eSDK Huawei Storage Kubernetes CSI Plugins V4.2.0

User Guide

Issue 02

Date 2024-01-15

HUAWEI TECHNOLOGIES CO., LTD.

Trademarks and Permissions


and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Technologies Co., Ltd.

Address: Huawei Industrial Base
Bantian, Longgang
Shenzhen 518129
People's Republic of China

Website: https://e.huawei.com

Security Declaration

Product Lifecycle

Huawei's regulations on product lifecycle are subject to the *Product End of Life Policy.* For details about this policy, visit the following web page:
https://support.huawei.com/ecolumnsweb/en/warranty-policy

Vulnerability

Huawei's regulations on product vulnerability management are subject to the *Vul. Response Process.* For details about this process, visit the following web page:
https://www.huawei.com/en/psirt/vul-response-process
For vulnerability information, enterprise customers can visit the following web page:
https://securitybulletin.huawei.com/enterprise/en/security-advisory

Preconfigured Digital Certificate

The digital certificates preconfigured on Huawei devices are subject to the *Rights and Responsibilities of Preconfigured Digital Certificates on Huawei Devices.* For details about this document, visit the following web page:
https://support.huawei.com/enterprise/en/bulletins-service/ENEWS2000015789

Huawei Enterprise End User License Agreement

This agreement is the end user license agreement between you (an individual, company, or any other entity) and Huawei for the use of the Huawei Software. Your use of the Huawei Software will be deemed as your acceptance of the terms mentioned in this agreement. For details about this agreement, visit the following web page:
https://e.huawei.com/en/about/eula

Lifecycle of Product Documentation

Huawei after-sales user documentation is subject to the *Product Documentation Lifecycle Policy.* For details about this policy, visit the following web page:
https://support.huawei.com/enterprise/en/bulletins-website/ENEWS2000017761

About This Document

Intended Audience

This document is intended for:

- Technical support engineers
- O&M engineers
- Engineers with basic knowledge of storage and Kubernetes

Symbol Conventions

The symbols that may be found in this document are defined as follows.

Symbol	Description
DANGER	Indicates a hazard with a high level of risk which, if not avoided, will result in death or serious injury.
WARNING	Indicates a hazard with a medium level of risk which, if not avoided, could result in death or serious injury.
CAUTION	Indicates a hazard with a low level of risk which, if not avoided, could result in minor or moderate injury.
NOTICE	Indicates a potentially hazardous situation which, if not avoided, could result in equipment damage, data loss, performance deterioration, or unanticipated results. NOTICE is used to address practices not related to personal injury.
NOTE	Supplements the important information in the main text. NOTE is used to address information not related to personal injury, equipment damage, and environment deterioration.

Change History

Issue	Date	Description
02	2024-01-15	This issue is the second official release.
01	2023-09-30	This issue is the first official release.

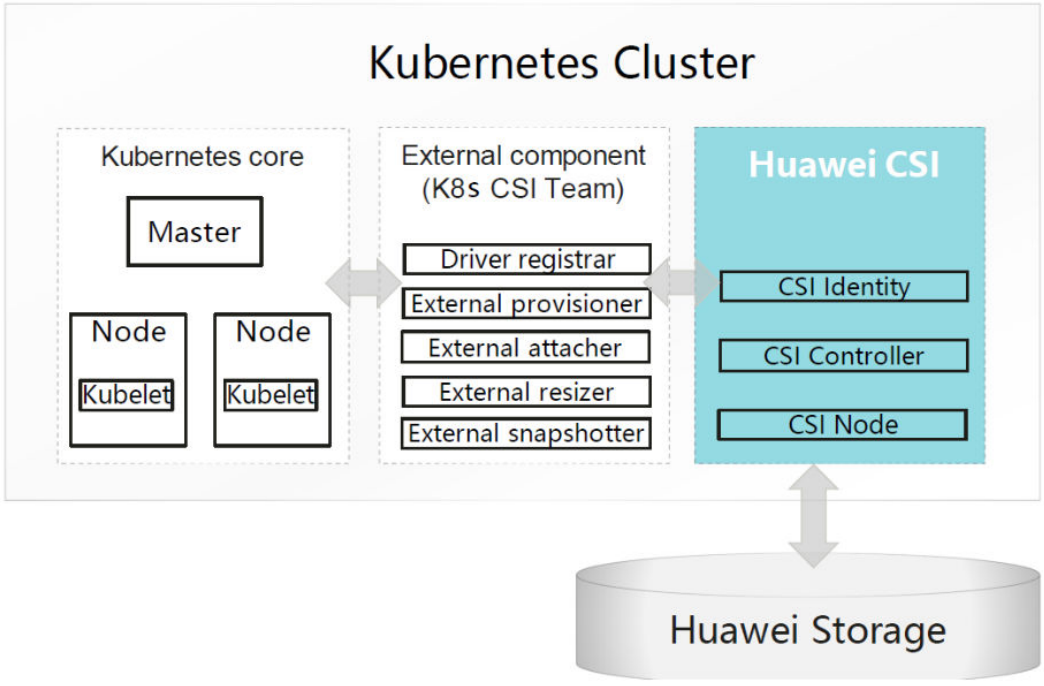
Contents

1 Overview

Container Storage Interface (CSI) is an industry standard used to expose block and file storage systems to container workloads on container orchestration systems (COs) such as Kubernetes. Huawei CSI plug-in is used to communicate with Huawei enterprise storage and distributed storage products and provide storage services for Kubernetes container workloads. It is a mandatory plug-in used by Huawei enterprise storage and distributed storage in the Kubernetes environment.

Kubernetes uses a series of officially maintained sidecar components to register and listen to Kubernetes object resources and call CSI Driver when necessary. Huawei CSI Driver implements the call initiated by sidecar on Huawei storage, for example, creating a **Persistent Volume (PV)** is to create a LUN or file system on Huawei storage. The following figure shows the overall structure of Kubernetes, Huawei CSI, and Huawei storage.

Figure 1-1 CSI overall architecture

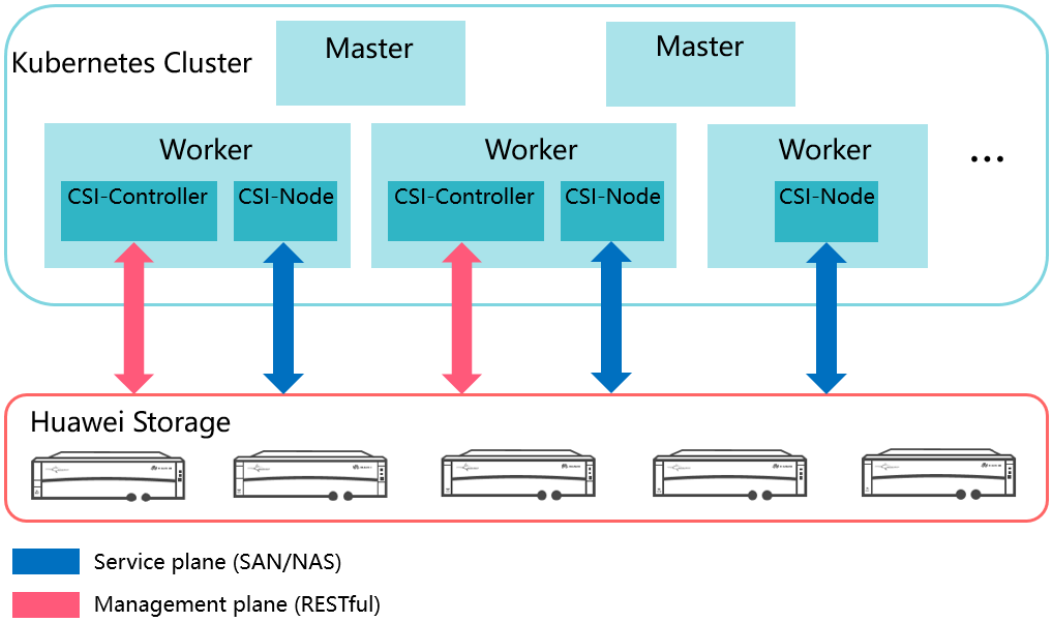


Huawei CSI consists of two components: CSI Controller and CSI Node.

- CSI Controller: one or more Pods running in Deployment mode. It is used to interact with Huawei storage using RESTful. Therefore, the node running the CSI Controller component must be connected to the management plane network of the storage.
- CSI Node: a Pod that runs on Kubernetes worker nodes in DaemonSet mode. It is used to mount and unmount a LUN/file system provided by Huawei storage on worker nodes. Therefore, the node running the CSI Node component must be connected to the service plane network of the storage.

The following figure shows the deployment model of Huawei CSI.

Figure 1-2 CSI deployment model



This document describes how to install, deploy, and use the Huawei CSI V4.2.0 plug-in.

2 Compatibility and Features

This chapter describes the container management platforms, operating systems (OSs), and multipathing software supported by Huawei CSI plug-in, as well as the features and functions provided by the CSI plug-in when working with Huawei storage.

2.1 Kubernetes and OS Compatibility

Huawei CSI plug-in supports the following container management platforms.

Table 2-1 Supported container management platforms

Container Management Platform	Version
Kubernetes	1.16, 1.18 to 1.27
Red Hat OpenShift Container Platform	4.6 EUS, 4.7, 4.8, 4.9, 4.10, 4.11, 4.12, 4.13
Tanzu Kubernetes	TKGI 1.14.1, TKGI 1.15, TKGI 1.16, TKGI 1.17
CCE Agile	22.3.2
CCE	22.9.5

NOTICE

- For details about how to connect Huawei CSI to Red Hat OpenShift, see ***eSDK Enterprise Storage Plugins User Guide (Kubernetes CSI for Red Hat OpenShift)***.
- The connection between Huawei CSI and Tanzu Kubernetes supports only the NAS scenario. For related FAQ, see **11.15 Common Problems and Solutions for Interconnecting with the Tanzu Kubernetes Cluster**.

The following table lists the OSs and multipathing software supported by the Huawei CSI plug-in.

Table 2-2 Supported host OSs and multipathing software versions

OS Name	OS Version	Native DM-Multipath Version	Huawei UltraPath version
CentOS x86_64	7.6, 7.7, 7.9	Delivered with the OS, supporting FC/iSCSI	UltraPath 31.1.0, supporting FC/iSCSI
CentOS x86_64	8.2, 8.4	Delivered with the OS, supporting FC/iSCSI	UltraPath 31.1.0, supporting FC/iSCSI UltraPath-NVMe 31.1.RC8, supporting NVMe over RoCE/NVMe over FC
CentOS ARM	7.6	Delivered with the OS, supporting FC/iSCSI	Not supported
Rocky Linux x86_64	8.6	Delivered with the OS, supporting FC/iSCSI	Not supported
SUSE 15 x86_64	SP2, SP3	Delivered with the OS, supporting FC/iSCSI	UltraPath 31.1.0, supporting FC/iSCSI UltraPath-NVMe 31.1.RC8, supporting NVMe over RoCE/NVMe over FC
Red Hat CoreOS x86_64	4.6, 4.7, 4.8, 4.9, 4.10, 4.11, 4.12, 4.13	Delivered with the OS, supporting FC/iSCSI	Not supported
Ubuntu x86_64	18.04, 20.04, 22.04	Delivered with the OS, supporting FC/iSCSI	Not supported
Kylin x86_64	V10 SP1, V10 SP2, 7.6	Delivered with the OS, supporting FC/iSCSI	Not supported

OS Name	OS Version	Native DM-Multipath Version	Huawei UltraPath version
Kylin ARM	V10 SP1, V10 SP2	Delivered with the OS, supporting FC/iSCSI	Not supported
Debian x86_64	9, 11	Delivered with the OS, supporting FC/iSCSI	Not supported
EulerOS x86_64	V2R9, V2R10, V2R11	Delivered with the OS, supporting FC/iSCSI	Not supported
EulerOS ARM	V2R10	Delivered with the OS, supporting FC/iSCSI	Not supported

NOTE

For DM-Multipath 0.7, some virtual devices may not be displayed in the command output after the **multipathd show maps** command is executed. Therefore, you are advised to use version 0.8 or later.

You can query the DM-Multipath version in either of the following ways:

- If the rpm package is used, run the **rpm -qa | grep device-mapper** command.
- If the deb package is used, run the **dpkg -l | grep multipath** command.

2.2 Kubernetes Feature Matrix

This section describes the features of different Kubernetes versions supported by Huawei CSI.

Table 2-3 Kubernetes versions and supported features

Feature	V1.16	V1.18	V1.19	V1.20	V1.21+
Static Provisioning	√	√	√	√	√
Dynamic Provisioning	√	√	√	√	√
Manage Provisioning[1]	√	√	√	√	√
Expand Persistent Volume	√	√	√	√	√
Create VolumeSnapshot	x	√	√	√	√

Feature	V1.16	V1.18	V1.19	V1.20	V1.21+
Restore VolumeSnapshot	x	√	√	√	√
Delete VolumeSnapshot	x	√	√	√	√
Clone Persistent Volume	x	√	√	√	√
Raw Block Volume	√	√	√	√	√
Topology	√	√	√	√	√
Generic Ephemeral Volumes	×	×	×	×	√

- Note 1: Manage Provisioning is a volume management feature customized by Huawei CSI. This feature allows existing storage resources to be managed by Kubernetes. A storage resource cannot be managed for multiple times.

