NetApp

Upgrade Astra Trident

Astra Trident

NetApp
March 17, 2023

Table of Contents

Upgrade Astra Trident

Upgrade Astra Trident

Astra Trident follows a quarterly release cadence, delivering four major releases every calendar year. Each new release builds on top of the previous releases, providing new features and performance enhancements as well as bug fixes and improvements. We encourage you to upgrade at least once a year to take advantage of the new features in Astra Trident.

Select a version

Astra Trident versions follow a date-based `YY.MM` naming convention, where "YY" is the last two digits of the year and "MM" is the month. Dot releases follow a `YY.MM.X` convention, where "X" is the patch level. You will select the version to upgrade to based on the version you are upgrading from.

- You can perform a direct upgrade to any target release that is within a four-release window of your installed version. For example, you can upgrade to 23.01 from 22.01 (including any dot releases, such as 22.01.1) directly.

- If you have an earlier release, you should perform a multi-step upgrade using the documentation of the respective release for specific instructions. This requires you to first upgrade to the most recent release that fits your four-release window. For example, if you are running 18.07 and want to upgrade to the 20.07 release, then follow the multi-step upgrade process as given below:

  1. First upgrade from 18.07 to 19.07.

  2. Then upgrade from 19.07 to 20.07.

> ⚠️
> - All upgrades for versions 19.04 and earlier require the migration of Astra Trident metadata from it own `etcd` to CRD objects. Ensure you check the documentation of the release to understand how the upgrade works.
>
> - When upgrading, it is important you provide `parameter.fsType` in `StorageClasses` used by Astra Trident. You can delete and re-create `StorageClasses` without disrupting pre-existing volumes. This is a **requirement** for enforcing security contexts for SAN volumes. The sample input directory contains examples, such as `storage-class-basic.yaml.templ` and `storage-class-bronze-default.yaml`. For more information, see Known Issues.

Select an upgrade option

There are two options to upgrade Astra Trident. Generally, you will use the same option you used for the initial installation, however you can move between installation methods.

- Upgrade using the Trident operator
- Upgrade using `tridentctl`

> ⚠ CSI Volume Snapshots is now a feature that is GA, beginning with Kubernetes 1.20. When upgrading Astra Trident, all previous alpha snapshot CRs and CRDs (Volume Snapshot Classes, Volume Snapshots and Volume Snapshot Contents) must be removed before the upgrade is performed. Refer to this blog to understand the steps involved in migrating alpha snapshots to the beta/GA spec.

Changes to the operator

The 21.01 release of Astra Trident introduces some key architectural changes to the operator, namely the following:

- The operator is now **cluster-scoped**. Previous instances of the Trident operator (versions 20.04 through 20.10) were **namespace-scoped**. An operator that is cluster-scoped is advantageous for the following reasons:

  - Resource accountability: The operator now manages resources associated with an Astra Trident installation at the cluster level. As part of installing Astra Trident, the operator creates and maintains several resources by using `ownerReferences`. Maintaining `ownerReferences` on cluster-scoped resources can throw up errors on certain Kubernetes distributors such as OpenShift. This is mitigated with a cluster-scoped operator. For auto-healing and patching Trident resources, this is an essential requirement.

  - Cleaning up during uninstallation: A complete removal of Astra Trident would require all associated resources to be deleted. A namespace-scoped operator might experience issues with the removal of cluster-scoped resources (such as the clusterRole, ClusterRoleBinding and PodSecurityPolicy) and lead to an incomplete clean-up. A cluster-scoped operator eliminates this issue. Users can completely uninstall Astra Trident and install afresh if needed.

- `TridentProvisioner` is now replaced with `TridentOrchestrator` as the Custom Resource used to install and manage Astra Trident. In addition, a new field is introduced to the `TridentOrchestrator` spec. Users can specify that the namespace Trident must be installed/upgraded from using the `spec.namespace` field. You can take a look at an example here.

Upgrade with the operator

You can easily upgrade an existing Astra Trident installation using the operator.

Before you begin

To upgrade using the operator, the following conditions should be met:

- You must have a CSI-based Astra Trident installation. All releases from 19.07 on are CSI-based. You can examine the pods in your Trident namespace to verify.

  - Pod naming in versions earlier than 23.01 follows a `trident-csi-*` convention.

  - Pod naming in 23.01 and later uses: `trident-controller-<generated id>` for the controller pod; `trident-node-<operating system>-<generated id>` for the node pods; `trident-operator-<generated id>` for the operator pod.

- If you have uninstalled CSI Trident and the metadata from the installation persists, you can upgrade by using the operator.

- Only one Astra Trident installation should exist across all the namespaces in a given Kubernetes cluster.

