CSI Driver for Dell EMC PowerScale

Product Guide

Version 1.3

September 2020

Contents

Notes, cautions, and warnings

NOTE: A NOTE indicates important information that helps you make better use of your product.

CAUTION: A CAUTION indicates either potential damage to hardware or loss of data and tells you how to avoid the problem.

WARNING: A WARNING indicates a potential for property damage, personal injury, or death.

1

Introduction

This chapter includes the following topics:

Topics:

Product overview

The CSI driver for Dell EMC PowerScale implements an interface between CSI enabled Container Orchestrator (CO) and Dell EMC PowerScale storage system. It allows dynamically provisioning PowerScale volumes and attaching them to workloads.

The CSI driver for Dell EMC PowerScale conforms to CSI specification v1.1. This release of CSI driver for Dell EMC PowerScale supports:

- Kubernetes 1.17, 1.18, and 1.19
- OpenShift 4.3 and 4.4

To learn more about the CSI specification see: https://github.com/container-storage-interface/spec/tree/release-1.1.

Features of the CSI driver for Dell EMC PowerScale

The CSI driver for Dell EMC PowerScale has the following features:

- Persistent Volume (PV) capabilities:
  - Create
  - List
  - Delete
  - Mount
  - Unmount
  - Clone
- Supports mounting volumes as NFS export
- Supports snapshot capabilities:

  NOTE: Snapshot capabilities for OpenShift platform are supported from version 4.4 or later.

  - Create snapshot
  - Create volume from snapshot
  - Delete snapshot
- Supports static and dynamic provisioning of volumes and volume expansion
- Supports the following access modes:
  - SINGLE_NODE_WRITER
  - MULTI_NODE_READER_ONLY
  - MULTI_NODE_MULTI_WRITER
- Conforms to CSI 1.1 specification
- Supports Kubernetes 1.17, 1.18 and 1.19
- Supports OpenShift 4.3 and 4.4
- Supports PowerScale OneFS 8.1, 8.2 and 9.0
- Supports Red Had Enterprise Linux 7.6, 7.7 and 7.8 and CentOS 7.6, 7.7 and 7.8 as host operating system
- Supports Helm3 installer
- Supports installation in OpenShift environments by using Dell CSI Operator.
- Supports CoreOS as host operating system for worker nodes on Openshift platform.

Use SmartQuotas to Limit Storage Consumption

Learn how CSI driver for Dell EMC PowerScale handles capacity limiting using *SmartQuotas* feature.

About this task

To indicate whether to use the *SmartQuotas* feature, in the helm chart the user can specify the boolean value *enableQuota* in *values.yaml* .

For example, when a user creates a *PersistentVolumeClaim* using the following *yaml* file:

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvol0
  namespace: test
spec:
  accessModes:
  - ReadWriteOnce
  volumeMode: Filesystem
  resources:
    requests:
      storage: 8Gi
  storageClassName: isilon
```

The following is how CSI PowerScale handles the *8Gi* capacity requested:

Steps

1. When *enableQuota* is *false*, the CSI PowerScale ignores the *8Gi* request.
2. When *enableQuota* is *true*:
   - If *SmartQuotas* is not activated, the CSI PowerScale ignores the *8Gi* request and continue.
   - If *SmartQuotas* is activated, the CSI PowerScale attempts to create a *8Gi* quota on the volume.
     - If creating quota is successful, the call continues.
     - If creating quota fails, the call reports an error and the transaction is rolled back by removing the volume (directory) that was created. As a result, no PVC is created.

Use Volume Expansion in CSI driver for Dell EMC PowerScale

Volume Expansion is implemented in CSI driver for Dell EMC PowerScale as part of ControllerExpandVolume RPC call.

A resizer container is added to the Controller Pod for the volume expansion feature.

For custom storage classes to support volume expansion feature, they need to set **`allowVolumeExpansion: true`** field.

No additional feature gates need to be enabled to leverage Volume Expansion. Static and dynamic volumes both are supported for volume expansion.

Volumes can be expanded when they are mounted to pod as well.

Expand PersistentVolumeClaim (PVC) capacity

To expand PVC capacity, edit the `storage` field of the volume. For example, the following shows how to edit the `storage` field of the myclaim volume:

`kubectl edit pvc myclaim`

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
name: myclaim
namespace: default
spec:
accessModes:
- ReadWriteMany
resources:
requests:
```

```
storage: 5Gi
storageClassName: isilon
```

NOTE: Reducing current PVC size is not supported

CSI Driver components

This topic describes the components of the CSI driver for Dell EMC PowerScale.