2.3 Compatibility with Huawei Enterprise Storage

Huawei CSI plug-in is compatible with Huawei OceanStor series all-flash storage and hybrid flash storage. The following table lists the supported storage versions.

Table 2-4 Supported Huawei enterprise storage

Storage Product	Version
OceanStor V3	V300R006
OceanStor V5	V500R007, V500R007 Kunpeng
OceanStor Dorado V3	V300R002
OceanStor V6	6.1.3, 6.1.5, 6.1.6
OceanStor Dorado V6	6.1.0, 6.1.2, 6.1.3, 6.1.5, 6.1.6

Huawei CSI plug-in supports the following features for Huawei enterprise storage.

Table 2-5 Features supported by Huawei enterprise storage and constraints

Feature	OceanStor V3	OceanStor V5	OceanStor Dorado V3	OceanStor V6	OceanStor Dorado V6
Static Provisioning	SAN: FC/ iSCSI[2] NAS: NFS 3	SAN: FC/ iSCSI[2] NAS: NFS 3	SAN: FC/ iSCSI[2]	SAN: FC/ iSCSI/NVMe over RoCE/ NVMe over FC[3] NAS: NFS 3/4.0/4.1	SAN: FC/ iSCSI/NVMe over RoCE/ NVMe over FC[3] NAS: NFS 3/4.0/4.1[4]
Dynamic Provisioning					
Manage Provisioning[1]					
Expand Persistent Volume	Volumes created in Dynamic Provisioning or Manage Provisioning mode are supported.				
Create VolumeSnapshot	Volumes created in Dynamic Provisioning or Manage Provisioning mode are supported.				
Delete VolumeSnapshot	Supported	Supported	Supported	Supported	Supported
Restore VolumeSnapshot	Supported	Supported	Supported	SAN: supported NAS: supported only in 6.1.5 and later versions	SAN: supported NAS: supported only in 6.1.5 and later versions
Clone Persistent Volume	Non-HyperMetro volumes created in Dynamic Provisioning or Manage Provisioning mode are supported.			SAN: supports non-HyperMetro volumes created in Dynamic Provisioning or Manage Provisioning mode. NAS: Only 6.1.5 and later versions support non-HyperMetro volumes created in Dynamic Provisioning or Manage Provisioning mode.	
Raw Block Volume	Only SAN volumes are supported.	Only SAN volumes are supported.	Only SAN volumes are supported.	Only SAN volumes are supported.	Only SAN volumes are supported.

Feature	OceanStor V3	OceanStor V5	OceanStor Dorado V3	OceanStor V6	OceanStor Dorado V6
Topology	Supported	Supported	Supported	Supported	Supported
Generic Ephemeral Volumes	Supported	Supported	Supported	Supported	Supported
Access Mode	RWO/ROX/RWOP: supported by all types of volumes. RWOP is supported only by Kubernetes 1.22 and later versions. RWX: supported only by Raw Block volumes and NFS volumes.				
QoS	Supported[5]	Supported[5]	Supported	Supported	Supported
Application type	N/A	N/A	N/A	Supported	Supported
Volume HyperMetro	Not supported	Not supported	N/A	Only NAS volumes are supported.	
Storage multi-tenant	Only NAS volumes are supported.		N/A	Only NAS volumes are supported.[6]	

- Note 1: Manage Provisioning is a volume management feature customized by Huawei CSI. This feature allows existing storage resources to be managed by Kubernetes. A storage resource cannot be managed for multiple times.
- Note 2: If the user's container platform is deployed in a virtualization environment, only iSCSI networking is supported.
- Note 3: If NVMe over RoCE or NVMe over FC is used, the version of the nvme-cli tool on worker nodes must be 1.9 or later. To query the version, run the **nvme version** command.
- Note 4: Only OceanStor Dorado V6 6.1.0 and later versions support NFS. Only OceanStor Dorado V6 6.1.3 and later versions support NFS 4.1.
- Note 5: Only system users can configure QoS.
- Note 6: Only OceanStor Dorado V6 6.1.3 and later versions support multi-tenant.

Huawei CSI plug-in supports the following Dtree features for Huawei enterprise storage.

Table 2-6 Features supported by Dtree

Feature	Supported
Static Provisioning	√

Feature	Supported
Dynamic Provisioning	√
Expand Persistent Volume	√
Access Mode	√ (RWX/RWO/ROX/RWOP: Kubernetes 1.22 or later supports RWOP.)
Multi-tenancy	√
Create VolumeSnapshot	X
Delete VolumeSnapshot	X
Restore VolumeSnapshot	X
Clone Persistent Volume	X
QoS	X
Volume HyperMetro	X
Application type	X

Table 2-7 Huawei storage versions supported by Dtree

Storage Product	Version
OceanStor Dorado V6	6.1.0, 6.1.2, 6.1.3, 6.1.5, 6.1.6
OceanStor Dorado V3	Not supported

2.4 Compatibility with Huawei Distributed Storage

Huawei CSI plug-in is compatible with Huawei OceanStor series distributed storage systems. The following table lists the supported storage versions.

Table 2-8 Supported Huawei distributed storage

Storage Product	Version
FusionStorage Block	8.0.0, 8.0.1
OceanStor Pacific series	8.1.0, 8.1.1, 8.1.2, 8.1.3, 8.1.5

Huawei CSI plug-in supports the following features for Huawei distributed storage.

Table 2-9 Features supported by Huawei distributed storage and constraints

Feature	FusionStorage	FusionStorage Block	OceanStor Pacific Series
Static Provisioning	SAN: iSCSI/SCSI	SAN: iSCSI/SCSI	SAN: iSCSI/SCSI NAS: DPC[2]/NFS 3/4.1[3]
Dynamic Provisioning			
Manage Provisioning[1]			
Expand Persistent Volume	Volumes created in Dynamic Provisioning or Manage Provisioning mode are supported.		
Create VolumeSnapshot	SAN volumes created in Dynamic Provisioning or Manage Provisioning mode are supported.		
Delete VolumeSnapshot	Supported	Supported	Only SAN volume snapshots are supported.
Restore VolumeSnapshot	Supported	Supported	Only SAN volume snapshots are supported.
Clone Persistent Volume	SAN volumes created in Dynamic Provisioning or Manage Provisioning mode are supported.		
Raw Block Volume	Only SAN volumes are supported.	Only SAN volumes are supported.	Only SAN volumes are supported.
Topology	Supported	Supported	Supported
Generic Ephemeral Volumes	Supported	Supported	Supported
Access Mode	RWO/ROX/RWOP: supported by all types of volumes. RWOP is supported only by Kubernetes 1.22 and later versions. RWX: supported only by Raw Block volumes and NFS volumes.		
QoS	Supported	Supported	Supported
Soft and hard quotas	Not supported	Not supported	Only NAS volumes are supported.
Storage multi-tenant	Not supported	Not supported	Only NAS volumes are supported.

- Note 1: Manage Provisioning is a volume management feature customized by Huawei CSI. This feature allows existing storage resources to be managed by Kubernetes. A storage resource cannot be managed for multiple times.

- Note 2: Only OceanStor Pacific series 8.1.2 and later versions support DPC. For details about whether the **OSs supported by Huawei CSI** support DPC, see the compatibility document of the corresponding product version.
- Note 3: Only OceanStor Pacific series 8.1.2 and later versions support NFS 4.1.

3 Installation Preparations

This chapter describes the preparations for the installation.

3.1 Prerequisites

Before performing the operations described in this chapter, ensure that the following conditions are met:

- A container management platform has been deployed and is running properly, and its compatibility meets the requirements described in **2.1 Kubernetes and OS Compatibility**.
- (Mandatory for enterprise storage) Initial configuration for interconnecting with Huawei enterprise storage has been completed, including storage pool division and port configuration. The version of the storage product meets the requirements in **2.3 Compatibility with Huawei Enterprise Storage**.
- (Mandatory for distributed storage) Initial configuration for interconnecting with Huawei distributed storage has been completed, including storage pool division and port configuration. The version of the storage product meets the requirements in **2.4 Compatibility with Huawei Distributed Storage**.
- If a multipathing network is used, ensure that multipathing software has been installed on all worker nodes. For details, see **Table 2-2**.
- The connectivity between Huawei storage and the container platform host has been configured. For example, the worker node running huawei-csi-

controller communicates properly with the management IP address of the storage device to be connected, and the worker node running huawei-csi-node communicates properly with the service IP address of the storage device to be connected. In iSCSI scenarios, the **ping** command can be used to verify the connectivity.

- Software clients required by the corresponding protocol, such as iSCSI and NFS clients, have been installed on all worker nodes of the container cluster.
- Ensure that the language of the operating system is English.

3.2 Downloading Huawei CSI Software Package

This section describes how to download the software package and the component structure of the software package.

Step 1 Open a browser and enter **https://github.com/Huawei/eSDK_K8S_Plugin/releases** in the address box.

Step 2 Download the software package of the 4.2.0 version based on the storage type and CPU architecture.

NOTE

Software package naming rule: Storage type + Plug-in name (**Kubernetes_CSI_Plugin**) + Version number + CPU architecture

For example, if distributed storage is used to connect to x86 hosts, the name of the software package to be downloaded is **eSDK_Huawei_Storage_Kubernetes_CSI_Plugin_V4.2.0_X86_64.zip**.

Step 3 Decompress the downloaded software package. The following table shows the component structure of the software package.

Table 3-1 Component description

Component	Description
image/huawei-csi-v4.2.0-*arch*.tar	huawei-csi-driver image. *arch* is **x86** or **arm**.
image/storage-backend-controller-v4.2.0-*arch*.tar	Back-end management controller image. *arch* is **x86** or **arm**.
image/storage-backend-sidecar-v4.2.0-*arch*.tar	Back-end management sidecar image. *arch* is **x86** or **arm**.
bin/	Binary file used by an image provided by Huawei.
bin/oceanctl	Command line tool provided by Huawei, which can be used to manage storage backends.
helm/	Helm project used to deploy Huawei CSI.
manual/	Used to manually install and deploy Huawei CSI.
examples/	.yaml sample file used during CSI use.

Component	Description
examples/backend	.yaml sample file used to create a storage backend.
tools/	Script used to upload images when no image repository is available.

----End

3.3 Uploading a Huawei CSI Image

Huawei provides the **huawei-csi** image for users. For details about how to obtain the image file, see **3.2 Downloading Huawei CSI Software Package**.

To use the CSI image on the container management platform, you need to import the CSI image to the cluster in advance using either of the following methods:

- (Recommended) Use Docker to upload the CSI image to the image repository.
- Use the image upload script to import the CSI image to all nodes where Huawei CSI needs to be deployed.

3.3.1 Uploading an Image to the Image Repository

The installation of Huawei CSI depends on the following three image files provided by Huawei. Import and upload the image files in sequence. For details about how to obtain the image files, see **3.2 Downloading Huawei CSI Software Package**.

- huawei-csi-v4.2.0-arch.tar
- storage-backend-controller-v4.2.0-*arch*.tar
- storage-backend-sidecar-v4.2.0-*arch*.tar

Prerequisites

A Linux host with Docker installed is available, and the host can access the image repository.

Procedure

Step 1 Run the **docker load -i huawei-csi.tar** command to import the CSI image to the current node.

```
# docker load -i huawei-csi.tar
Loaded image: huawei-csi:4.2.0
```

Step 2 Run the **docker tag huawei-csi:4.2.0 repo.huawei.com/huawei-csi:4.2.0** command to add the image repository address to the image tag. **repo.huawei.com** indicates the image repository address.

```
# docker tag huawei-csi:4.2.0 repo.huawei.com/huawei-csi:4.2.0
```

Step 3 Run the **docker push repo.huawei.com/huawei-csi:4.2.0** command to upload the CSI image to the image repository. **repo.huawei.com** indicates the image repository address.


```
# docker push repo.huawei.com/huawei-csi:4.2.0
```

----End

NOTICE

- You can also use containerd to import and upload the images.
- For details about how to import and upload images to the CCE/CCE Agile platform, see the user manual of the platform.

3.3.2 Importing an Image to All Nodes

If the image has been uploaded to the image repository, skip this section.

Prerequisites

- The host where the image is located can communicate with all hosts to which the image is to be imported using SSH.
- The **expect**, **sshpass**, and **scp** software packages have been installed on the host where the image is located.

Procedure

Step 1 Run the **vi** *worker-list.txt* command to create and open the **worker-list.txt** configuration file.

```
# vi worker-list.txt
```

Step 2 Configure the **worker-list.txt** file. The template of the **worker-list.txt** file is as follows. Press **I** or **Insert** to enter the editing mode and add node information. After the modification is complete, press **Esc** and enter **:wq!** to save the modification.

```
# ip
192.168.128.16
192.168.128.17
```

Step 3 Upload and import an image.

- If containerd is used as the container runtime, run the **./containerd-upload.sh** *worker-list.txt huawei-csi.tar* command, enter the user name and password as prompted, and upload and import an image.
  ```
  # ./containerd-upload.sh worker-list.txt huawei-csi.tar
  ```
- If Docker is used as the container runtime, run the **./docker-upload.sh** *worker-list.txt huawei-csi.tar* command, enter the user name and password as prompted, and upload and import an image.
  ```
  # ./docker-upload.sh worker-list.txt huawei-csi.tar
  ```

Step 4 Check whether the image is successfully imported.