- You should be using a Kubernetes cluster running a supported Kubernetes version.

- If alpha snapshot CRDs are present, you should remove them with `tridentctl obliviate alpha-snapshot-crd`. This deletes the CRDs for the alpha snapshot spec. For existing snapshots that should be deleted/migrated, see this blog.

> ⚠
> - When upgrading Trident using the operator on OpenShift Container Platform, you should upgrade to Trident 21.01.1 or later. The Trident operator released with 21.01.0 contains a known issue that has been fixed in 21.01.1. For more details, see the issue details on GitHub.
> - Do not use the operator to upgrade Trident if you are using an `etcd`-based Trident release (19.04 or earlier).

Upgrade a cluster-scoped Trident operator installation

Follow these steps to upgrade a cluster-scoped Trident operator installation. All Astra Trident versions 21.01 and above use a cluster-scoped operator.

Steps

1. Delete the Trident operator that was used to install the current Astra Trident instance. For example, if you are upgrading from 22.01, run the following command:

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

2. If you customized your initial installation using `TridentOrchestrator` attributes, you can edit the `TridentOrchestrator` object to modify the installation parameters. This might include changes made to specify mirrored Trident and CSI image registries for offline mode, enable debug logs, or specify image pull secrets.

3. Install Astra Trident using the correct bundle YAML file for your environment and Astra Trident version. For example, if you are installing Astra Trident 23.01 for Kubernetes 1.26, run the following command:

```
kubectl create -f 23.01.0/trident-installer/deploy/bundle_post_1_25.yaml -n trident
```

> ⓘ
> Trident provides a bundle file that can be used to install the operator and create associated objects for your Kubernetes version.
> - For clusters running Kubernetes 1.24 or lower, use bundle_pre_1_25.yaml.
> - For clusters running Kubernetes 1.25 or higher, use bundle_post_1_25.yaml.

Results

The Trident operator will identify an existing Astra Trident installation and upgrade it to the same version as the operator.

Upgrade a namespace-scoped operator installation

Follow these steps to upgrade from an instance of Astra Trident installed using the namespace-scoped operator (versions 20.07 through 20.10).

Steps

1. Verify the status of the existing Trident installation. To do this, check the **Status** of `TridentProvisioner`. The status should be `Installed`.

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

ⓘ If status shows `Updating`, ensure you resolve it before proceeding. For a list of possible status values, see here.

2. Create the `TridentOrchestrator` CRD by using the manifest provided with the Trident installer.

```
# Download the release required [23.01.0]
mkdir 23.01.0
cd 23.01.0
wget
https://github.com/NetApp/trident/releases/download/v23.01.0/trident-
installer-23.01.0.tar.gz
tar -xf trident-installer-23.01.0.tar.gz
cd trident-installer
kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
```

3. Delete the namespace-scoped operator by using its manifest. To complete this step, you need the bundle YAML file used to deploy the namespace-scoped operator from `https://github.com/NetApp/trident/tree/stable/vXX.XX/deploy/BUNDLE.YAML` where `vXX.XX` is the version number and `BUNDLE.YAML` is the bundle YAML file name.

ⓘ You should make the necessary changes to the Trident install parameters (for example, changing the values for `tridentImage`, `autosupportImage`, private image repository, and providing `imagePullSecrets`) after deleting the namespace-scoped operator and before installing the cluster-scoped operator. For a complete list of parameters that can be updated, refer to the configuration options.

```
#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>
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
```

At this stage, the `trident-operator-xxxxxxxxxx-xxxxx` pod is deleted.

4. (Optional) If the install parameters need to be modified, update the `TridentProvisioner` spec. These could be changes such as modifying the private image registry to pull container images from, enabling debug logs, or specifying image pull secrets.

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

5. Install the Trident operator.

> ⓘ   Installing the cluster-scoped operator initiates the migration of `TridentProvisioner` objects to `TridentOrchestrator` objects, deletes `TridentProvisioner` objects and the `tridentprovisioner` CRD, and upgrades Astra Trident to the version of the cluster-scoped operator being used. In the example that follows, Trident is upgraded to 23.01.0.

> ⓘ   Upgrading Astra Trident using the Trident operator results in the migration of `tridentProvisioner` to a `tridentOrchestrator` object with the same name. This is automatically handled by the operator. The upgrade will also have Astra Trident installed in the same namespace as before.

```
#Ensure you are in the correct directory
pwd
/root/23.01.0/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:               v23.01.0
```

(i) The `trident-controller` and pod names reflect the naming convention introduced in 23.01.

Upgrade a Helm-based operator installation

Perform the following steps to upgrade a Helm-based operator installation.

