11. Install and configure NetApp Trident storage provisioner

NetApp Solutions

Alan V Cowles, Kevin Hoke
August 11, 2021

Table of Contents

11. Install and configure NetApp Trident storage provisioner

Trident is a storage orchestrator for containers. With Trident, microservices and containerized applications can take advantage of enterprise-class storage services provided by the full NetApp portfolio of storage systems for persistent storage mounts. Depending on an application's requirements, Trident dynamically provisions storage for ONTAP-based products such as NetApp AFF and FAS systems and Element storage systems like NetApp SolidFire and NetApp HCI.

To install Trident on the deployed user cluster and provision a persistent volume, complete the following steps:

The following instructions are screen-capped from a Trident 21.01 install, but the same steps to manually deploy the Trident Operator also apply to the current 21.04 release.

1. Download the installation archive to the admin workstation and extract the contents. The current version of Trident is 21.04, which can be downloaded here.

```
ubuntu@gke-admin-ws-200915-151421:~$ wget
https://github.com/NetApp/trident/releases/download/v21.01.0/trident-
installer-21.01.0.tar.gz
--2021-02-17 12:40:42--
https://github.com/NetApp/trident/releases/download/v21.01.0/trident-
installer-21.01.0.tar.gz
Resolving github.com (github.com)... 140.82.121.4
Connecting to github.com (github.com)|140.82.121.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://github-
releases.githubusercontent.com/77179634/0a63b600-6273-11eb-98df-
3d542851f6ff?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=AKIAIWNJYAX4CSVEH53A%2F20210217%2Fus-east-
1%2Fs3%2Faws4_request&X-Amz-Date=20210217T173945Z&X-Amz-Expires=300&X-
Amz-
Signature=58f26bcac7eeee64673a84d46696490acec357b97a651af42653f973b778ee
88&X-Amz-
SignedHeaders=host&actor_id=0&key_id=0&repo_id=77179634&response-
content-disposition=attachment%3B%20filename%3Dtrident-installer-
21.01.0.tar.gz&response-content-type=application%2Foctet-stream
[following]
--2021-02-17 12:40:43--  https://github-
releases.githubusercontent.com/77179634/0a63b600-6273-11eb-98df-
3d542851f6ff?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-
Credential=AKIAIWNJYAX4CSVEH53A%2F20210217%2Fus-east-
1%2Fs3%2Faws4_request&X-Amz-Date=20210217T173945Z&X-Amz-Expires=300&X-
Amz-
Signature=58f26bcac7eeee64673a84d46696490acec357b97a651af42653f973b778ee
88&X-Amz-
```

```
SignedHeaders=host&actor_id=0&key_id=0&repo_id=77179634&response-
content-disposition=attachment%3B%20filename%3Dtrident-installer-
21.01.0.tar.gz&response-content-type=application%2Foctet-stream
Resolving github-releases.githubusercontent.com (github-
releases.githubusercontent.com)... 185.199.111.154, 185.199.108.154,
185.199.109.154, ...
Connecting to github-releases.githubusercontent.com (github-
releases.githubusercontent.com)|185.199.111.154|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 38527217 (37M) [application/octet-stream]
Saving to: 'trident-installer-21.01.0.tar.gz'

100%[=============================================================
==============================================>] 38,527,217  84.9MB/s
in 0.4s

2021-02-17 12:40:44 (84.9 MB/s) - 'trident-installer-21.01.0.tar.gz'
saved [38527217/38527217]
```

2. Extract the Trident install from the downloaded bundle.

```
ubuntu@gke-admin-ws-200915-151421:~$ tar -xf trident-installer-
21.01.0.tar.gz
ubuntu@gke-admin-ws-200915-151421:~$ cd trident-installer
```

3. First set the location of the user cluster's `kubeconfig` file as an environment variable so that you don't have to reference it, because Trident has no option to pass this file.

```
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ export
KUBECONFIG=~/anthos-cluster01-kubeconfig
```

4. The `trident-installer` directory contains manifests for defining all the required resources. Using the appropriate manifests, create the `TridentOrchestrator` custom resource definition.

```
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ kubectl create -f
deploy/crds/trident.netapp.io_tridentorchestrators_crd_post1.16.yaml
customresourcedefinition.apiextensions.k8s.io/tridentorchestrators.tride
nt.netapp.io created
```