1. If **All images are uploaded successfully** is displayed at the end of the script, the image has been successfully imported to all nodes.
   ```
   All images are uploaded successfully
   ```
2. If the following information is displayed at the end of the script, the image fails to be imported to the nodes in the list.


```
List of nodes to which the image fails to be imported:
  192.168.128.16
  192.168.128.17
```

----End

3.4 Checking the Images on Which CSI Depends

The installation of Huawei CSI depends on the images listed in the following table. If all worker nodes in the cluster have been connected to the Internet and can pull images online, skip this section. If nodes in the cluster cannot connect to the Internet, download the corresponding image file based on the Kubernetes version and upload it to the image repository or import it to all worker nodes in the Kubernetes cluster.

The huawei-csi-controller service depends on the following sidecar images: livenessprobe, csi-provisioner, csi-attacher, csi-resizer, csi-snapshotter, and snapshot-controller.

The huawei-csi-node service depends on the following sidecar images: livenessprobe, csi-node-driver-registrar, and huawei-csi-driver.

For details about the functions and details of each image, see the following table.

Table 3-2 Images on which Huawei CSI depends

Container Name	Container Image	K8s Version Requirements	Feature Description
livenessprobe	k8s.gcr.io/sig-storage/livenessprobe:v2.5.0	v1.13+	Monitors the health status of CSI and reports it to Kubernetes so that Kubernetes can automatically detect CSI program problems and restart the Pod to rectify the problems.
csi-resizer	k8s.gcr.io/sig-storage/csi-resizer:v1.4.0	v1.13+	Calls CSI to provide more storage space for a PVC when expanding the capacity of the PVC.
csi-node-driver-registrar	k8s.gcr.io/sig-storage/csi-node-driver-registrar:v2.3.0	v1.13+	Obtains CSI information and registers a node with kubelet using the plug-in registration mechanism of kubelet so that Kubernetes can detect the connection between the node and Huawei storage.

Container Name	Container Image	K8s Version Requirements	Feature Description
csi-snapshotter	k8s.gcr.io/sig-storage/csi-snapshotter:v4.2.1	v1.17+	Calls CSI to create or delete a snapshot on the storage system when creating or deleting a VolumeSnapshot.
snapshot-controller	k8s.gcr.io/sig-storage/snapshot-controller:v4.2.1	v1.17+	Listens to the VolumeSnapshot and VolumeSnapshotContent objects in the Kubernetes API and triggers csi-snapshotter to create a snapshot on the storage system when creating or deleting a VolumeSnapshot.
csi-provisioner	k8s.gcr.io/sig-storage/csi-provisioner:v3.0.0	v1.17+	• Calls the CSI Controller service to create a LUN or file system on the storage system as a PV and bind the PV to a PVC when creating a PVC. • Calls the CSI Controller service to unbind a PV from a PVC and delete the LUN or file system corresponding to the PV when deleting a PVC.
	quay.io/k8scsi/csi-provisioner:v1.4.0	v1.16.x	
csi-attacher	k8s.gcr.io/sig-storage/csi-attacher:v3.4.0	v1.17+	Calls the CSI Controller service to perform the "Publish/Unpublish Volume" operation when creating or deleting a Pod.
	quay.io/k8scsi/csi-attacher:v1.2.1	v.1.16.x	

NOTE

For details about how to download container images to the local host, see **11.13 How Do I Download a Container Image to the Local PC?**.

3.5 Checking Volume Snapshot-Dependent Components

This section describes how to check the volume snapshot-dependent components in the cluster.

> **CAUTION**
>
> Kubernetes earlier than v1.17.0 does not support the snapshot function. If the snapshot CRD is deployed, the cluster may be faulty. Therefore, if Huawei CSI is deployed on Kubernetes earlier than v1.17.0, perform the check according to **Kubernetes Earlier Than v1.17.0**.

Kubernetes Earlier Than v1.17.0

If the Kubernetes version is earlier than v1.17.0, the cluster may be faulty during snapshot deployment. Perform the following steps to delete the snapshot CRD installation file.

Step 1 Run the following command to check the Kubernetes version. In the following example, the Kubernetes version is v1.16.0.

```
# kubectl get node
NAME          STATUS      ROLES    AGE    VERSION
test-master   Ready       master   311d   v1.16.0
test-node     Ready       <none>   311d   v1.16.0
```

Step 2 Go to the **/helm/esdk/crds/snapshot-crds** directory and run the following command to delete the snapshot CRD installation file. For details about the component package path, see **Table 3-1**.

```
# rm -rf ./huawei-csi-snapshot-crd-v1.yaml
```

----End

Kubernetes v1.17.0 or Later

If you need to use volume snapshots and features associated with volume snapshots in the container environment, perform the following steps to check whether volume snapshot-dependent components have been deployed in your environment and check the api-versions information about volume snapshots.

Step 1 Run the following command to view the installation details of snapshot-related resource services.

```
# kubectl api-resources | grep snapshot | awk '{print $1}'
volumesnapshotclasses
volumesnapshotcontents
volumesnapshots
```

- If the preceding command output is displayed, the snapshot-dependent component services have been installed. In this case, go to **Step 2** and check the api-versions information.
- If any of the services in the preceding command output is not displayed, go to **Step 3** and install the snapshot-dependent component service.

Step 2 Run the following command to query the api-versions information about volume snapshots.

```
# kubectl api-versions | grep "snapshot.storage.k8s.io"
snapshot.storage.k8s.io/v1
snapshot.storage.k8s.io/v1beta1
```

- If the preceding command output is displayed, the snapshot-dependent component services support v1 and v1beta1. In this case, skip this section.
- If any of the services in the preceding command output is not displayed, go to the next step to install it.

Step 3 Go to the **/helm/esdk/crds/snapshot-crds** directory and run the following command to install the snapshot-dependent component services. For details about the component package path, see **Table 3-1**.

```
# kubectl apply -f huawei-csi-snapshot-crd-v1.yaml --validate=false
Warning: resource volumesnapshots/mysnapshot is missing the kubectl.kubernetes.io/last-applied-configuration annotation which is required by kubectl apply. kubectl apply should only be used on resources created declaratively by either kubectl create --save-config or kubectl apply. The missing annotation will be patched automatically.
volumesnapshot.snapshot.storage.k8s.io/mysnapshot configured
```

----End

After the installation is complete, you can run the command in **Step 1** to check the installation details of snapshot-related resource services.

3.6 Checking the Host Multipathing Configuration

If you plan to use the FC/iSCSI/NVMe over RoCE/NVMe over FC protocol to access Huawei storage in a container environment, you are advised to use host multipathing software to enhance the link redundancy and performance of the host and storage. If you do not want to use the software, skip this section.

For details about the OSs and multipathing software supported by Huawei CSI, see **Table 2-2**.

NOTE

- If you want to use the FC/iSCSI protocol to connect to Huawei storage, you are advised to use native DM-Multipath provided by the OS.
- If you want to use the NVMe over RoCE/NVMe over FC protocol to connect to Huawei storage, you are advised to use Huawei-developed UltraPath-NVMe.
- If you want to use the SCSI protocol to connect to Huawei storage, disable DM-Multipath provided by the OS.

Prerequisites

Multipathing software has been correctly installed on a host.

- If you use native DM-Multipath provided by the OS, contact your host or OS provider to obtain the documents and software packages required for the installation.
- If you use Huawei-developed UltraPath or UltraPath-NVMe, contact Huawei engineers to obtain the UltraPath or UltraPath-NVMe documents and software packages. For details about the software package versions, see **Table 2-2**.

Procedure

Step 1 If you use the iSCSI/FC protocol to connect to Huawei enterprise storage, configure and check host multipathing by referring to **Configuring Multipathing** > **Non-HyperMetro Scenarios** in ***OceanStor Dorado 6.x and OceanStor 6.x Host Connectivity Guide for Red Hat***.

Step 2 If you use the NVMe over RoCE/NVMe over FC protocol to connect to Huawei enterprise storage, configure and check host multipathing by referring to

Configuring Multipathing > **Non-HyperMetro Scenarios** > **UltraPath** in *OceanStor Dorado 6.x and OceanStor 6.x Host Connectivity Guide for Red Hat*.

Step 3 If you use iSCSI to connect to Huawei distributed storage, configure and check host multipathing by referring to **Configuring Multipathing for an Application Server** in *FusionStorage 8.0.1 Block Storage Basic Service Configuration Guide*.

Step 4 If you use the native multipathing software provided by the OS, check whether the **/etc/multipath.conf** file contains the following configuration item.

```
defaults {
        user_friendly_names yes
        find_multipaths no
}
```

If the configuration item does not exist, add it to the beginning of the **/etc/multipath.conf** file.

NOTE

For details about the functions of the **user_friendly_names** and **find_multipaths** parameters, see **dm_multipath/config_file_defaults**.

----End

3.7 Checking the Accounts on Huawei Storage

After Huawei storage is connected to the container platform, Huawei CSI needs to manage storage resources on Huawei storage based on service requirements, such as creating and mapping volumes. In this case, Huawei CSI needs to use the accounts created on Huawei storage to communicate with Huawei storage. The following table lists the accounts required for different storage devices.

Table 3-3 Account requirements for connecting storage to CSI

Storage Type	User Type	Role	Level	Type
OceanStor V3/V5	System user	Administrator	Administrator	Local user
	vStore user	vStore administrator	Administrator	Local user
OceanStor Dorado V3	System user	Administrator	Administrator	Local user
OceanStor 6.1.3, 6.1.5, 6.1.6	System user	Administrator/User-defined role[1]	N/A	Local user
OceanStor Dorado 6.1.0, 6.1.2, 6.1.3, 6.1.5, 6.1.6	System user	Administrator/User-defined role[1]	N/A	Local user

Storage Type	User Type	Role	Level	Type
	vStore user	vStore administrator	N/A	Local user
OceanStor Pacific series	System user	Administrator	N/A	Local user

- Note 1: If a user-defined role is used, you need to configure permissions for the role. For details about how to configure the minimum permissions, see **12.5 Configuring Custom Permissions**.

3.8 Checking the Status of Host-Dependent Software

This section describes how to check whether the status of host-dependent software on worker nodes in a cluster is normal. In this example, the host OS is CentOS 7.9 x86_64.

- Check the status of the iSCSI client.
  ```
  # systemctl status iscsi iscsid
  ```
- Check the status of the NFS client.
  ```
  # systemctl status rpcbind
  ```
- Check the status of DM-Multipath.
  ```
  # systemctl status multipathd.socket multipathd
  ```
- Check the status of UltraPath.
  ```
  # systemctl status nxup
  ```
- Check the status of UltraPath-NVMe.
  ```
  # systemctl status upudev upService_plus
  ```

3.9 Communication Matrix

Contact Huawei engineers to obtain the document from Huawei support website. For details about enterprise storage, see eSDK Enterprise Storage Plugins 2.5.0. For details about distributed storage, see eSDK Cloud Storage Plugins 2.5.RC4.

4 Installation and Deployment

This chapter describes how to install Huawei CSI on the Kubernetes, OpenShift, Tanzu, CCE, and CCE Agile platforms.

4.1 Installing Huawei CSI Using Helm

4.1.1 Preparing the values.yaml File

When using Helm to install CSI, you need to prepare the **values.yaml** file of the Helm project based on the features required during deployment. Huawei CSI provides the **values.yaml** template file in the **helm/esdk** directory of the software package.

This section describes the configuration items in the **values.yaml** file and backend configuration examples in typical scenarios.

4.1.1.1 images Parameters

The images parameters in the **values.yaml** file are used to configure the component image information on which Huawei CSI depends during running. Set the following parameters:

Table 4-1 images parameters

Parameter	Description	Mandatory	Default Value
images.huaweiCSIService	huawei-csi image.	Yes	huawei-csi:4.2.0
images.storageBackendSidecar	Huawei back-end management sidecar image.	Yes	storage-backend-sidecar:4.2.0

Parameter	Description	Mandatory	Default Value
images.storageBackendController	Huawei back-end management controller image.	Yes	storage-backend-controller:4.2.0
images.sidecar.livenessProbe	**livenessprobe** sidecar image.	Yes	k8s.gcr.io/sig-storage/livenessprobe:v2.5.0
images.sidecar.provisioner	**csi-provisioner** sidecar image.	Yes	k8s.gcr.io/sig-storage/csi-provisioner:v3.0.0
images.sidecar.attacher	**csi-attacher** sidecar image.	Yes	k8s.gcr.io/sig-storage/csi-attacher:v3.4.0
images.sidecar.resizer	**csi-resizer** sidecar image.	Yes	k8s.gcr.io/sig-storage/csi-resizer:v1.4.0
images.sidecar.snapshotter	**csi-snapshotter** sidecar image.	Yes	k8s.gcr.io/sig-storage/csi-snapshotter:v4.2.1
images.sidecar.snapshotController	**snapshot-controller** sidecar image.	Yes	k8s.gcr.io/sig-storage/snapshot-controller:v4.2.1
images.sidecar.registrar	**csi-node-driver-registrar** sidecar image.	Yes	k8s.gcr.io/sig-storage/csi-node-driver-registrar:v2.3.0

NOTICE

- For details about the values of **huaweiCSIService**, **storageBackendSidecar**, and **storageBackendController**, see **3.3 Uploading a Huawei CSI Image**. Use the name and version of the finally generated image.
- For details about other sidecar image parameters, see **3.4 Checking the Images on Which CSI Depends**. Use the name and version of the finally uploaded image.