The CSI driver for Dell EMC PowerScale has two components:

- Controller plug-in
- Node plug-in

Controller Plug-in

The Controller plug-in is deployed in a StatefulSet in the cluster with maximum number of replicas set to 1. There is one pod for the Controller plug-in that gets scheduled on any node which is not necessarily the master.

This pod contains the CSI driver for Dell EMC PowerScale container and a few sidecar containers like the *provisioner*, *attacher*, *snapshotter*, and *resizer* that the community provides.

The Controller plug-in primarily deals with provisioning activities like creating volumes, deleting volumes, attaching the volume to a node, and detaching the volume from a node.

Node Plug-in

The Node plug-in is deployed in a DaemonSet in the kubernetes cluster. The Node plug-in deploys the pod containing the driver container on all nodes in the cluster (where the scheduler can schedule the pod).

This pod contains the CSI driver for Dell EMC PowerScale container and sidecar container like registrar that the community provides.

The Node plug-in communicates with the Kubelet to perform tasks like identifying, publishing, and unpublishing a volume to the node where the plug-in is running.

2

Installation

This chapter includes the following topics:

Topics:

Installation overview

This topic gives an overview of the CSI driver for Dell EMC PowerScale installation.

The CSI driver for Dell EMC PowerScale is deployed in the Kubernetes platform using Helm charts. The Helm chart installs CSI driver for Dell EMC PowerScale using a shell script (`dell-csi-helm-installer/csi-install.sh`). This shell script installs the CSI driver container image along with the required Kubernetes sidecar containers.

The controller section of the Helm chart installs the following components in a StatefulSet in the namespace used for installing the driver:

- CSI driver for Dell EMC PowerScale
- Kubernetes Provisioner, which provisions the persistent volumes
- Kubernetes Attacher, which attaches the volumes to the node
- Kubernetes Snapshotter, which provides snapshot support
- Kubernetes Resizer, which provides resize support

The node section of the Helm chart installs the following component in a DaemonSet in the namespace used for installing the driver:

- CSI driver for Dell EMC PowerScale
- Kubernetes Registrar, which handles the driver registration

Prerequisites

This topic lists the prerequisites to install the CSI driver for Dell EMC PowerScale.

Before you install the CSI driver for Dell EMC PowerScale, you must complete the following task:

- Install supported kubernetes version (v1.17, v1.18, or v1.19)
- Configure Docker service on page 8
- Install Helm 3 on page 8
- Install Volume Snapshot Components on page 8

After completing these prerequisites, you are able to Install CSI driver for Dell EMC PowerScale on page 9.

Configure Docker service

This topic gives the procedure to configure docker service. Configure the mount propagation in Docker on all Kubernetes nodes before installing the CSI driver for Dell EMC PowerScale. The recommended docker version is 19.03.

Steps

1. Edit the service section of */etc/systemd/system/multi-user.target.wants/docker.service* file to add the following lines:

   ```
   [Service]...
   MountFlags=shared
   ```

2. Restart the docker service with `systemctl daemon-reload` and `systemctl restart docker` on all the nodes.

Install Helm 3

Install Helm 3 on all master nodes before you install the CSI driver for Dell EMC PowerScale.

Steps

Run the `curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 | bash` command to install Helm 3.

Install Volume Snapshot Components

Use the procedure in this topic to install volume snapshot components.

Install Snapshot Beta CRDs

To install snapshot CRDs specify *--snapshot-crd* flag to driver installation script `dell-csi-helm-installer/csi-install.sh` during driver installation.

Install Common Snapshot Controller

See, Install Common Snapshot Controller, if not already installed for the cluster.

Run the following commands:

- `kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/master/deploy/kubernetes/snapshot-controller/rbac-snapshot-controller.yaml`
- `kubectl apply -f https://raw.githubusercontent.com/kubernetes-csi/external-snapshotter/master/deploy/kubernetes/snapshot-controller/setup-snapshot-controller.yaml`

Certificate validation for OneFS REST API calls

This topic provides details about setting up the certificate validation for the CSI driver for Dell EMC PowerScale.

Prerequisites

As part of the CSI driver installation, the CSI driver requires a secret with the name *isilon-certs* present in the namespace where the driver is installed. This secret contains the X509 certificates of the CA which signed the OneFS SSL certificate in PEM format.