> ⚠️ When upgrading a Kubernetes cluster from 1.24 to 1.25 or later that has Astra Trident installed, you must update values.yaml to set `excludePodSecurityPolicy` to `true` or add `--set excludePodSecurityPolicy=true` to the `helm upgrade` command before you can upgrade the cluster.

Steps

1. Download the latest Astra Trident release.

2. Use the `helm upgrade` command where `trident-operator-23.01.0.tgz` reflects the version that you want to upgrade to.

   ```
   helm upgrade <name> trident-operator-23.01.0.tgz
   ```

> ℹ️ If you set any non-default options during the initial installation (such as specifying private, mirrored registries for Trident and CSI images), use `--set` to ensure those options are included in the upgrade command, otherwise the values will reset to default.
>
> For example, to change the default value of `tridentDebug`, run the following command:
>
> ```
> helm upgrade <name> trident-operator-23.01.0-custom.tgz --set
> tridentDebug=true
> ```

3. Run `helm list` to verify that the chart and app version have both been upgraded. Run `tridentctl logs` to review any debug messages.

Results

The Trident operator will identify an existing Astra Trident installation and upgrade it to the same version as the operator.

Upgrade from a non-operator installation

You can upgrade to the latest release of the Trident operator from a `tridentctl` installation.

Steps

1. Download the latest Astra Trident release.

```
# Download the release required [23.01.0]
mkdir 23.01.0
cd 23.01.0
wget
https://github.com/NetApp/trident/releases/download/v22.01.0/trident-
installer-23.01.0.tar.gz
tar -xf trident-installer-23.01.0.tar.gz
cd trident-installer
```

2. Create the `tridentorchestrator` CRD from the manifest.

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

3. Deploy the operator.

```
#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. Create a `TridentOrchestrator` CR for installing Astra Trident.

```
#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:               v23.01.0
```

Results

The existing backends and PVCs are automatically available.

Upgrade with `tridentctl`

You can easily upgrade an existing Astra Trident installation using `tridentctl`.

Considerations before upgrading

When upgrading to the latest release of Astra Trident, consider the following:

- Starting with Trident 20.01, only the beta release of volume snapshots is supported. Kubernetes administrators should take care to safely back up or convert the alpha snapshot objects to beta to retain the legacy alpha snapshots.

- The beta release of volume snapshots introduces a modified set of CRDs and a snapshot controller, both of which should be set up before installing Astra Trident. This blog discusses the steps involved in migrating alpha volume snapshots to the beta format.

- Uninstalling and reinstalling Astra Trident acts as an upgrade. When you uninstall Trident, the Persistent Volume Claim (PVC) and Persistent Volume (PV) used by the Astra Trident deployment are not deleted. PVs that have already been provisioned will remain available while Astra Trident is offline, and Astra

Trident will provision volumes for any PVCs that are created in the interim once it is back online.

> ⓘ When upgrading Astra Trident, do not interrupt the upgrade process. Ensure that the installer runs to completion.

Next steps after upgrade

To make use of the rich set of features that are available in newer Trident releases (such as, On-Demand Volume Snapshots), you can upgrade the volumes by using the `tridentctl upgrade` command.

If there are legacy volumes, you should upgrade them from a NFS/iSCSI type to the CSI type to be able to use the complete set of new features in Astra Trident. A legacy PV that has been provisioned by Trident supports the traditional set of features.

Consider the following when deciding to upgrade volumes to the CSI type:

- You might not need to upgrade all the volumes. Previously created volumes will continue to be accessible and function normally.
- A PV can be mounted as part of a deployment/StatefulSet when upgrading. It is not required to bring down the deployment/StatefulSet.
- You **cannot** attach a PV to a standalone pod when upgrading. You should shut down the pod before upgrading the volume.
- You can upgrade only a volume that is bound to a PVC. Volumes that are not bound to PVCs should be removed and imported before upgrading.

Volume upgrade example

Here is an example that shows how a volume upgrade is performed.

1. Run `kubectl get pv` to list the PVs.

```
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
```

There are currently four PVs that have been created by Trident 20.07, using the `netapp.io/trident` provisioner.

2. Run `kubectl describe pv` to get the details of the PV.

```
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
```

The PV was created by using the `netapp.io/trident` provisioner and is of the type NFS. To support all the new features provided by Astra Trident, this PV should be upgraded to the CSI type.

3. Run the `tridentctl upgrade volume <name-of-trident-volume>` command to upgrade a legacy Astra Trident volume to the CSI spec.

```
./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. Run a `kubectl describe pv` to verify that the volume is a CSI 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>
```

In this manner, you can upgrade volumes of the NFS/iSCSI type that were created by Astra Trident to the CSI type, on a per-volume basis.