4.1.1.2 controller Parameters

The controller parameters are used to configure the huawei-csi-controller component.

Table 4-2 controller parameters

Parameter	Description	Mandatory	Default Value	Remarks
controller.controllerCount	Number of huawei-csi-controller component copies.	Yes	1	-

Parameter	Description	Mandatory	Default Value	Remarks
controller.volumeNamePrefix	PV name prefix. The default value is **pvc**, that is, the name of a created PV is **pvc-***<uuid>*. The prefix must comply with the naming rules of a **DNS subdomain name**, and the total length of the PV name cannot exceed 253 characters.	No	pvc	The corresponding provisioner parameter name is **--volume-name-prefix**. For details, see **Configuring the PV Name Prefix**. • If the connected backend is OceanStor V3/V5 SAN storage, it is recommended that the prefix contain a maximum of 5 characters. • If the connected backend is OceanStor V3/V5 NAS storage, the prefix can contain only lowercase letters, hyphens (-), and digits. • If the connected backend is OceanStor Dorado V6 or converged V6 storage, the prefix can contain only lowercase letters, hyphens (-), and digits. • If the connected backend is OceanStor Pacific series storage, the

Parameter	Description	Mandatory	Default Value	Remarks
				prefix can contain a maximum of 95 characters.
controller.webhookPort	Port used by the webhook service.	Yes	4433	If a port conflict occurs, change the port number to one that does not conflict with the port number.
controller.snapshot.enabled	Whether to enable the snapshot feature.	Yes	true	If you want to use snapshot-related functions, enable this feature. The Kubernetes version must be later than v1.17.
controller.resizer.enabled	Whether to enable the capacity expansion feature.	Yes	true	The Kubernetes version must be later than v1.16.
controller.nodeSelector	Node selector of huawei-csi-controller. After this parameter is set, huawei-csi-controller will be scheduled only to a node with the label.	No	-	For details about the node selector, see **Assign Pods to Nodes**.
controller.tolerations	Taint toleration of huawei-csi-controller. After this parameter is set, huawei-csi-controller can tolerate taints on a node.	No	-	For details about taints and tolerations, see **Taints and Tolerations**.

NOTE

If **controller.snapshot.enabled** is set to **true**, when the **helm install** command is executed, the system automatically reads the volume snapshot CRD in the **helm/crd** directory and installs the snapshot CRD resource.

4.1.1.3 node Parameters

The node parameters are used to configure the huawei-csi-node component.

Table 4-3 node parameters

Parameter	Description	Mandatory	Default Value	Remarks
node.maxVolumesPerNode	Maximum number of volumes provisioned by Huawei CSI that can be used by a node. If this parameter is not specified or is set to **0**, the number is unlimited. If **nodeName** is specified during Pod creation, this configuration will be ignored.	No	100	For details, see **Volume Limits**.
node.nodeSelector	Node selector of huawei-csi-node. After this parameter is set, huawei-csi-node will be scheduled only to a node with the label.	No	-	For details about the node selector, see **Assign Pods to Nodes**.
node.tolerations	Taint toleration of huawei-csi-node. After this parameter is set, huawei-csi-node can tolerate taints on a node.	No	- key: "node.kubernetes.io/memory-pressure" operator: "Exists" effect: "NoExecute" - key: "node.kubernetes.io/disk-pressure" operator: "Exists" effect: "NoExecute" - key: "node.kubernetes.io/network-unavailable" operator: "Exists" effect: "NoExecute"	For details about taints and tolerations, see **Taints and Tolerations**.

4.1.1.4 csiDriver Parameters

The csiDriver parameters include the basic configurations for running Huawei CSI, such as Huawei driver name and multipathing type.

Table 4-4 csiDriver parameters

Parameter	Description	Mandatory	Default Value	Remarks
csiDriver.driverName	Registered driver name.	Yes	csi.huawei.com	• Use the default value. • For the CCE Agile platform, modify this field. For example, **csi.oceanstor.com**.
csiDriver.endpoint	Communication endpoint.	Yes	/csi/csi.sock	Use the default value.
csiDriver.connectorThreads	Maximum number of disks that can be concurrently scanned/ detached. The value is an integer ranging from 1 to 10.	Yes	4	A larger value indicates that more concurrent disk scanning and detaching operations are performed on a single node at the same time. When DM-Multipath is used, a large number of concurrent requests may cause unknown problems and affect the overall time.
csiDriver.volumeUseMultipath	Whether to use multipathing software. The value is a Boolean value.	Yes	true	It is strongly recommended that multipathing software be enabled to enhance the redundancy and performance of storage links.

Parameter	Description	Mandatory	Default Value	Remarks
csiDriver.scsiMultipathType	Multipathing software used when the storage protocol is **fc** or **iscsi**. The following parameter values can be configured: ● DM-multipath ● HW-UltraPath ● HW-UltraPath-NVMe	Mandatory when **volumeUseMultipath** is set to **true**.	DM-multipath	The **DM-multipath** value is recommended.
csiDriver.nvmeMultipathType	Multipathing software used when the storage protocol is **roce** or **fc-nvme**. Only **HW-UltraPath-NVMe** is supported.	Mandatory when **volumeUseMultipath** is set to **true**.	HW-UltraPath-NVMe	-
csiDriver.scanVolumeTimeout	Timeout interval for waiting for multipathing aggregation when DM-Multipath is used on the host. The value ranges from 1 to 600 seconds.	Yes	3	-

Parameter	Description	Mandatory	Default Value	Remarks
csiDriver.execCommandTimeout	Timeout interval for running commands on the host.	Yes	30	In scenarios such as mounting and capacity expansion, the CSI plug-in needs to run some host commands, for example, running the **mount** command to mount a file system. This parameter is used to control the timeout interval for running a single command.

Parameter	Description	Mandatory	Default Value	Remarks
csi_driver.allPathOnline	Whether to check whether the number of paths aggregated by DM-Multipath is equal to the actual number of online paths. The following parameter values can be configured: • **true**: The drive letter mounting condition is met only when the number of paths aggregated by DM-Multipath is equal to the actual number of online paths. • **false**: By default, the number of paths aggregated by DM-Multipath is not checked. As long as virtual drive letters are generated upon aggregation, the drive letter	This parameter is mandatory when **csi_driver.scsiMultipathType** is set to **DM-multipath**.	false	-

Parameter	Description	Mandatory	Default Value	Remarks
	mounting condition is met.			
csiDriver.enableLabel	Whether Kubernetes resource object labels can be delivered to storage.	Yes	false	Only centralized storage 6.1.7 and later versions support this function. Distributed storage, dtrees, and static PVs do not support this function.
csiDriver.backendUpdateInterval	Interval for updating backend capabilities. The value ranges from 60 to 600 seconds.	Yes	60	-
csiDriver.controllerLogging.module	Record type of the controller log. The following parameter values can be configured: • file • console	Yes	file	When the value is **file**, logs are retained in the specified directory of the node. When the Pod where CSI is located is destroyed, logs are still retained. When the value is **console**, logs are retained in the temporary space of the Pod where CSI is located. When the Pod where CSI is located is destroyed, the logs are also destroyed.

Parameter	Description	Mandatory	Default Value	Remarks
csiDriver.controllerLogging.level	Output level of the controller log. The following parameter values can be configured: • debug • info • warning • error • fatal	Yes	info	-
csiDriver.controllerLogging.fileDir	Directory of the controller log in **file** output mode.	Yes	/var/log/huawei	Ensure that the directory has sufficient space for storing logs. It is recommended that the space be greater than or equal to 200 MB.
csiDriver.controllerLogging.fileSize	Size of a single controller log file in **file** output mode.	Yes	20M	-
csiDriver.controllerLogging.maxBackups	Maximum number of controller log file backups in **file** output mode.	Yes	9	-

Parameter	Description	Mandatory	Default Value	Remarks
csiDriver.nodeLogging.module	Record type of the node log. The following parameter values can be configured: • file • console	Yes	file	When the value is **file**, logs are retained in the specified directory of the node. When the Pod where CSI is located is destroyed, logs are still retained. When the value is **console**, logs are retained in the temporary space of the Pod where CSI is located. When the Pod where CSI is located is destroyed, the logs are also destroyed.
csiDriver.nodeLogging.level	Output level of the node log. The following parameter values can be configured: • debug • info • warning • error • fatal	Yes	info	-
csiDriver.nodeLogging.fileDir	Directory of the node log in **file** output mode.	Yes	/var/log/huawei	Ensure that the directory has sufficient space for storing logs. It is recommended that the space be greater than or equal to 200 MB.

Parameter	Description	Mandatory	Default Value	Remarks
csiDriver.nodeLogging.fileSize	Size of a single node log file in **file** output mode.	Yes	20M	-
csiDriver.nodeLogging.maxBackups	Maximum number of node log file backups in **file** output mode.	Yes	9	-

CAUTION

If Huawei CSI has been deployed in your container environment, ensure that the value of **csiDriver.driverName** is the same as that configured during previous deployment. Otherwise, existing volumes or snapshots provisioned by Huawei CSI in the system cannot be managed by the newly deployed Huawei CSI.

4.1.1.5 Other Parameters

Other parameters include some features of the CSI plug-in or the policies for obtaining images.

Table 4-5 Other parameters

Parameter	Description	Mandatory	Default Value	Remarks
kubernetes.namespace	Kubernetes namespace where Huawei CSI is running, which can be customized. The name must consist of lowercase letters, digits, and hyphens (-), for example, **my-name** and **123-abc**.	No	huawei-csi	-

Parameter	Description	Mandatory	Default Value	Remarks
kubeletConfigDir	Working directory of kubelet.	Yes	/var/lib/kubelet	• Use the default value. • For the Tanzu platform, change the value of this field to **/var/vcap/data/kubelet**. • For the CCE Agile platform, change the value of this field to **/mnt/paas/kubernetes/kubelet**.
sidecarImagePullPolicy	Pull policy of the sidecar image.	Yes	IfNotPresent	-
huaweiImagePullPolicy	Pull policy of the huawei-csi image.	Yes	IfNotPresent	-
CSIDriverObject.isCreate	Whether to create the CSIDriver object.	Yes	false	The **CSIDriver** feature is a GA version in Kubernetes v1.18. Therefore, to use this feature, the Kubernetes version must be later than v1.18. If the Kubernetes version is earlier than v1.18, set this parameter to **false**.

Parameter	Description	Mandatory	Default Value	Remarks
CSIDriverObject.attachRequired	Whether the CSI plug-in skips the attach operation. The following parameter values can be configured: • **true**: The attach operation is required. • **false**: The attach operation is skipped.	Yes	true	The **attachRequired** parameter can be configured in Kubernetes v1.18. If **CSIDriverObject.isCreate** is set to **true** and **attachRequired** is set to **false**, the huawei-csi plug-in will not deploy the csi-attacher sidecar. • If NAS storage is used, this parameter can be set to **false**. • If SAN storage is used, set this parameter to **true**.

Parameter	Description	Mandatory	Default Value	Remarks
CSIDriverObject.fsGroupPolicy	Whether the ownership and permissions of a basic volume can be changed before the volume is mounted. The following parameter values can be configured: • **ReadWriteOnceWithFSType**: The volume ownership and permission can be changed only when **fsType** is specified and **accessModes** of the volume contains **ReadWriteOnce**. • **File**: Kubernetes can use **fsGroup** to change the permissions and ownership of a volume to match **fsGroup** requested by a user in the Pod security policy, regardless of **fsGroup** or **accessModes**. • **None**: A volume is mounted without any change. • **null**: The **fsGroupPolicy** parameter is not set.	No	null	The **fsGroupPolicy** parameter can be configured in Kubernetes v1.20, and takes effect only when **CSIDriverObject.isCreate** is set to **true**. This feature is a Beta version in Kubernetes v1.20 but a GA version in Kubernetes v1.23. Therefore, the Kubernetes version must be later than v1.20.
leaderElection.leaseDuration	Leader duration.	No	8s	This parameter takes effect only in the multi-controller scenario.

Parameter	Description	Mandatory	Default Value	Remarks
leaderElection.renewDeadline	Time for the leader to be re-elected.	No	6s	This parameter takes effect only in the multi-controller scenario.
leaderElection.retryPeriod	Leader election retry time.	No	2s	This parameter takes effect only in the multi-controller scenario.

NOTICE

Ensure that the namespace entered in **kubernetes.namespace** exists on Kubernetes. If the namespace does not exist, run the following command to create it. In this example, the namespace for running Huawei CSI is **huawei-csi**.

```
# kubectl create namespace huawei-csi
```

4.1.2 Installing Huawei CSI

This section describes how to install Huawei CSI using Helm 3.

NOTICE

- Huawei CSI can be installed as the root user or a non-root user. When installing Huawei CSI as a non-root user, ensure that the current user can access the API Server of the Kubernetes cluster. For details about how to configure access to the Kubernetes cluster as a non-root user, see **10.6 Configuring Access to the Kubernetes Cluster as a Non-root User**.
- Huawei CSI must be run as the root user.

Helm is a software package management tool in the Kubernetes ecosystem. Similar to Ubuntu APT, CentOS YUM, or Python pip, Helm manages Kubernetes application resources.

You can use Helm to package, distribute, install, upgrade, and roll back Kubernetes applications in a unified manner.

- For details about how to obtain and install Helm, see **https://helm.sh/docs/intro/install/**.
- For details about the mapping between Helm and Kubernetes versions, see **https://helm.sh/docs/topics/version_skew/**.