About this task

The CSI driver exposes an install parameter `isiInsecure` which determines if the driver performs client-side verification of the OneFS certificates. The `isiInsecure` parameter is set to true by default and the driver does not verify the OneFS certificates.

Steps

- If the `isiInsecure` is set to false, then the secret isilon-certs must contain the CA certificate for OneFS.
- If this secret is an empty secret, then the validation of the certificate fails, and the driver fails to start.
- If the `isiInsecure` parameter is set to false and a previous installation attempt created the empty secret, then this secret must be deleted and re-created using the CA certs.
- If the OneFS certificate is self-signed, then perform the following steps:
  1. To fetch the certificate:

     ```
     openssl s_client -showcerts -connect <OneFS IP> </dev/null 2>/dev/null | openssl x509
     -outform PEM > ca_cert.pem
     ```
  2. To create the secret:

     ```
     kubectl create secret generic isilon-certs --from-file=ca_cert.pem -n isilon
     ```

Install CSI driver for Dell EMC PowerScale

CSI driver for Dell EMC PowerScale can be installed in one of two ways. Choose the method appropriate for your environement.

Install the CSI driver for Dell EMC PowerScale with one of the following methods:

- Install the CSI driver for Dell EMC PowerScale using Helm on page 9
- Install the CSI driver for Dell EMC PowerScale using the Operator on page 12

Install the CSI driver for Dell EMC PowerScale using Helm

Install the CSI driver for Dell EMC PowerScale using this procedure.

Prerequisites

Ensure that you meet the following prerequisites before you install the CSI driver for Dell EMC PowerScale:

- You have the downloaded files ready for this procedure.
- You have the Helm chart from the source repository at GitHub ready for this procedure.
- In the `dell-csi-helm-installer` directory, it contains two shell scripts,`csi-install.sh` and `csi-uninstall.sh`.

  The scripts perform certain pre-installation and post-installation operations, which cannot be performed in the helm chart.
- You have the Kubernetes secret with your OneFS username and password.

  You can use the *secret.yaml* file to create the secret with the following values to match the default installation parameters:

  - Name: **`isilon-creds`**
  - Namespace: **`isilon`**

  NOTE: The username must be from the authentication providers in the System zone of Isilon. The user must have enough privileges to perform the actions. The suggested privileges are as follows:

  - ISI_PRIV_LOGIN_PAPI
  - ISI_PRIV_NFS
  - ISI_PRIV_QUOTA
  - ISI_PRIV_SNAPSHOT
  - ISI_PRIV_IFS_RESTORE
  - ISI_PRIV_NS_IFS_ACCESS

  For more information about creating a Kubernetes secret, see https://kubernetes.io/docs/concepts/configuration/secret/.

- The mount propagations are configured in Docker.

Steps

1. Collect information from the PowerScale system, such as IP address, username and password.

   Make a note of the value for these parameters as they must be entered in the `secret.yaml` and `myvalues.yaml` file.

2. Copy the `helm/csi-isilon/values.yaml` into a new location for example, `my-isilon-settings.yaml` to customize settings for installation.
3. Edit `my-isilon-settings.yaml` to set the following parameters for installation:

   The following table lists the primary configurable parameters of the PowerScale driver helm chart and their default values.

Table 1. Primary parameters

Parameter	Description	Required	Default
isiIP	This parameter is the HTTPs endpoint of the PowerScale OneFS API server.	true	
isiPort	This parameter is the HTTPs port number of the PowerScale OneFS API server.	false	8080
isiInsecure	This parameter specifies whether the PowerScale OneFS API server's certificate chain and host name should be verified.	false	true
isiAccessZone	This parameter is the name of the access zone a volume can be created in. This parameter is used if accessZone is not specified for storageClass.	false	System
volumeNamePrefix	This parameter defines a string prepended to each volume created by the CSI driver.	false	k8s
enableDebug	This parameter indicates whether debug level logs should be logged.	false	true
verbose	This parameter Indicates what content of the OneFS REST API message should be logged in debug level logs.	false	1
enableQuota	This parameter indicates whether the provisioner should attempt to set (later unset) quota on a newly provisioned volume. This requires SmartQuotas to be enabled.	false	true
noProbeOnStart	This parameter specifies whether the controller/node should probe during initialization.	false	false
isiPath	The default base path for the volumes to be created, this will be used if a storage class does not have the IsiPath parameter specified.	false	/ifs/data/csi
autoProbe	This parameter enables auto probe.	false	true
nfsV3	This parameter specifies whether to set the version to v3 when mounting an NFS export. If the value is *false*, the default version that is supported is used.	false	false

