∏ NetApp

9. Download and Install NetApp Trident: NetApp HCI for Red Hat OpenShift on RHV

HCI

NetApp
July 08, 2020

Table of Contents

9. Download and Install NetApp Trident: NetApp HCI for Red Hat OpenShift on RHV

To download and install NetApp Trident, complete the following steps:

1. Make sure that the user that is logged in to the OCP cluster has sufficient privileges for installing Trident.

```
[user@rhel7 openshift-deploy]$ oc auth can-i '*' '*' --all-namespaces
 yes
```

2. Verify that you can download an image from the registry and access the MVIP of the NetApp Element cluster.

```
[user@rhel7 openshift-deploy]$ oc run -i --tty ping --image=busybox
--restart=Never --rm -- ping 10.63.172.140
If you don't see a command prompt, try pressing enter.
64 bytes from 10.63.172.140: seq=1 ttl=63 time=0.312 ms
64 bytes from 10.63.172.140: seq=2 ttl=63 time=0.271 ms
64 bytes from 10.63.172.140: seq=3 ttl=63 time=0.254 ms
64 bytes from 10.63.172.140: seq=4 ttl=63 time=0.309 ms
64 bytes from 10.63.172.140: seq=5 ttl=63 time=0.319 ms
64 bytes from 10.63.172.140: seq=6 ttl=63 time=0.303 ms
^C
--- 10.63.172.140 ping statistics ---
7 packets transmitted, 7 packets received, 0% packet loss
round-trip min/avg/max = 0.254/0.387/0.946 ms
pod "ping" deleted
```

3. Download the Trident installer bundle using the following commands and extract it to a directory.

```
[user@rhel7 ~]$ wget
[user@rhel7 ~]$ tar -xf trident-installer-20.04.0.tar.gz
[user@rhel7 ~]$ cd trident-installer
```

4. The Trident installer contains manifests for defining all the required resources. Using the appropriate manifests, create the TridentProvisioner custom resource definition.

```
[user@rhel7 trident-installer]$ oc create -f
deploy/crds/trident.netapp.io_tridentprovisioners_crd_post1.16.yaml

customresourcedefinition.apiextensions.k8s.io/tridentprovisioners.triden
t.netapp.io created
```

5. Create a Trident namespace, which is required for the Trident operator.

```
[user@rhel7 trident-installer]$ oc create namespace trident
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.

```
[user@rhel7 trident-installer]$ oc kustomize deploy/ >
deploy/bundle.yaml
[user@rhel7 trident-installer]$ oc 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. Verify that the Trident operator is deployed.

```
[user@rhel7 trident-installer]$ oc get deployment -n trident
NAME                READY    UP-TO-DATE    AVAILABLE    AGE
trident-operator    1/1      1             1            56s
[user@rhel7 trident-installer]$ oc get pods -n trident
NAME                              READY    STATUS     RESTARTS    AGE
trident-operator-564d7d66f-qrz7v  1/1      Running    0           71s
```

8. After the Trident operator is installed, install Trident using this operator. In this example, TridentProvisioner custom resource (CR) was created. The Trident installer comes with definitions for creating a TridentProvisioner CR. These can be modified based on the requirements.

```
[user@rhel7 trident-installer]$ oc create -f
deploy/crds/tridentprovisioner_cr.yaml
tridentprovisioner.trident.netapp.io/trident created
```

9. Approve the Trident serving CSR certificates by using `oc get csr -o name | xargs oc adm certificate approve`.

```
[user@rhel7 trident-installer]$ oc get csr -o name | xargs oc adm
certificate approve
certificatesigningrequest.certificates.k8s.io/csr-4b7zh approved
certificatesigningrequest.certificates.k8s.io/csr-4hkwc approved
certificatesigningrequest.certificates.k8s.io/csr-5bgh5 approved
certificatesigningrequest.certificates.k8s.io/csr-5g4d6 approved
certificatesigningrequest.certificates.k8s.io/csr-5j9hz approved
certificatesigningrequest.certificates.k8s.io/csr-5m8qb approved
certificatesigningrequest.certificates.k8s.io/csr-66hv2 approved
certificatesigningrequest.certificates.k8s.io/csr-6rdgg approved
certificatesigningrequest.certificates.k8s.io/csr-6t24f approved
certificatesigningrequest.certificates.k8s.io/csr-76wgv approved
certificatesigningrequest.certificates.k8s.io/csr-78qsq approved
certificatesigningrequest.certificates.k8s.io/csr-7r58n approved
certificatesigningrequest.certificates.k8s.io/csr-8ghmk approved
certificatesigningrequest.certificates.k8s.io/csr-8sn5q approved
```

