How to request storage for Container Platform?

Created by Archana Pillai, last modified on Jun 22, 2022

Table of Contents

Overview

To use NetApp storage on the Container Platform via Trident, users need to create a Persistent Volume Claim (PVC) of the required storage class. Trident tightly integrates with Kubernetes and allows you to request and manage persistent volumes using the native Kubernetes interfaces and constructs. Trident also manages the provisioning of the storage and creation of export rules on the backend NetApp storage. Each Trident storage class is connected to a Trident backend, which can provision NetApp `qtrees` or volumes as required.

In the Container- Platform, volume storage classes follow the storage class naming convention and include a `–volume` suffix. The default storage class for Trident is `trident–csi` (which provisions `qtrees`) or `trident–csi–volume`, which provisions full volumes.

Installing Trident

Trident is installed in the Container Platform using the install role from the rke2_trident collection (`containerplatform.rke2_trident.install`). It allows users to create a persistent volume claim (PVC) using a specific storage class that creates a NetApp volume or

makes it available to mount in a Kubernetes pod.

Prerequisites:

Before you begin, review the following requirements:

- The rke2 cluster must have installed the Kubernetes python modules using the `rke2_k8s_module deps` role.
- The node running the role must have the `Kubernetes.core.k8s` ansible module installed.
- The NetApp storage cluster must have been set up (by the storage team) with the Trident storage virtual machine (SVM) and the Trident credentials (username and password) set up.
- The SVM must have an aggregate assigned with sufficient storage for Trident and data and management logical interfaces (LIFs) configured.
- A default snapshot policy should be applied to all volumes in the SVM so that all Trident volumes get a snapshot created every 15 mins. retained for 6 hours.
- If the SVM LIFs are not directly routable from the Kubernetes nodes then set up a route so that the SVM LIFs can connect to the **HTTPS:** ports on the management LIF and the NFS ports on the data LIFs

Helm Charts required:

Trident-operator - source <https://netapp.github.io/trident-helm-chart>.
Should be available from the local Artifactory Helm repo.

Docker images required:

The following Docker images should be available in the local

Artifactory instance:

- docker.io/netapp/trident

- docker.io/netapp/trident-autosupport
- docker.io/netapp/trident-operator
- k8s.gcr.io/sig-storage/csi-attacher
- k8s.gcr.io/sig-storage/csi-node-driver-registrar
- k8s.gcr.io/sig-storage/csi-provisioner
- k8s.gcr.io/sig-storage/csi-resizer
- k8s.gcr.io/sig-storage/csi-snapshotter

Variables required for the installation:

- *trident_username*: username for NetApp SVM
- *trident_password*: password for NetApp SVM
- *tridentctl_url*: Download the URL for the `tridentctl` utility used for Trident management
- *trident_version*: The version of the Helm chart to be installed
- *trident_management LIF*: The management IP of the Trident SVM.
- *trident_dataLIF_list*: List of data LIFs on the SVM.

Trident storageClass Variables

Trident will allow for multiple backends, allowing the Container Platform to support multiple types of storage on the SVM, i.e., volumes/qtrees or specific aggregates for different Kubernetes storage classes. There are multiple backends per storage class, one per data LIF on the SVM. The default configuration on all new clusters is to have two storage classes, `trident-csi` (qtree-based) and `trident-csi-volume` (volume-based), pointing at each of the data LIFS available on the NetApp.

The following **variable definitions** example will create a storage class called `trident-csi`, which will point at all the data LIFs in the `trident_dataLIF_list` giving the nodes multiple paths to the NetApp storage. The `tridentStorageDriveName` is set to "ontap-nas-economy" which creates a `qtree` per Kubernetes PVC. The second storage class, `trident-csi-volume` will use the same data LIFs to create volume for each new PVC.

Example:

```
trident_storageclass_list:

   - name: trident-csi

     trident_storageDriverName: \"ontap-nas-economy\"

   - name: trident-csi-volume

     trident_storageDriverName: \"ontap-nas\"
```

Role Installation Process:

The role performs the following steps as part of the installation process:

1. Installs NFS Utilities required by Trident to access the NetApp-storage.
2. Installs the `tridentctl` binary on the nodes to enable troubleshooting.
3. Creates the Trident namespace on the Kubernetes cluster.
4. Deploys Trident using the Helm chart using the version specified by the variable from inventory.
5. Loops through the Trident storageclass definitions and creates all of the backends and storage classes.

Post Installation Verification

A user should be able to create a persistent volume claim using a Trident storage class, and an appropriate persistent volume should be created in Kubernetes with a storage object created on the NetApp SVM.

Troubleshooting Installation

You can try the following steps to troubleshoot issues related to Trident installation:

1. If the Trident volumes are not being created, check that the SVM details are correct and manually log in to the NetApp SVM to verify.
2. If the Trident volumes are being created but cannot be mounted within a Kubernetes pod, then the issue is either with the default export policy on the SVM or a problem with connectivity between the node and the Trident data LIFs.
3. Use `tridentctl` to view the current state of Trident volumes and backends.

cp_how_to cp_getting_started