When installing huawei-csi-controller, Helm deploys the following components in the workloads of the Deployment type in the specified namespace:

- huawei-csi-driver: Huawei CSI driver.
- storage-backend-controller: Huawei backend management controller, used to manage storageBackendClaim resources.
- storage-backend-sidecar: used to manage storageBackendContent resources.
- Kubernetes External Provisioner: used to provide volumes.
- Kubernetes External Attacher: used to attach volumes.
- (Optional) Kubernetes External Snapshotter: used to provide snapshot support (installed as CRD).
- (Optional) Kubernetes External Snapshot Controller: used to control volume snapshots.
- Kubernetes External Resizer: used to expand the capacity of volumes.
- Kubernetes External liveness-probe: used to determine the health status of a Pod.

When installing huawei-csi-node, Helm deploys the following components in the workloads of the DaemonSet type in the specified namespace:

- huawei-csi-driver: Huawei CSI driver.
- Kubernetes Node Registrar: used to process driver registration.
- liveness-probe: used to determine the health status of a Pod.

4.1.2.1 Installing Huawei CSI on Kubernetes, OpenShift, and Tanzu

This section describes how to install Huawei CSI on the Kubernetes, OpenShift, and Tanzu platforms.

Prerequisites

- A Huawei CSI image has been created and uploaded to the image repository or imported to all nodes by following the instructions provided in **3.3 Uploading a Huawei CSI Image**.
- The component images on which Huawei CSI installation and running depend have been uploaded to the image repository or imported to all nodes. For details, see **3.4 Checking the Images on Which CSI Depends**.
- The volume snapshot component CRD on which the running of Huawei CSI depends has been installed. For details, see **3.5 Checking Volume Snapshot-Dependent Components**.
- If you want to use multipathing to connect to Huawei storage, ensure that multipathing software has been installed on all compute nodes. For details, see **3.6 Checking the Host Multipathing Configuration**.
- Helm 3 has been installed on the container management platform.
- The **values.yaml** file required for installing CSI has been prepared. For details, see **4.1.1 Preparing the values.yaml File**.
- All worker nodes of the cluster communicate properly with the service IP address of the storage device to be connected. In iSCSI scenarios, the **ping** command can be used to verify the connectivity.
- Software clients required by the corresponding protocol, such as iSCSI and NFS clients, have been installed on all worker nodes of the cluster.

- The accounts required for connecting to Huawei CSI have been created on the Huawei storage to be connected. For details, see **3.7 Checking the Accounts on Huawei Storage**.

Installation Preparations

For the OpenShift platform, run the following commands to create the **SecurityContextConstraints** resource.

- Run the **vi** *helm_scc.yaml* command to create a **SecurityContextConstraints** file. In the following command output, **huawei-csi** indicates the created namespace. Replace it based on site requirements.

```
# vi helm_scc.yaml
apiVersion: security.openshift.io/v1
kind: SecurityContextConstraints
metadata:
  name: helm-scc
allowHostDirVolumePlugin: true
allowHostIPC: true
allowHostNetwork: true
allowHostPID: true
allowHostPorts: true
allowPrivilegeEscalation: true
allowPrivilegedContainer: true

defaultAddCapabilities:
- SYS_ADMIN
runAsUser:
  type: RunAsAny
seLinuxContext:
  type: RunAsAny
fsGroup:
  type: RunAsAny
users:
- system:serviceaccount:huawei-csi:huawei-csi-controller
- system:serviceaccount:huawei-csi:huawei-csi-node
```

- Run the **oc create -f helm_scc.yaml** command to create a **SecurityContextConstraints** file.

```
# oc create -f helm_scc.yaml
```

On the Tanzu platform, run the following command to configure the **kubelet** installation directory.

- Go to the **helm/esdk** directory in the installation package, run the **vi values.yaml** command to open the configuration file, modify the file, and save the file. For details about the installation package directory, see **Table 3-1**.

```
# vi values.yaml

# Specify kubelet config dir path.
# kubernetes and openshift is usually /var/lib/kubelet
# Tanzu is usually /var/vcap/data/kubelet
# CCE is usually /mnt/paas/kubernetes/kubelet
kubeletConfigDir: /var/vcap/data/kubelet
```

For TKGI 1.16 or earlier of the Tanzu platform, run the following commands to configure the RBAC permission.

- Run the **vi** *rbac.yaml* command to create a file named *rbac.yaml*.

```
# vi rbac.yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
```



```
  name: huawei-csi-psp-role
rules:
- apiGroups: ['policy']
  resources: ['podsecuritypolicies']
  verbs: ['use']
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: huawei-csi-psp-role-cfg
roleRef:
  kind: ClusterRole
  name: huawei-csi-psp-role
  apiGroup: rbac.authorization.k8s.io
subjects:
- kind: Group
  apiGroup: rbac.authorization.k8s.io
  name: system:serviceaccounts:huawei-csi
- kind: Group
  apiGroup: rbac.authorization.k8s.io
  name: system:serviceaccounts:default
```

- Run the **kubectl create -f** *rbac.yaml* command to create the RBAC permission.

  ```
  # kubectl create -f rbac.yaml
  ```

Deployment Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the cluster through the management IP address.

Step 2 Copy the **helm** directory in the Kubernetes CSI component package to any directory on the master node. For details about the Helm tool path, see **Table 3-1**.

Step 3 Go to the **helm/esdk** working directory.

```
# cd helm/esdk
```

Step 4 Run the **kubectl apply -f** *crds/backend/* command to update the CRD.

```
# kubectl apply -f crds/backend/
customresourcedefinition.apiextensions.k8s.io/storagebackendclaims.xuanwu.huawei.io configured
customresourcedefinition.apiextensions.k8s.io/storagebackendcontents.xuanwu.huawei.io configured
```

Step 5 Ensure that snapshot-dependent components have been checked by following the instructions provided in **3.5 Checking Volume Snapshot-Dependent Components**. Then run the **helm install** *helm-huawei-csi* **./ -n** *huawei-csi* **--create-namespace** command to install Huawei CSI.

In the preceding command, *helm-huawei-csi* indicates the custom Helm chart name, **./** indicates that the Helm project in the current directory is used, and *huawei-csi* indicates the custom Helm chart namespace.

```
# helm install helm-huawei-csi ./ -n huawei-csi --create-namespace
NAME: helm-huawei-csi
LAST DEPLOYED: Wed Jun  8 11:50:28 2022
NAMESPACE: huawei-csi
STATUS: deployed
REVISION: 1
TEST SUITE: None
```

Step 6 After the huawei-csi service is deployed, run the **kubectl get pod -n** *huawei-csi* command to check whether the service is started. On the OpenShift platform, run the **oc get pod -n** *huawei-csi* command to check whether the service is started.

```
# kubectl get pod -n huawei-csi
NAME                                  READY   STATUS    RESTARTS   AGE
huawei-csi-controller-6dfcc4b79f-9vjtq   9/9     Running   0          24m
```



```
huawei-csi-controller-6dfcc4b79f-csphc   9/9     Running   0          24m
huawei-csi-node-g6f4k                    3/3     Running   0          20m
huawei-csi-node-tqs87                    3/3     Running   0          20m
```

----End

4.1.2.2 Installing Huawei CSI on CCE/CCE Agile

This section describes how to install Huawei CSI on the CCE/CCE Agile platform.

Prerequisites

- A Huawei CSI image has been created and uploaded to the image repository or imported to all nodes by following the instructions provided in **3.3 Uploading a Huawei CSI Image**.
- The component images on which Huawei CSI installation and running depend have been uploaded to the image repository or imported to all nodes. For details, see **3.4 Checking the Images on Which CSI Depends**.
- The volume snapshot component CRD on which the running of Huawei CSI depends has been installed. For details, see **3.5 Checking Volume Snapshot-Dependent Components**.
- If you want to use multipathing to connect to Huawei storage, ensure that multipathing software has been installed on all compute nodes. For details, see **3.6 Checking the Host Multipathing Configuration**.
- Helm 3 has been installed on the container management platform.
- The **values.yaml** file required for installing CSI has been prepared. For details, see **4.1.1 Preparing the values.yaml File**.
- All worker nodes of the cluster communicate properly with the service IP address of the storage device to be connected. In iSCSI scenarios, the **ping** command can be used to verify the connectivity.
- Software clients required by the corresponding protocol, such as iSCSI and NFS clients, have been installed on all worker nodes of the cluster.
- The accounts required for connecting to Huawei CSI have been created on the Huawei storage to be connected. For details, see **3.7 Checking the Accounts on Huawei Storage**.

4.1.2.2.1 Creating a Helm Installation Package

To install Huawei CSI on the CCE/CCE Agile platform, you need to create a Helm installation package. This section describes how to create a Helm installation package.

Prerequisites

- Helm 3 has been installed on a node server.
- The **values.yaml** file required for installing CSI has been prepared. For details, see **4.1.1 Preparing the values.yaml File**.

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any node where Helm is deployed through the management IP address.

Step 2 Copy the **helm** directory in the Huawei CSI component package to any directory on the node. For details about the Helm tool path, see **Table 3-1**.

Step 3 Go to the **helm** working directory.

```
# cd helm/
```

Step 4 Modify the **kubeletConfigDir** and **csiDriver.driverName** parameters in the **helm/esdk/values.yaml** file.

```
# vi ./esdk/values.yaml
# Specify kubelet config dir path.
# kubernetes and openshift is usually /var/lib/kubelet
# Tanzu is usually /var/vcap/data/kubelet
# CCE is usually /mnt/paas/kubernetes/kubelet
kubeletConfigDir: /mnt/paas/kubernetes/kubelet

# The CSI driver parameter configuration
csiDriver:
  # Driver name, it is strongly recommended not to modify this parameter
  # The CCE platform needs to modify this parameter, e.g. csi.oceanstor.com
  driverName: csi.oceanstor.com
```

Step 5 Run the **helm package ./esdk/ -d ./** command to create a Helm installation package. This command will generate the installation package to the current path.

```
# helm package ./esdk/ -d ./
Successfully packaged chart and saved it to: esdk-1.0.0.tgz
```

----End

4.1.2.2.2 Installing Huawei CSI

This section describes how to install Huawei CSI on the CCE/CCE Agile platform. The following uses CCE Agile v22.3.2 as an example.

Prerequisites

- Installation preparations have been checked according to **Prerequisites**.
- A Huawei CSI Helm installation package has been created. For details, see **4.1.2.2.1 Creating a Helm Installation Package**.

Procedure

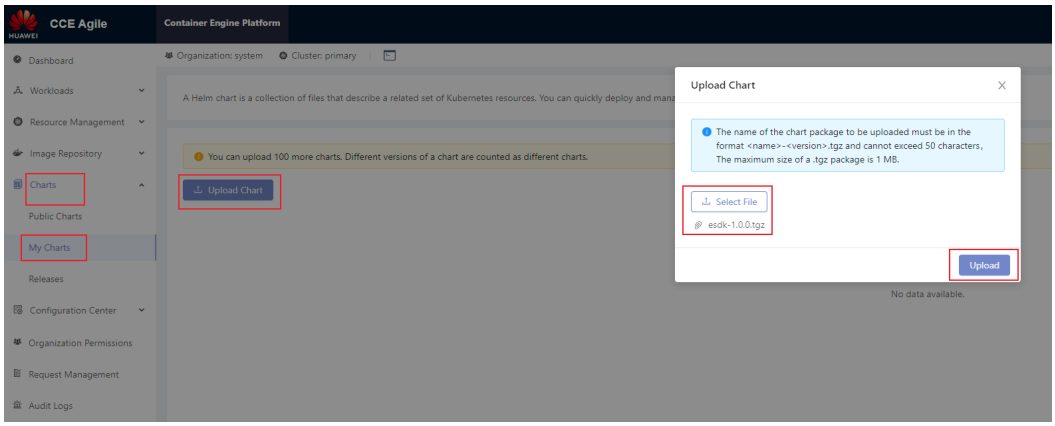
Step 1 Use a remote access tool, such as PuTTY, to log in to any master node where the CCE Agile platform is deployed through the management IP address.

Step 2 Run the **kubectl create namespace** *huawei-csi* command to create a namespace for deploying Huawei CSI. *huawei-csi* indicates the custom namespace.

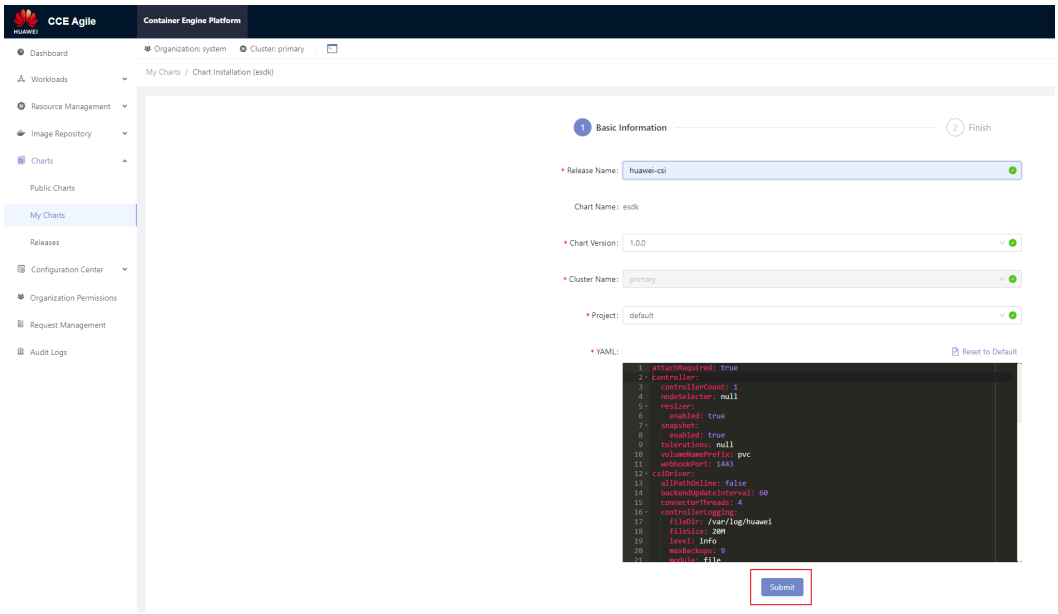
```
# kubectl create namespace huawei-csi
```

Step 3 Export the Helm installation package. For details, see **4.1.2.2.1 Creating a Helm Installation Package**.