Table 2. Storage Class parameters

Parameter	Description	Required	Default
name	This parameter is the name of the storage class to be defined.	false	isilon
isDefault	This parameter specifies whether this storage class should be default.	false	true

Table 2. Storage Class parameters (continued)

Parameter	Description	Required	Default
reclaimPolicy	This parameter indicates what happens when a volume is removed from the Kubernetes API. Valid values are *Retain* and *Delete*.	false	Delete
accessZone	This parameter is the name of the access zone a volume can be created in. Ensure that this accessZone exists on Isilon.	false	System
AzServiceIP	This parameter is the Access Zone service IP. If the value is different from isiIP, specify in values and refer in storageClass.	false	
rootClientEnabled	When a PVC is being created, it takes the storage class' value of *storageclass.rootClientEnabled*	false	false

NOTE: User must provide all boolean values with double quotes. This applicable only for `my-isilon-settings.yaml`. For example: *"true"/"false"*.

4. Run `kubectl create namespace isilon` to create the *isilon* namespace.

   Specify the same namespace name while installing the driver.

   NOTE: CSI driver for Dell EMC PowerScale also supports installation of driver in custom namespace.

5. Create a secret file for the OneFS credentials by editing the `secret.yaml` in the helm directory. Replace the values for the username and password parameters. Use the following command to convert username/password to base64 encoded string:
   - `echo -n 'admin' | base64`
   - `echo -n 'password' | base64`
6. Run `kubectl create -f secret.yaml` to create the secret.
7. Install OneFS CA certificates by following the instructions from next section, if you want to validate OneFS API server's certificates. If not, create an empty secret using the following command:

   `kubectl create -f emptysecret.yaml`
8. Install the driver using the following script from the `dell-csi-helm-installer` directory:

   `./csi-install.sh --namespace isilon --values ./my-isilon-settings.yaml --snapshot-crd`.

   This script also runs the verify.sh script that is present in the `dell-csi-helm-installer` directory.

   NOTE: The installation might fail if you are installing the CSI driver for the first time. It can be due to network speed, since the sidecar containers might still be creating or downloading the resources. Uninstall the driver and install it again, if it occurs.

9. Optional: Run `./csi-uninstall.sh --namespace isilon` to uninstall CSI driver for Dell EMC PowerScale.

Results

The CSI driver for Dell EMC PowerScale is installed. You can check for the pods that are deployed by running the following command:

- `kubectl get pods -n isilon`

  You will see the following:
  - isilon-controller-0 with 5/5 containers ready, and status displayed as Running.
  - Node pods with 2/2 containers and the status displayed as Running.

The script creates a default storage class "isilon". Additional storage classes can be created for different combinations of file system types and PowerScale storage pools. The script also creates volumesnapshotclass "isilon-snapclass".

Install the CSI driver for Dell EMC PowerScale using the Operator

CSI driver for Dell EMC PowerScale can also be installed by using Dell EMC CSI Operator.

The Dell EMC Storage CSI Operator is a Kubernetes Operator that can be used to install and manage the CSI Drivers that Dell EMC provides for various storage platforms. This operator is available as a community operator for upstream Kubernetes and can be deployed using `OperatorHub.io`. It is also available as a community operator for OpenShift clusters and can be deployed on the OpenShift Container Platform. Both these methods of installation use Operator Lifecycle Manager (OLM).

Instructions for either deploying the Operator directly and deploying the CSI driver for Dell EMC PowerScale using the Operator can be found on GitHub.

These instructions include sample manifests that can be edited for the installation of the driver.

NOTE: The deployment of the driver using the Operator does not use any Helm charts. The installation and configuration parameters are slightly different from the ones that are specified in the Helm installer.

Kubernetes Operators make it easy to deploy and manage entire lifecycle of complex Kubernetes applications. Operators use Custom Resource Definitions (CRD), which represents the application and uses custom controllers to manage them.