5. If a Trident namespace does not exist, create one in your cluster using the provided manifest.

```
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ kubectl apply -f
deploy/namespace.yaml
namespace/trident created
```

6. Create the resources required for the Trident operator deployment, such as a `ServiceAccount` for the operator, a `ClusterRole` and `ClusterRoleBinding` to the `ServiceAccount`, a dedicated `PodSecurityPolicy`, or the operator itself.

```
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ 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
```

7. You can check the status of the operator after it's deployed with the following commands:

```
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ kubectl get
deployment -n trident
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
trident-operator   1/1     1            1           54s
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ kubectl get pods
-n trident
NAME                                READY   STATUS    RESTARTS   AGE
trident-operator-5c8bbf6754-h957z   1/1     Running   0          68s
```

8. With the operator deployed, we can now use it to install Trident. This requires creating a `TridentOrchestrator`.

```
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ kubectl create -f
deploy/crds/tridentorchestrator_cr.yaml
tridentorchestrator.trident.netapp.io/trident created
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ kubectl describe
torc trident
Name:         trident
Namespace:
Labels:       <none>
Annotations:  <none>
API Version:  trident.netapp.io/v1
Kind:         TridentOrchestrator
Metadata:
  Creation Timestamp:  2021-02-17T18:25:43Z
```

```
  Generation:          1
  Managed Fields:
    API Version:  trident.netapp.io/v1
    Fields Type:  FieldsV1
    fieldsV1:
      f:spec:
        .:
        f:debug:
        f:namespace:
    Manager:      kubectl
    Operation:    Update
    Time:         2021-02-17T18:25:43Z
    API Version:  trident.netapp.io/v1
    Fields Type:  FieldsV1
    fieldsV1:
      f:status:
        .:
        f:currentInstallationParams:
          .:
          f:IPv6:
          f:autosupportHostname:
          f:autosupportImage:
          f:autosupportProxy:
          f:autosupportSerialNumber:
          f:debug:
          f:enableNodePrep:
          f:imagePullSecrets:
          f:imageRegistry:
          f:k8sTimeout:
          f:kubeletDir:
          f:logFormat:
          f:silenceAutosupport:
          f:tridentImage:
        f:message:
        f:namespace:
        f:status:
        f:version:
    Manager:         trident-operator
    Operation:       Update
    Time:            2021-02-17T18:25:43Z
  Resource Version:  14836643
  Self Link:
/apis/trident.netapp.io/v1/tridentorchestrators/trident
  UID:               0e5f2c3b-6ca2-4b85-8453-0382e1426160
Spec:
  Debug:      true
```

```
  Namespace:  trident
Status:
  Current Installation Params:
    IPv6:
    Autosupport Hostname:
    Autosupport Image:
    Autosupport Proxy:
    Autosupport Serial Number:
    Debug:
    Enable Node Prep:
    Image Pull Secrets:         <nil>
    Image Registry:
    k8sTimeout:
    Kubelet Dir:
    Log Format:
    Silence Autosupport:
    Trident Image:
  Message:                      Installing Trident
  Namespace:                    trident
  Status:                       Installing
  Version:
Events:
  Type    Reason      Age   From                           Message
  ----    ------      ----  ----                           -------
  Normal  Installing  23s   trident-operator.netapp.io  Installing
Trident
  Normal  Installed   15s   trident-operator.netapp.io  Trident
installed
```

9. You can verify that Trident is successfully installed by checking the pods that are running in the namespace or by using the tridentctl binary to check the installed version.

```
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ kubectl get pod
-n trident
NAME                                READY   STATUS    RESTARTS   AGE
trident-csi-2cp7x                   2/2     Running   0          4m16s
trident-csi-2xr5h                   2/2     Running   0          4m16s
trident-csi-bnwvh                   2/2     Running   0          4m16s
trident-csi-d6cfc6bb-lxm2p          6/6     Running   0          4m16s
trident-operator-5c8bbf6754-h957z   1/1     Running   0          8m55s

ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ ./tridentctl -n
trident version
+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 21.01.1        | 21.01.1        |
+----------------+----------------+
```

10. The next step in enabling Trident integration with the NetApp HCI solution and Anthos is to create a backend that enables communication with the storage system. NetApp has been validated for several different protocols through the Anthos-ready partner storage validation program. This allows NetApp Trident to provide support in Anthos environments for NFS through our ONTAP platforms and iSCSI from both the ONTAP and Element storage used in NetApp HCI.