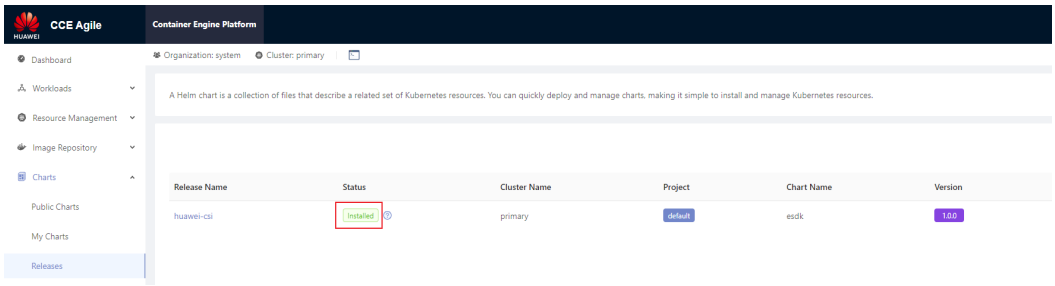
Step 4 On the home page, choose **Charts** > **My Charts** > **Upload Chart**. The **Upload Chart** dialog box is displayed. Import the exported Helm installation package to the CCE Agile platform.



Step 5 After the installation package is uploaded, choose **Charts** > **My Charts**. On the **My Charts** page that is displayed, choose **Install** > **Submit**. The chart release name can be customized.



Step 6 On the home page, choose **Charts** > **Releases** and select the project specified during installation (for example, **default** in the following figure). After the installation is successful, **Installed** is displayed in the **Status** column.



----End

4.2 Manually Installing Huawei CSI

This section describes how to manually install Huawei CSI.

NOTE

Currently, only the Kubernetes platform supports manual installation of Huawei CSI.

Prerequisites

- A Huawei CSI image has been created and uploaded to the image repository or imported to all nodes by following the instructions provided in **3.3 Uploading a Huawei CSI Image**.
- The component images on which Huawei CSI installation and running depend have been uploaded to the image repository or imported to all nodes. For details, see **3.4 Checking the Images on Which CSI Depends**.
- The volume snapshot component CRD on which the running of Huawei CSI depends has been installed. For details, see **3.5 Checking Volume Snapshot-Dependent Components**.
- If you want to use multipathing to connect to Huawei storage, ensure that multipathing software has been installed on all compute nodes. For details, see **3.6 Checking the Host Multipathing Configuration**.
- All worker nodes of the cluster communicate properly with the service IP address of the storage device to be connected. In iSCSI scenarios, the **ping** command can be used to verify the connectivity.
- Software clients required by the corresponding protocol, such as iSCSI and NFS clients, have been installed on all worker nodes of the cluster.
- The accounts required for connecting to Huawei CSI have been created on the Huawei storage to be connected. For details, see **3.7 Checking the Accounts on Huawei Storage**.

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the cluster through the management IP address.

Step 2 Copy the **manual** directory in the Kubernetes CSI component package to any directory on the master node.

Step 3 Go to the **manual/esdk** working directory. For details about the path, see **Table 3-1**.

```
# cd manual/esdk
```

Step 4 Run the **kubectl apply -f** *crds/backend/* command to install the CRD.

```
# kubectl apply -f crds/backend/
customresourcedefinition.apiextensions.k8s.io/resourcetopologies.xuanwu.huawei.io configured
customresourcedefinition.apiextensions.k8s.io/storagebackendclaims.xuanwu.huawei.io configured
customresourcedefinition.apiextensions.k8s.io/storagebackendcontents.xuanwu.huawei.io configured
```

Step 5 Run the **kubectl create -f** *./deploy* command to install and deploy CSI.

```
# kubectl create -f ./deploy
csidriver.storage.k8s.io/csi.huawei.com created
clusterrole.rbac.authorization.k8s.io/storage-backend-controller-role created
```



```
clusterrole.rbac.authorization.k8s.io/storage-backend-sidecar-role created
clusterrolebinding.rbac.authorization.k8s.io/storage-backend-controller-binding created
clusterrolebinding.rbac.authorization.k8s.io/storage-backend-sidecar-binding created
serviceaccount/huawei-csi-controller created
rolebinding.rbac.authorization.k8s.io/huawei-csi-attacher-role-cfg created
role.rbac.authorization.k8s.io/huawei-csi-attacher-cfg created
rolebinding.rbac.authorization.k8s.io/huawei-csi-provisioner-role-cfg created
role.rbac.authorization.k8s.io/huawei-csi-provisioner-cfg created
clusterrolebinding.rbac.authorization.k8s.io/huawei-csi-provisioner-role created
clusterrole.rbac.authorization.k8s.io/huawei-csi-provisioner-runner created
clusterrolebinding.rbac.authorization.k8s.io/huawei-csi-attacher-role created
clusterrole.rbac.authorization.k8s.io/huawei-csi-attacher-runner created
rolebinding.rbac.authorization.k8s.io/huawei-csi-csi-resizer-role-cfg created
role.rbac.authorization.k8s.io/huawei-csi-resizer-cfg created
clusterrolebinding.rbac.authorization.k8s.io/huawei-csi-csi-resizer-role created
clusterrole.rbac.authorization.k8s.io/huawei-csi-resizer-runner created
clusterrole.rbac.authorization.k8s.io/huawei-csi-controller-runner created
clusterrolebinding.rbac.authorization.k8s.io/huawei-csi-controller-role created
deployment.apps/huawei-csi-controller created
service/huawei-csi-controller created
serviceaccount/huawei-csi-node created
clusterrolebinding.rbac.authorization.k8s.io/huawei-csi-driver-registrar-role created
clusterrole.rbac.authorization.k8s.io/huawei-csi-driver-registrar-runner created
clusterrole.rbac.authorization.k8s.io/huawei-csi-node-runner created
clusterrolebinding.rbac.authorization.k8s.io/huawei-csi-node-role created
daemonset.apps/huawei-csi-node created
```

Step 6 After the service is deployed in **Step 5**, run the **kubectl get pod -n** *huawei-csi* command to check whether the service is started.

```
# kubectl get pod -n huawei-csi
NAME                                  READY   STATUS    RESTARTS   AGE
huawei-csi-controller-6dfcc4b79f-9vjtq   9/9     Running   0          24m
huawei-csi-controller-6dfcc4b79f-csphc   9/9     Running   0          24m
huawei-csi-node-g6f4k                    3/3     Running   0          20m
huawei-csi-node-tqs87                    3/3     Running   0          20m
```

----End

5 Uninstalling Huawei CSI

5.1 Uninstalling Huawei CSI Using Helm

This chapter describes how to uninstall Huawei CSI. The uninstallation method varies according to the installation mode.

CAUTION

If you do not uninstall Huawei CSI for the purpose of an upgrade, ensure that all resources (such as PV, PVC, snapshot, and storage backend resources) provisioned by Huawei CSI have been cleared on your container platform before uninstalling Huawei CSI. Otherwise, once you uninstall Huawei CSI, these resources cannot be automatically scheduled, managed, or cleared.

5.1.1 Uninstalling Huawei CSI on Kubernetes, OpenShift, and Tanzu

This section describes how to uninstall Huawei CSI on the Kubernetes, OpenShift, and Tanzu platforms.

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the **helm uninstall** *helm-huawei-csi* **-n** *huawei-csi* command to uninstall Huawei CSI. In the command, *helm-huawei-csi* indicates the custom Helm chart name and *huawei-csi* indicates the namespace where the Helm chart resides. This command will delete the huawei-csi-controller, huawei-csi-node, and RBAC resources of Huawei CSI.

```
# helm uninstall helm-huawei-csi -n huawei-csi
release "helm-huawei-csi" uninstalled
```


After the deletion command is executed, you need to check whether the uninstallation is successful.

```
# helm list -n huawei-csi
NAME    NAMESPACE       REVISION        UPDATED STATUS  CHART   APP VERSION
```

In the preceding command, *huawei-csi* indicates the namespace where the chart is located.

If the command output is empty, the service is successfully deleted.

Step 3 Delete the huawei-csi-host-info object. For details, see **5.1.3.1 Deleting the huawei-csi-host-info Object**.

Step 4 Delete the webhook resource. For details, see **5.1.3.2 Deleting a Webhook Resource**.

Step 5 (Optional) Uninstall the snapshot-dependent component service. For details, see **5.1.3.3 Uninstalling the Snapshot-Dependent Component Service**.

----End

5.1.2 Uninstalling Huawei CSI on CCE/CCE Agile

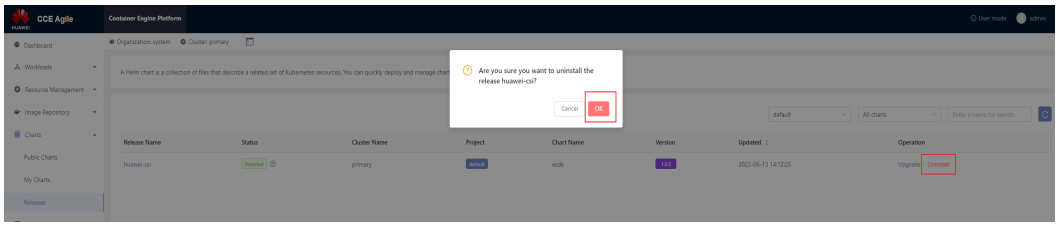
This section describes how to uninstall Huawei CSI on the CCE/CCE Agile platform. The following uses CCE Agile v22.3.2 as an example.

Procedure

Step 1 Log in to the CCE Agile platform.

Step 2 On the home page, choose **Charts** > **Releases**. The **Releases** page is displayed.

Step 3 Select a Huawei CSI release and click **Uninstall**. In the displayed dialog box, click **OK**.



Step 4 Delete the huawei-csi-host-info object. For details, see **5.1.3.1 Deleting the huawei-csi-host-info Object**.

Step 5 Delete the webhook resource. For details, see **5.1.3.2 Deleting a Webhook Resource**.

Step 6 (Optional) Uninstall the snapshot-dependent component service. For details, see **5.1.3.3 Uninstalling the Snapshot-Dependent Component Service**.

----End

5.1.3 Deleting CSI-Dependent Component Services

This section describes how to delete the CSI-dependent component services.

5.1.3.1 Deleting the huawei-csi-host-info Object

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the **kubectl delete secret** *huawei-csi-host-info* **-n** *huawei-csi* command to delete the **secret** object. *huawei-csi-host-info* is the name of the **secret** object, and *huawei-csi* is the namespace where the **secret** object is located.

```
# kubectl delete secret huawei-csi-host-info -n huawei-csi
```

Step 3 Run the following command to check whether the **secret** object is successfully deleted. If **NotFound** is displayed in the command output, the **huawei-csi-host-info** object is successfully deleted.

```
# kubectl get secret huawei-csi-host-info -n huawei-csi
Error from server (NotFound): secrets "huawei-csi-host-info" not found
```

----End

5.1.3.2 Deleting a Webhook Resource

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the following command to query the webhook-dependent component service.

```
# kubectl get validatingwebhookconfigurations.admissionregistration.k8s.io
NAME                                      WEBHOOKS   AGE
storage-backend-controller.xuanwu.huawei.io   1          12d
```

Step 3 Run the following command to uninstall the webhook-dependent component service.

```
# kubectl delete validatingwebhookconfigurations.admissionregistration.k8s.io  storage-backend-controller.xuanwu.huawei.io
```

Step 4 Run the following command to check whether the service is successfully uninstalled. If the command output is empty, the uninstallation is successful.

```
# kubectl get validatingwebhookconfigurations.admissionregistration.k8s.io
```

----End

5.1.3.3 Uninstalling the Snapshot-Dependent Component Service

CAUTION

- Do not uninstall the snapshot-dependent component service when snapshots exist. Otherwise, Kubernetes will automatically delete all user snapshots and they cannot be restored. Exercise caution when performing this operation. For details, see **Delete a CustomResourceDefinition**.
- Do not uninstall the snapshot-dependent component service during the CSI upgrade.

Scenario Description

- Currently, Huawei CSI uses the snapshot feature.
- Currently, only Huawei CSI is available in the Kubernetes cluster, and Huawei CSI is no longer used.
- Before the uninstallation, ensure that no VolumeSnapshot resource managed by Huawei CSI exists in the Kubernetes cluster.