Listing CSI-PowerScale drivers

About this task

User can query for csi-isilon driver using the following command:

`kubectl get csiisilon --all-namespaces`

Create a new CSI-Isilon driver

Steps

1. Create the Isilon namespace:

   `kubectl create namespace isilon`
2. Create `isilon-creds`
   a. Create a file called isilon-creds.yaml with the following content:

   ```
   yaml
       apiVersion: v1
       kind: Secret
       metadata:
         name: isilon-creds
         namespace: isilon
       type: Opaque
       data:
         # set username to the base64 encoded username
         username: <base64 username>
         # set password to the base64 encoded password
         password: <base64 password>
   ```

   b. Replace the values for the username and password parameters. These values can be optioned using base64 encoding as described in the following example:

   ```
   echo -n "myusername" | base64
   echo -n "mypassword" | base64
   ```

   c. Run `kubectl create -f isilon-creds.yaml` command to create the secret.
3. Create a CR (Custom Resource) for PowerScale using the sample files provided at `test/sample_files/CR/`.
4. Run the following command to create Isilon custom resource:

   `kubectl create -f` *<input_sample_file.yaml>*

   NOTE: The above command will deploy the csi-powerscale driver.

5. User can configure the following parameters in a CR:

Parameter	**Description**	**Required**	**Default**
Common parameters for node and controller			
CSI_ENDPOINT	Specifies the CSI endpoint	No	`/var/run/csi/csi.sock`
X_CSI_DEBUG	To enable debug mode	No	false
X_CSI_ISI_ENDPOINT	HTTPs endpoint of the Isilon OneFS API server	Yes	
X_CSI_ISI_INSECURE	Specifies whether SSL security needs to be enabled for communication between Isilon and CSI Driver	No	true
X_CSI_ISI_PATH	Base path for the volumes to be created	Yes	
X_CSI_ISI_AUTOPROBE	To enable auto probing for driver	No	true
X_CSI_ISILON_NO_PROBE_ON_START	Indicates whether the controller/node should probe during initialization	Yes	
Controller parameters			
X_CSI_MODE	Driver starting mode	No	controller
X_CSI_ISI_ACCESS_ZONE	Name of the access zone a volume can be created in	No	System
X_CSI_ISI_QUOTA_ENABLED	To enable SmartQuotas	No	
Node parameters			
X_CSI_ISILON_NFS_V3	Set the version to v3 when mounting an NFS export. If the value is "false", then the default version supported will be used	Yes	
X_CSI_MODE	Driver starting mode	No	node

Upgrade CSI driver for Dell EMC PowerScale

Use the procedure in this topic to upgrade the CSI driver for Dell EMC PowerScale version 1.2.0 to version 1.3.0.

Prerequisites

NOTE: If CSI driver for Dell EMC PowerScale version 1.2 was installed using helm2, you must migrate to helm3 before upgrading the driver. See, Helm3 Migration for instructions.

Ensure that you meet the following pre-requisites before you upgrade the CSI driver for Dell EMC PowerScale.

- You must remove all volume snapshots, volume snapshot contents and volume snapshot class objects.
- Upgrade the Kubernetes version to Kubernetes 1.17.x version.
- Uninstall alpha snapshot CRDs if any.
- Uninstall CSI driver for Dell EMC PowerScale version 1.2.0.
- Verify that all pre-requisites to install CSI driver for Dell EMC PowerScale version 1.3.0 are met.

Steps

1. Clone the repository https://github.com/dell/csi-powerscale.
2. Copy the `helm/csi-isilon/values.yaml` into a new location, for example, *my-isilon-settings.yaml* to customize settings for installation.

3. Edit *my-isilon-settings.yaml* as per the requirements.
4. Change to directory `dell-csi-helm-installer` to install CSI driver for Dell EMC PowerScale.

   Use the following command:

   ```
   cd dell-csi-helm-installer
   ```
5. Install the CSI driver for Dell EMC PowerScale version 1.3.0 using following command:

   ```
   ./csi-install.sh --namespace isilon --values ./my-isilon-settings.yaml --snapshot-crd
   ```

Test deploying a simple pod with PowerScale storage

Test the deployment workflow of a simple pod on PowerScale storage.

Steps

1. Create a volume.
   a. Create a file `pvc.yaml` using the sample yaml file located at `test/sample_files`
   b. Run the following command to create the test volume:

      ```
      kubectl create -f $PWD/pvc.yaml
      ```

   After executing the above command PVC will be created in the default namespace, and the user can see the pvc by executing `kubectl get pvc`.
2. Verify system for the new volume.
3. Attach the volume to a Host.