10. Verify that Trident 20.04 is installed by using the TridentProvisioner CR, and verify that the pods related to Trident are.

```
[user@rhel7 trident-installer]$ oc get tprov -n trident
NAME       AGE
trident    9m49s

[user@rhel7 trident-installer]$ oc describe tprov trident -n trident
Name:         trident
Namespace:    trident
Labels:       <none>
Annotations:  <none>
API Version:  trident.netapp.io/v1
Kind:         TridentProvisioner
Metadata:
  Creation Timestamp:  2020-05-26T18:49:19Z
  Generation:        1
  Resource Version:    640347
  Self Link:
/apis/trident.netapp.io/v1/namespaces/trident/tridentprovisioners/triden
t
  UID:                 52656806-0414-4ed8-b355-fc123fafbf4e
Spec:
  Debug:  true
Status:
  Message:  Trident installed
  Status:   Installed
  Version:  v20.04
```

```
Events:
  Type     Reason      Age                      From
Message
  ----     ------      ----                     ----
-------
  Normal   Installing  9m32s                    trident-operator.netapp.io
Installing Trident
  Normal   Installed   3m47s (x5 over 8m56s)  trident-operator.netapp.io
Trident installed
[user@rhel7 trident-installer]$ oc get pods -n trident
NAME                               READY   STATUS    RESTARTS   AGE
trident-csi-7f769c7875-s6fmt       5/5     Running   0          10m
trident-csi-cp7wg                  2/2     Running   0          10m
trident-csi-hhx94                  2/2     Running   0          10m
trident-csi-l72bt                  2/2     Running   0          10m
trident-csi-xfl9d                  2/2     Running   0          10m
trident-csi-xrhqx                  2/2     Running   0          10m
trident-csi-zb7ws                  2/2     Running   0          10m
trident-operator-564d7d66f-qrz7v   1/1     Running   0          27m

[user@rhel7 trident-installer]$ ./tridentctl -n trident version
+----------------+----------------+
| SERVER VERSION | CLIENT VERSION |
+----------------+----------------+
| 20.04.0        | 20.04.0        |
+----------------+----------------+
```

11. Create a storage backend that will be used by Trident to provision volumes. The storage backend specifies the Element cluster in NetApp HCI. You also can specify sample bronze, silver, and gold types with corresponding QoS specs.

```
[user@rhel7 trident-installer]$ vi backend.json
{
    "version": 1,
    "storageDriverName": "solidfire-san",
    "Endpoint": "https://admin: admin- password@10.63.172.140/json-
rpc/8.0",
    "SVIP": "10.61.185.205:3260",
    "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}}]
}
[user@rhel7 trident-installer]$ ./tridentctl -n trident create backend
-f backend.json
+------------------------+----------------
+-----------------------------------+--------+---------+
|          NAME          | STORAGE DRIVER |                 UUID
| STATE   | VOLUMES |
+------------------------+----------------
+-----------------------------------+--------+---------+
| solidfire_10.61.185.205 | solidfire-san  | 40f48d99-5d2e-4f6c-89ab-
8aee2be71255 | online |       0 |
+------------------------+----------------
+-----------------------------------+--------+---------+
```

Modify the `backend.json` to accommodate the details or requirements of your environment for the following values:

- Endpoint corresponds to the credentials and the MVIP of the NetApp HCI Element cluster.

- SVIP corresponds to the SVIP configured over the VM network in the section titled Create Storage Network VLAN.

- Types corresponds to different QoS bands. New persistent volumes can be created with specific QoS settings by specifying the exact storage pool.

12. Create a StorageClass that specifies Trident as the provisioner and the storage backend as `solidfire-san`.

```
[user@rhel7 trident-installer]$ vi storage-class-basic.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: basic-csi
  annotations:
    storageclass.kubernetes.io/is-default-class: "true"
provisioner: csi.trident.netapp.io
parameters:
  backendType: "solidfire-san"
  provisioningType: "thin"

[user@rhel7 trident-installer]$ oc create -f storage-class-basic.yaml
storageclass.storage.k8s.io/basic created
```

In this example, the StorageClass created is set as a default, however an OpenShift administrator can define multiple storage classes corresponding to different QoS requirements and other factors based upon their applications. Trident selects a storage backend that can satisfy all the criteria specified in the parameters section in the storage class definition. End users can then provision storage as needed, without administrative intervention.