> ⓘ A NetApp HCI platform deploys with NetApp Element storage by default. In this guide we configure a backend for this system specifically. In addition to this, a customer can choose to connect to a remote ONTAP storage system or deploy an ONTAP Select software-defined storage system as a virtual appliance in VMware vSphere to provide additional NFS and iSCSI services. The configuration of each of these additional storage backends is beyond the scope of this guide.

11. There are sample backend files available in the downloaded installation archive in the `sample-input` folder. Copy `backend-solidfire.json` to your working directory and edit it to provide information detailing the storage system environment. For Element-based iSCSI connections, copy and edit the `backend-solidfire.json` file.

```
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ cp sample-
input/backend-solidfire.json ./
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ $ vi backend-
solidfire.json
```

    a. Edit the user, password, and MVIP value on the EndPoint line.
    b. Edit the SVIP value.

```
 {
    "version": 1,
    "storageDriverName": "solidfire-san",
    "Endpoint": "https://trident:password@172.21.224.150/json-
rpc/8.0",
    "SVIP": "10.63.172.100:3260",
    "TenantName": "trident",
    "Types": [{"Type": "Bronze", "Qos": {"minIOPS": 1000, "maxIOPS":
2000, "burstIOPS": 4000}},
              {"Type": "Silver", "Qos": {"minIOPS": 4000, "maxIOPS":
6000, "burstIOPS": 8000}},
              {"Type": "Gold", "Qos": {"minIOPS": 6000, "maxIOPS":
8000, "burstIOPS": 10000}}]
}
```

12. With this backend file in place, run the following command to create your first backend.

```
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ ./tridentctl -n
trident create backend -f backend.json
+------------------+---------------
+--------------------------------------+--------+---------+
|    NAME          | STORAGE DRIVER |                 UUID
| STATE  | VOLUMES |
+------------------+---------------
+--------------------------------------+--------+---------+
| solidfire-backend | solidfire-san  | a5f9e159-c8f4-4340-a13a-
c615fef0f433 | online |       0 |
+------------------+---------------
+--------------------------------------+--------+---------+
```

13. With the backend created, you must next create a storage class. Just as with the backend, there is a sample storage class file that can be edited for the environment available in the sample-inputs folder. Copy it to the working directory and make necessary edits to reflect the backend created.

```
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ cp sample-
input/storage-class-csi.yaml.templ ./storage-class-basic.yaml
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ vi storage-class-
basic.yaml
```

14. The only edit that must be made to this file is to define the `backendType` value to the name of the storage driver from the newly created backend. Also note the name-field value that must be referenced in a later step.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: basic-csi
provisioner: csi.trident.netapp.io
parameters:
  backendType: "solidfire-san"
```

15. Run the `kubectl` command to create the storage class.

```
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ kubectl create -f
sample-input/storage-class-basic.yaml
```

16. With the storage class created, you must then create the first persistent volume claim (PVC). There is a sample `pvc-basic.yaml` file that can be used to perform this action located in sample-inputs as well. The only edit that must be made to this file is ensuring that the storageClassName field matches the one just created.

```
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ vi sample-
input/pvc-basic.yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: basic
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: basic-csi
```

17. Create the PVC by issuing the `kubectl` command. Creation can take some time depending on the size of the backing volume being created, so you can watch the process as it completes.

```
ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ kubectl create -f
sample-input/pvc-basic.yaml

ubuntu@gke-admin-ws-200915-151421:~/trident-installer$ kubectl get pvc
--watch
NAME      STATUS    VOLUME                                     CAPACITY
ACCESS MODES  STORAGECLASS   AGE
basic     Pending
basic          1s
basic     Pending   pvc-2azg0d2c-b13e-12e6-8d5f-5342040d22bf   0
basic          5s
basic     Bound     pvc-2azg0d2c-b13e-12e6-8d5f-5342040d22bf   1Gi
RWO           basic          7s
```

Next: Reference videos.