   To attach the test volume to a host, create a new application(Pod) and use the PVC created above in the Pod. This scenario is explained using the Nginx application.

   a. Create `nginx.yaml` using the sample yaml file located at `test/sample_files`.
   b. Run the following command to mount the volume to kubernetes node:

      ```
      kubectl create -f $PWD/nginx.yaml
      ```

   After running the above command, a new nginx pod will be successfully created and started in the default namespace.
4. Verify PowerScale system for host to be part of clients/rootclients field of export created for volume and used by nginx application.
5. Create a snapshot of the volume in the container using VolumeSnapshot objects defined in `snap.yaml`. The sample file for snapshot creation is located at `test/sample_files/`

   ```
   kubectl create -f $PWD/snap.yaml
   ```

   The spec.source section contains the volume that will be snapped in the default namespace. For example, if the volume to be snapped is `testvolclaim1`, then the created snapshot is named `testvolclaim1-snap1`.
6. Verify the Isilon system for new created snapshot.

   NOTE:

   - User can see the snapshots using `kubectl get volumesnapshot`
   - VolumeSnapshot class has a reference to a snapshotClassName:isilon-snapclass. The CSI Driver for PowerScale installation creates this class as its default snapshot class.
   - You can see its definition using `kubectl get volumesnapshotclasses isilon-snapclass -o yaml`.
7. Create a volume from the snapshot.

   The following will create a new volume from a given snapshot which is specified in spec dataSource field.

   The sample file for volume creation from a snapshot, `volume_from_snap.yaml`, is located at `test/sample_files/`.

   ```
   kubectl create -f $PWD/volume_from_snap.yaml
   ```
8. Verify the PowerScale system for new created volume from snapshot.
9. Delete test snapshot.

   ```
   kubectl get volumesnapshot
   ```

   ```
   kubectl delete volumesnapshot testvolclaim1-snap1
   ```

10. Delete the nginx application to unattach the volume from the host.

    kubectl delete -f nginx.yaml

11. Delete the test volume.

    kubectl get pvc

    kubectl delete pvc testvolclaim1

    kubectl get pvc

CSI Driver Usage

Once you install the plug-in, it creates a default storage class using parameters from my-isilon-settings.yaml. You can also create your own storage class by specifying parameters which decide how storage gets provisioned on the Dell EMC array. The `StorageClass` parameters are as follows:

Parameter	Description
`name`	The name of the storage class to be defined.
`reclaimPolicy`	This parameter indicates what happens when a volume is removed from the Kubernetes API. Valid values are *Retain* and *Delete*.
`AccessZone`	This parameter is the name of the access zone a volume can be created in. Ensure that this `AccessZone` exists on Isilon.
`IsiPath`	This parameter is the base path for the volumes to be created. Ensure that this path exists on Isilon.
`AzServiceIP`	Access Zone service IP.
`RootClientEnabled`	This parameter determines whether to add the Kubernetes node FQDN to the client field when false or set node FQDN to clients and root client fields when true of the NFS export.

`VolumeSnapshotClass` parameter:

- `IsiPath` – The base path for the source volume the snapshot is created from. Ensure that this path exists on Isilon. It should be consistent with `isiPath` in the `storageclass` that the source volume uses.

Controller Plug-in query commands

This topic lists the commands to view the details of StatefulSet and check logs for the Controller Plug-in.

Steps

1. Run the following command to query the details of the StatefulSet:

```
kubectl get statefulset -n isilon
kubectl describe statefulset isilon-controller -n isilon
```

2. Run the following command to check the logs for the Controller plug-in:

```
kubectl logs isilon-controller-0 driver -n isilon
```

   Similarly, logs for provisioner and attacher sidecars can be obtained by specifying the container names.

   Set the log levels in the verbose field of `my-isilon-settings.yaml` to refine the granularity of the logs on RESTful calls to OneFS. Ensure the parameter `enableDebug` in `my-isilon-settings.yaml` is set to true to see these RESTful calls.

Log Level	Definition
0	Log full content of the HTTP request and response

Log Level	Definition
1	Log without the HTTP response body
2	Log only first line of the HTTP request and response

Node Plug-in query commands

This topic lists the commands to view the details of DaemonSet.