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the following command to uninstall the snapshot-dependent component service.

```
# kubectl delete crd volumesnapshotclasses.snapshot.storage.k8s.io
volumesnapshotcontents.snapshot.storage.k8s.io volumesnapshots.snapshot.storage.k8s.io
```

Step 3 Run the following command to check whether the service is successfully uninstalled.

If the command output is empty, the uninstallation is successful.

```
# kubectl get crd | grep snapshot.storage.k8s.io
```

----End

5.2 Manually Uninstalling Huawei CSI

This section describes how to manually uninstall Huawei CSI.

CAUTION

If you do not uninstall Huawei CSI for the purpose of an upgrade, ensure that all resources (such as PV, PVC, snapshot, and storage backend resources) provisioned by Huawei CSI have been cleared on your container platform before uninstalling Huawei CSI. Otherwise, once you uninstall Huawei CSI, these resources cannot be automatically scheduled, managed, or cleared.

5.2.1 Uninstalling the huawei-csi-node Service

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the **kubectl delete daemonset huawei-csi-node -n** *huawei-csi* command to uninstall the huawei-csi-node service. Replace *huawei-csi* with the namespace where Huawei CSI is located.

```
# kubectl delete daemonset huawei-csi-node -n huawei-csi
```

Step 3 Run the following command to check whether the service is successfully uninstalled. If **NotFound** is displayed, the service is successfully uninstalled.


```
# kubectl get daemonset huawei-csi-node -n huawei-csi
```

----End

5.2.2 Uninstalling the huawei-csi-controller Service

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the **kubectl delete deployment huawei-csi-controller -n** *huawei-csi* command to uninstall the huawei-csi-controller service. Replace *huawei-csi* with the namespace where Huawei CSI is located.

```
# kubectl delete deployment huawei-csi-controller -n huawei-csi
```

Step 3 Run the following command to check whether the service is successfully uninstalled. If **NotFound** is displayed, the service is successfully uninstalled.

```
# kubectl get deployment huawei-csi-controller -n huawei-csi
```

----End

5.2.3 Uninstalling the csidriver Object

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the **kubectl delete csidriver csi.huawei.com** command to uninstall the csidriver object.

```
# kubectl delete csidriver csi.huawei.com
```

Step 3 Run the following command to check whether the service is successfully uninstalled. If **NotFound** is displayed, the service is successfully uninstalled.

```
# kubectl get csidriver csi.huawei.com
```

----End

5.2.4 Deleting the RBAC Permission

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Delete the RBAC permission.

```
# kubectl -n huawei-csi -l provisioner=csi.huawei.com delete
ServiceAccount,Service,role,rolebinding,ClusterRole,ClusterRoleBinding
```

----End

5.2.5 Uninstalling Other Resources

Procedure

Step 1 Delete the huawei-csi-host-info object. For details, see **5.1.3.1 Deleting the huawei-csi-host-info Object**.

Step 2 Delete the webhook resource. For details, see **5.1.3.2 Deleting a Webhook Resource**.

Step 3 (Optional) Uninstall the snapshot-dependent component service. For details, see **5.1.3.3 Uninstalling the Snapshot-Dependent Component Service**.

----End

6 Upgrade/Rollback Operations

This chapter describes how to upgrade or roll back Huawei CSI.

6.1 Upgrading or Rolling Back Huawei CSI Using Helm

6.1.1 Upgrading Huawei CSI

This section describes how to upgrade Huawei CSI.

During the upgrade or rollback, the existing resources such as PVCs, snapshots, and Pods will run properly and will not affect your service access.

CAUTION

- Some CSI 2.*x* versions are unavailable now. If the upgrade fails, CSI may fail to be rolled back to a version which is unavailable now.
- During the upgrade or rollback, you cannot use Huawei CSI to create new resources or mount or unmount an existing PVC.
- During the upgrade or rollback, do not uninstall the snapshot-dependent component service.

6.1.1.1 Upgrading Huawei CSI on Kubernetes, OpenShift, and Tanzu

Upgrading CSI from 2.x or 3.x to 4.2.0

To upgrade CSI from 2.*x* or 3.*x* to 4.2.0, perform the following operations:

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the **kubectl get cm huawei-csi-configmap -n** *huawei-csi* **-o json > configmap.json** command to back up the backend information to the **configmap.json** file. For the OpenShift platform, replace **kubectl** with **oc**.

```
# kubectl get cm huawei-csi-configmap -n huawei-csi -o json > configmap.json
```

Step 3 Uninstall CSI. For details, see **5.1 Uninstalling Huawei CSI Using Helm**.

Step 4 Install CSI of the current version. For details, see **4.1.2 Installing Huawei CSI**.

Step 5 Install the backend information backed up in **Step 2** according to **7.1 Adding a Storage Backend**.

----End

Upgrading CSI from 4.x to 4.2.0

To upgrade CSI from 4.*x* to 4.2.0, perform the following operations:

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Uninstall CSI. For details, see **5.1 Uninstalling Huawei CSI Using Helm**.

Step 3 Install CSI of the current version. For details, see **4.1.2 Installing Huawei CSI**.

----End

6.1.2 Rolling Back Huawei CSI

If CSI fails to be upgraded from 2.*x* or 3.*x* to 4.2.0 and needs to be rolled back, uninstall CSI by referring to **5.1 Uninstalling Huawei CSI Using Helm** and then download and install CSI of the source version.

CAUTION

- During the upgrade or rollback, the existing resources such as PVCs, snapshots, and Pods will run properly and will not affect your service access.
- During the upgrade or rollback, you cannot use Huawei CSI to create new resources or mount or unmount an existing PVC.
- During the upgrade or rollback, do not uninstall the snapshot-dependent component service.

6.1.2.1 Rolling Back Huawei CSI on Kubernetes, OpenShift, and Tanzu

Prerequisites

You have downloaded the CSI software package of the source version.

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Uninstall CSI. For details, see **5.1 Uninstalling Huawei CSI Using Helm**.

Step 3 Reinstall CSI of the original version by referring to the user guide in the CSI installation package of the original version.

----End

6.2 Manual Upgrade/Rollback

6.2.1 Upgrading Huawei CSI

This section describes how to manually upgrade Huawei CSI.

During the upgrade or rollback, the existing resources such as PVCs, snapshots, and Pods will run properly and will not affect your service access.

CAUTION

- Some CSI 2.*x* versions are unavailable now. If the upgrade fails, CSI may fail to be rolled back to a version which is unavailable now.
- During the upgrade or rollback, you cannot use Huawei CSI to create new resources or mount or unmount an existing PVC.
- During the upgrade or rollback, do not uninstall the snapshot-dependent component service.

Upgrading CSI from 2.x or 3.x to 4.2.0

To upgrade CSI from 2.*x* or 3.*x* to 4.2.0, perform the following operations:

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the **kubectl get cm huawei-csi-configmap -n** *huawei-csi* **-o json > configmap.json** command to back up the backend information to the **configmap.json** file. For the OpenShift platform, replace **kubectl** with **oc**.

```
# kubectl get cm huawei-csi-configmap -n huawei-csi -o json > configmap.json
```

Step 3 Uninstall CSI. For details, see **5.2 Manually Uninstalling Huawei CSI**.

Step 4 Install CSI of the current version. For details, see **4.2 Manually Installing Huawei CSI**.

Step 5 Install the backend information backed up in **Step 2** according to **7.1 Adding a Storage Backend**.

----End

Upgrading CSI from 4.x to 4.2.0

To upgrade CSI from 4.*x* to 4.2.0, perform the following operations:

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Uninstall CSI. For details, see **5.2 Manually Uninstalling Huawei CSI**.

Step 3 Install CSI of the current version. For details, see **4.2 Manually Installing Huawei CSI**.

----End

6.2.2 Rolling Back Huawei CSI

Uninstall CSI by referring to **5.2 Manually Uninstalling Huawei CSI**, and then download and install CSI of the source version.

CAUTION

- During the upgrade or rollback, the existing resources such as PVCs, snapshots, and Pods will run properly and will not affect your service access.
- During the upgrade or rollback, you cannot use Huawei CSI to create new resources or mount or unmount an existing PVC.
- During the upgrade or rollback, do not uninstall the snapshot-dependent component service.

Prerequisites

You have downloaded the CSI software package of the source version.

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Uninstall CSI. For details, see **5.2 Manually Uninstalling Huawei CSI**.

Step 3 Reinstall CSI of the source version. For details, see **4.2 Manually Installing Huawei CSI**.

----End

7 Storage Backend Management

Backend is an abstract concept of Huawei storage resources. Each Huawei storage device can abstract multiple backend resources using features such as tenants, storage pools, and protocols. Each backend exists independently and defines Huawei storage information required for providing persistent volumes for Kubernetes clusters.

This section describes how to use the oceanctl tool to manage storage backends, including creating, querying, updating, and deleting backends. For details about the help information of the oceanctl tool , run the **oceanctl --help** command.

Prerequisites

- You have obtained the oceanctl tool, copied the oceanctl tool to the environment directory, for example, **/usr/local/bin**, and obtained the execute permission. The oceanctl tool is stored in **/bin/oceanctl** of the software package.
- The oceanctl tool must be executed on a master node.
- **huawei-csi** is the default namespace used by oceanctl to create a backend.

7.1 Adding a Storage Backend

This section describes how to create a storage backend. Currently, you can create a backend based on the configured backend yaml file or the exported configmap.json file.

If you create a backend by adding a backend yaml file, configure the backend file by referring to **7.1.1 Preparing the Storage Backend Configuration File**.

If the exported configmap.json file exists, create a storage backend by referring to **7.1.2 Creating a Storage Backend**.

7.1.1 Preparing the Storage Backend Configuration File

NOTE

- Before configuring NAS HyperMetro, you need to configure the HyperMetro relationship between two storage devices, including the remote device, HyperMetro domain, and the like. The HyperMetro domain of the file system can only work in HyperMetro mode. For details about the configuration operation, see the product documentation of the corresponding storage model.
- The accounts for connecting to NAS HyperMetro backends must be the administrator accounts of the storage vStores.

Backend Configuration File Description

An example template of the backend configuration file is **/examples/backend/backend.yaml**. The following table lists the parameters.

Table 7-1 backend parameters

Parameter	Description	Mandatory	Default Value	Remarks
storage	Storage service type. ● If enterprise storage provides SAN, set this parameter to **oceanstor-san**. ● If enterprise storage provides NAS, set this parameter to **oceanstor-nas**. ● If enterprise storage provides NAS of the Dtree type, set this parameter to **oceanstor-dtree**. ● If distributed storage provides SAN, set this parameter to **fusionstorage-san**. ● If distributed storage provides NAS, set this parameter to **fusionstorage-nas**.	Yes	oceanstor-nas	One backend can provide only one storage service. If a single Huawei storage system can provide both SAN and NAS storage services, you can configure multiple backends and use different storage service types for each backend.
name	Storage backend name. The value can contain a maximum of 63 characters, including lowercase letters, digits, and hyphens (-). It must start with a letter or digit.	Yes	-	Ensure that the storage backend name is unique.
namespace	Namespace.	No	-	The storage backend must be in the same namespace as Huawei CSI.

Parameter	Description	Mandatory	Default Value	Remarks
vstoreName	vStore name on the storage side. This parameter needs to be specified when the connected backend is OceanStor V3/V5 and resources need to be provisioned under a specified vStore.	Conditionally mandatory	-	This parameter needs to be specified only when the backend is OceanStor V3/V5 and vStores need to be supported.
accountName	Account name on the storage side. This parameter is mandatory when OceanStor Pacific series NAS is connected and NAS resources need to be provisioned under a specified account.	Conditionally mandatory	-	This parameter needs to be specified only when the backend is OceanStor Pacific series NAS and accounts need to be supported.
urls	Management URLs of storage device. The value format is a list. The value can be a domain name or an IP address + port number. Only IPv4 addresses are supported.	Yes	-	If the connected backend is OceanStor V6 or OceanStor Dorado V6 and resources need to be provisioned under a specified vStore, set this parameter to the URL of the logical management port of the vStore.
pools	Storage pools of storage devices. The value format is a list.	Conditionally mandatory	-	This parameter is optional when **storage** is set to **oceanstor-dtree**.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.protocol	Storage protocol. The value is a character string. ● iscsi ● fc ● roce ● fc-nvme ● nfs ● dpc ● scsi	Yes	-	● If the value is set to **iscsi**, ensure that an iSCSI client has been installed on the connected compute node. ● If the value is set to **nfs**, ensure that an NFS client tool has been installed on the connected compute node. ● If the value is set to **fc-nvme** or **roce**, ensure that the nvme-cli tool of version 1.9 or later has been installed on the connected compute node. ● If the value is set to **dpc**, ensure that DPC has been installed on the connected compute node and the node has been added as a DPC compute node on the storage device to be connected. ● If the value is set to **scsi**, ensure that a distributed storage VBS client has been installed on the connected compute node.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.portals	Service access port. Nodes will use this port to read and write storage resources. The value format is a list. Multiple ports can be configured if the protocol is **iscsi** or **roce**. Only one port can be configured if the protocol is **nfs**. Service ports do not need to be configured if the protocol is **fc**, **fc-nvme**, or **dpc**. If the protocol is **scsi**, the port is in dictionary format where the key indicates the host name and the value indicates the IP address (only IPv4 addresses are supported).	Conditionally mandatory	-	• If a vStore or account is used to connect to a backend, **portals** must be set to the logical port information of the vStore or account. • If **nfs** is used, the value can be a domain name.
parameters.ALUA	ALUA configuration of the storage backend. If the worker node uses the native multipathing software provided by the OS and ALUA is enabled, you need to configure this parameter.	Conditionally mandatory	-	If ALUA is enabled for the host multipathing software, ensure that the backend ALUA configuration is the same as that of the host ALUA configuration. For details about the ALUA configuration, see **9.1.1 Configuring ALUA Using Helm**.
parameters.parentname	Name of a file system on the current storage device. Dtree is created in the file system. This parameter is mandatory when **storage** is set to **oceanstor-dtree**.	Conditionally mandatory	-	Query the name on the **File Systems** page of DeviceManager.

