astra Trident をアップグレードするときは、アップグレードプロセスを中断しないでください。インストーラが実行されていることを確認します。

アップグレード後の次の手順

新しいTridentリリース（On-Demand Volume Snapshotsなど）で利用できる豊富な機能を活用するには、を使用してボリュームをアップグレードします `tridentctl upgrade` コマンドを実行します

レガシーボリュームがある場合は、それらのボリュームを NFS/iSCSI タイプから CSI タイプにアップグレードして、 Astra Trident のすべての新機能を使用できるようにする必要があります。Trident によってプロビジョニングされたレガシー PV は、従来の機能セットをサポートします。

CSI タイプにボリュームをアップグレードする場合は、次の点を考慮してください。

- 場合によっては、すべてのボリュームをアップグレードする必要はありません。以前に作成したボリュームには引き続きアクセスでき、正常に機能します。
- PV は、アップグレード時に展開 / 起動可能セットの一部としてマウントできます。展開 / 起動セットを停止する必要はありません。
- アップグレード時に、スタンドアロンの POD に PV を接続することはできません。ボリュームをアップグレードする前に、ポッドをシャットダウンする必要があります。
- アップグレードできるのは、 PVC にバインドされているボリュームだけです。PVC にバインドされていないボリュームは、アップグレード前に削除およびインポートする必要があります。

ボリュームのアップグレードの例

次の例は、ボリュームのアップグレードを実行する方法を示しています。

1. を実行します `kubectl get pv` をクリックしてPVSをリスト表示します。

```
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
```

現在、Trident 20.07によって作成されたPVSのうちの4つが、を使用しています `netapp.io/trident` プロビジョニング担当者：

2. を実行します `kubectl describe pv` PVの詳細を確認します。

```
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はを使用して作成されました `netapp.io/trident` プロビジョニング担当者とプロビジョニングタイプはNFSです。Astra Trident が提供する新機能をすべてサポートするには、この PV を CSI タイプにアップグレードする必要があります。

3. を実行します `tridentctl upgrade volume <name-of-trident-volume>` 従来のAstra Tridentボリュームを**CSI**仕様にアップグレードするコマンド。

```
./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. を実行します `kubectl describe pv` ボリュームがCSIボリュームであることを確認します。

```
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>
```

このようにして、 Astra Trident によって作成された NFS/iSCSI タイプのボリュームを、ボリューム単位で CSI タイプにアップグレードできます。

著作権に関する情報

商標に関する情報