Steps

1. Run the following command to get the details of the DaemonSet:

```
kubectl get daemonset -n isilon
kubectl describe daemonset isilon-node -n isilon
```

2. Use the following sample command to check the logs for the Node plug-in:

```
kubectl logs -n isilon <node plugin pod name> driver
```

Test the CSI driver for Dell EMC PowerScale

This chapter includes the following topics:

Topics:

Test the CSI driver for Dell EMC PowerScale

This topic provides information about testing the CSI driver for Dell EMC PowerScale.

About this task

The *csi-powerscale* repository includes examples of how you can use the CSI driver for Dell EMC PowerScale. These examples automate the creation of pods using the default storage classes that were created during installation. The shell scripts are used to automate the installation and uninstallation of helm charts for the creation of pods with different number of volumes. To test the installation of the CSI driver, perform the following procedure:

Steps

1. Create a namespace with the name *test*.
2. Run the `cd csi-isilon/test/helm` command to go to the `csi-isilon/test/helm` directory, which contains the *starttest.sh* and the *2vols* directories.
3. Run the `starttest.sh` script and provide it a test name. The following is a sample script that can be used to run the 2vols test:

   ```
   ./starttest.sh -t 2vols -n test
   ```

   This script installs a helm chart that creates a pod with a container, creates two PVCs, and mounts them into the created container. You can now log in to the newly created container and check the mounts.
4. Run the `./stoptest.sh -t 2vols` script to stop the test.
   This script deletes the pod and the PVCs created during the test and uninstalls the helm chart.

Test creating snapshots

Test the workflow for snapshot creation.

Steps

1. Start the *2vols* container and leave it running.
2. Run the `snaptest.sh` shell script.

   This will create a snapshot of each of the volumes in the container using *VolumeSnapshot* objects defined in *snap1.yaml* and *snap2.yaml*. The following are the contents of *snap1.yaml*:

   ```
   apiVersion: snapshot.storage.k8s.io/v1beta1
   kind: VolumeSnapshot
   metadata:
     name: pvol0-snap1
     namespace: test
   spec:
     volumeSnapshotClassName: isilon-snapclass
   ```

```
  source:
    persistentVolumeClaimName: pvol0
```

The following are the contents of *snap2.yaml*:

```
apiVersion: snapshot.storage.k8s.io/v1beta1
kind: VolumeSnapshot
metadata:
  name: pvol0-snap2
  namespace: test
spec:
  volumeSnapshotClassName: isilon-snapclass
  source:
    persistentVolumeClaimName: pvol0
```

Results

The *snaptest.sh* script will create a snapshot using the definitions in the *snap1.yaml* file. The *spec.source* section contains the volume that will be snapped. For example, if the volume to be snapped is *pvol0*, then the created snapshot is named *pvol0-snap1*.

NOTE: The *snaptest.sh* shell script creates the snapshots, describes them, and then deletes them. You can see your snapshots using *kubectl get volumesnapshot -n test*.

Notice that this *VolumeSnapshot* class has a reference to a *snapshotClassName: isilon-snapclass*. The CSI driver for Dell EMC PowerScale installation creates this class as its default snapshot class. You can see its definition in the installation directory file *volumesnapshotclass.yaml*.

Test restoring from a snapshot

Test the restore operation workflow to restore from a snapshot.

Prerequisites

Ensure that you have stopped any previous test instance before performing this procedure.

About this task

To test the restore operation from a snapshot:

Steps

Run the `snaprestoretest.sh` shell script.

This script deploys the *2vols* example, creates a snap of pvol0, and then updates the deployed helm chart from the updated directory *2vols+restore*. This then adds an additional volume that is created from the snapshot.

Results

An outline of this workflow is described below:

1. The snapshot is taken using *snap1.yaml*.
2. *Helm* is called to upgrade the deployment with a new definition, which is found in the *2vols+restore* directory. The `csi-isilon/test/helm/2vols+restore/templates` directory contains the newly created *createFromSnap.yaml* file. The script then creates a *PersistentVolumeClaim*, which is a volume that is dynamically created from the snapshot. Then the helm deployment is upgraded to contain the newly created third volume. In other words, when the *snaprestoretest.sh* creates a new volume with data from the snapshot, the restore operation is tested. The contents of the *createFromSnap.yaml* are described below:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: restorepvc
  namespace: test
spec:
  storageClassName: isilon
  dataSource:
    name: pvol0-snap1
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 8Gi
```

NOTE: The *spec.dataSource* clause, specifies a source *VolumeSnapshot* named *pvol0-snap1* which matches the snapshot's name in *snap1.yaml*.