Parameter	Description	Mandatory	Default Value	Remarks
metrovStorePairID	HyperMetro vStore pair ID. This parameter is mandatory when a PV to be created on the storage side needs to support the NAS HyperMetro feature. In this case, you need to enter the ID of the HyperMetro vStore pair to which the PV to be created belongs.	Conditionally mandatory	-	You can query the HyperMetro vStore pair ID on DeviceManager.
metroBackend	Backend name of the HyperMetro peer. The value is a character string. This parameter is mandatory when a PV to be created on the storage side needs to support the NAS HyperMetro feature. In this case, you need to enter the name of the other backend to form a HyperMetro pair with the current backend.	Conditionally mandatory	-	The names of the two backends in the pair must be entered. After the two backends form a HyperMetro relationship, they cannot form a HyperMetro relationship with other backends.
supportedTopologies	Storage topology awareness configuration. The parameter format is JSON of the list type.	Conditionally mandatory	-	This parameter is mandatory if storage topology awareness is enabled. For details, see **9.2.1 Configuring Storage Topology Awareness Using Helm**.
maxClientThreads	Maximum number of concurrent connections to a storage backend.	No	30	If this parameter is not specified, the default maximum number of connections is 30.

For details about how to configure **backend** in typical scenarios, see the following examples:

Configuring a Storage Backend of the iSCSI Type

The following is an example of the backend configuration file of the iSCSI type for enterprise storage:

```
storage: "oceanstor-san"
name: "dorado-iscsi-155"
namespace: "huawei-csi"
urls:
  - "https://192.168.129.155:8088"
  - "https://192.168.129.156:8088"
pools:
  - "StoragePool001"
parameters:
  protocol: "iscsi"
  portals:
    - "192.168.128.120"
    - "192.168.128.121"
maxClientThreads: "30"
```

The following is an example of the backend configuration file of the iSCSI type for distributed storage:

```
storage: "fusionstorage-san"
name: "pacific-iscsi-125"
namespace: "huawei-csi"
urls:
  - "https://192.168.129.125:8088"
  - "https://192.168.129.126:8088"
pools:
  - "StoragePool001"
parameters:
  protocol: "iscsi"
  portals:
    - "192.168.128.122"
    - "192.168.128.123"
maxClientThreads: "30"
```

Configuring a Storage Backend of the FC Type

The following is an example of the backend configuration file of the FC type for enterprise storage:

```
storage: "oceanstor-san"
name: "fc-155"
namespace: "huawei-csi"
urls:
  - "https://192.168.129.155:8088"
  - "https://192.168.129.156:8088"
pools:
  - "StoragePool001"
```



```
parameters:
  protocol: "fc"
maxClientThreads: "30"
```

Configuring a Storage Backend of the NVMe over RoCE Type

The following is an example of the backend configuration file of the NVMe over RoCE type for enterprise storage:

```
storage: "oceanstor-san"
name: "roce-155"
namespace: "huawei-csi"
urls:
  - "https://192.168.129.155:8088"
  - "https://192.168.129.156:8088"
pools:
  - "StoragePool001"
parameters:
  protocol: "roce"
  portals:
    - "192.168.128.120"
    - "192.168.128.121"
maxClientThreads: "30"
```

Configuring a Storage Backend of the NVMe over FC Type

The following is an example of the backend configuration file of the NVMe over FC type for enterprise storage:

```
storage: "oceanstor-san"
name: "fc-nvme-155"
namespace: "huawei-csi"
urls:
  - "https://192.168.129.155:8088"
  - "https://192.168.129.156:8088"
pools:
  - "StoragePool001"
parameters:
  protocol: "fc-nvme"
maxClientThreads: "30"
```

Configuring a Storage Backend of the NFS Type

The following is an example of the backend configuration file of the NFS type for enterprise storage:

```
storage: "oceanstor-nas"
name: "nfs-155"
namespace: "huawei-csi"
urls:
  - "https://192.168.129.155:8088"
  - "https://192.168.129.156:8088"
pools:
  - "StoragePool001"
parameters:
  protocol: "nfs"
  portals:
    - "192.168.128.155"
maxClientThreads: "30"
```

The following is an example of the backend configuration file of the NFS type for distributed storage:

```
storage: "fusionstorage-nas"
name: "nfs-126"
```



```
namespace: "huawei-csi"
urls:
  - "https://192.168.129.125:8088"
  - "https://192.168.129.126:8088"
pools:
  - "StoragePool001"
parameters:
  protocol: "nfs"
  portals:
    - "192.168.128.123"
maxClientThreads: "30"
```

Configuring a Storage Backend of the SCSI Type

The following is an example of the backend configuration file of the SCSI type for distributed storage:

```
storage: "fusionstorage-san"
name: "scsi-155"
namespace: "huawei-csi"
urls:
  - "https://192.168.129.155:8088"
pools:
  - "StoragePool001"
parameters:
  protocol: "scsi"
  portals:
    - {"hostname01": "192.168.125.21"}
    - {"hostname02": "192.168.125.22"}
maxClientThreads: "30"
```

Configuring a Storage Backend of the DPC Type

The following is an example of the backend configuration file of the DPC type for distributed storage:

```
storage: "fusionstorage-nas"
name: "dpc-155"
namespace: "huawei-csi"
urls:
  - "https://192.168.129.155:8088"
  - "https://192.168.129.156:8088"
pools:
  - "StoragePool001"
parameters:
  protocol: "dpc"
maxClientThreads: "30"
```

Configuring Storage Backends of the Dtree Type

The following is an example of the backend configuration file of the Dtree type for enterprise storage:

```
storage: "oceanstor-dtree"
name: "nfs-dtree"
namespace: "huawei-csi"
urls:
- "https://192.168.129.155:8088"
parameters:
  protocol: "nfs"
  parentname: "parent-filesystem"
  portals:
    - "192.168.128.155"
maxClientThreads: "30"
```


Configuring Storage Backends of the HyperMetro Type

CSI allows you to provision HyperMetro volumes of the NFS type on the storage side when connecting with OceanStor V6 or OceanStor Dorado V6. When configuring storage backends that work in HyperMetro mode, you need to create two configuration files and create backends one by one.

This example shows how to configure backends of the HyperMetro type for Huawei OceanStor V6 or OceanStor Dorado V6. First, create local storage backend configuration file **nfs-hypermetro-155.yaml**.

```
storage: "oceanstor-nas"
name: "nfs-hypermetro-155"
namespace: "huawei-csi"
urls:
  - "https://192.168.129.155:8088"
  - "https://192.168.129.156:8088"
pools:
  - "StoragePool001"
metrovStorePairID: "f09838237b93c000"
metroBackend: "nfs-hypermetro-157"
parameters:
  protocol: "nfs"
  portals:
    - "192.168.129.155"
maxClientThreads: "30"
```

After the local backend is created, create remote storage backend configuration file **nfs-hypermetro-157.yaml**.

```
storage: "oceanstor-nas"
name: "nfs-hypermetro-157"
namespace: "huawei-csi"
urls:
  - "https://192.168.129.157:8088"
  - "https://192.168.129.158:8088"
pools:
  - "StoragePool001"
metrovStorePairID: "f09838237b93c000"
metroBackend: "nfs-hypermetro-155"
parameters:
  protocol: "nfs"
  portals:
    - "192.168.129.157"
maxClientThreads: "30"
```


7.1.2 Creating a Storage Backend

NOTE

1. When oceanctl is used to create a storage backend, the entered account and key information is stored in the **Secret** object. It is recommended that the customer container platform encrypt the Secret object based on the suggestions of the supplier or K8s community. For details about how to encrypt the Secret object in the K8s community, see **Enable Encryption at Rest**.
2. When a backend is created using a .json file, the backend name of an earlier version may contain uppercase letters or underscores (_). In this case, the old name is remapped to a new name. The mapping process automatically occurs and does not affect the original functions. For example, **ABC_123** is mapped to **abc-123-fd68e**. The mapping rules are as follows:
   - Uppercase letters are converted to lowercase letters.
   - An underscore (_) is converted to a hyphen (-).
   - A 5-digit hash code is added to the end.

Command Description

- Run the following command to obtain the help information about creating a backend.
  ```
  oceanctl create backend -h
  ```
- Run the following command to create a storage backend based on the specified yaml file.
  ```
  oceanctl create backend -f /path/to/backend.yaml -i yaml
  ```
- Run the following command to create a storage backend based on the specified json file. The **huawei-csi-configmap** file can be exported only in json format.
  ```
  oceanctl create backend -f /path/to/configmap.json -i json
  ```
- Run the following command to create a storage backend in the specified namespace.
  ```
  oceanctl create backend -f /path/to/backend.yaml -i yaml -n <namespace>
  ```
- Run the following command to create a storage backend and ignore the storage backend name verification, for example, uppercase letters and hyphens (-).
  ```
  oceanctl create backend -f /path/to/backend.yaml -i yaml --not-validate-name
  ```
- Run the following command to create a storage backend and specify **provisioner**. **csi.oceanstor.com** is the driver name specified during installation. For details, see **Step 4**.

  NOTE

  This command is used only when a backend is created on the CCE/CCE Agile platform.

  ```
  oceanctl create backend -f /path/to/backend.yaml  -i yaml --provisioner=csi.oceanstor.com
  ```

Example of Creating a Backend

Step 1 Prepare the backend configuration file, for example, **backend.yaml**. For details, see **7.1.1 Preparing the Storage Backend Configuration File**. To create multiple backends, separate them with **---**.

```
storage: "oceanstor-san"
name: "backend-1"
namespace: "huawei-csi"
urls:
```



```
  - "https://192.168.129.157:8088"
pools:
  - "StoragePool001"
parameters:
  protocol: "roce"
  portals:
    - "10.10.30.20"
    - "10.10.30.21"
maxClientThreads: "30"
---
storage: "oceanstor-san"
name: "backend-2"
namespace: "huawei-csi"
urls:
  - "https://192.168.129.158:8088"
pools:
  - "StoragePool001"
parameters:
  protocol: "roce"
  portals:
    - "10.10.30.20"
    - "10.10.30.21"
maxClientThreads: "30"
```

Step 2 Run the following command to create a storage backend.

```
# oceanctl create backend -f /path/to/backend.yaml -i yaml
+--------+------------+-----------+---------------+-----------------------------+
| NUMBER | CONFIGURED |   NAME    |    STORAGE    |          URLS               |
+--------+------------+-----------+---------------+-----------------------------+
|   1    |   false    | backend-2 | oceanstor-san | https://192.168.129.157:8088 |
+--------+------------+-----------+---------------+-----------------------------+
|   2    |   false    | backend-1 | oceanstor-san | https://192.168.129.158:8088 |
+--------+------------+-----------+---------------+-----------------------------+
Please enter the backend number to configure (Enter 'exit' to exit):
```

Step 3 Enter the serial number of the backend to be created and enter the account and password.

```
Please enter the backend number to configure (Enter 'exit' to exit):1
Please enter this backend user name:admin
Please enter this backend password:

Backend backend-1 is configured
+--------+------------+-----------+-----------------+-----------------------------+
| NUMBER | CONFIGURED |   NAME    |     STORAGE     |          URLS               |
+--------+------------+-----------+-----------------+-----------------------------+
|   1    |   true     | backend-1 | oceanstor-dtree | https://192.168.129.157:8088 |
+--------+------------+-----------+-----------------+-----------------------------+
|   2    |   false    | backend-2 | oceanstor-dtree | https://192.168.129.158:8088 |
+--------+------------+-----------+-----------------+-----------------------------+
Please enter the backend number to configure (Enter 'exit' to exit):
```

Step 4 Check the storage backend creation result.

```
# oceanctl get backend
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
| NAMESPACE  |   NAME    | PROTOCOL |   STORAGETYPE   |          SN          | STATUS | ONLINE |
URL                |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
| huawei-csi | backend-1 |   roce   | oceanstor-san   | xxxxxxxxxxxxxxxxxxx | Bound  |  true  | https://
192.168.129.157:8088   |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
| huawei-csi | backend-2 |   roce   | oceanstor-san   | xxxxxxxxxxxxxxxxxxx | Bound  |  true  | https://
192.168.129.158:8088   |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
```

----End

7.2 Querying a Storage Backend

Command Description

- Run the following command to obtain the help information about querying a backend.
  ```
  oceanctl get backend -h
  ```
- Run the following command to query a single storage backend in the default namespace.
  ```
  oceanctl get backend <backend-name>
  ```
- Run the following command to query all storage backends in the specified namespace.
  ```
  oceanctl get backend -n <namespace>
  ```
- Run the following command to format the output. Currently, **json**, **yaml**, and **wide** are supported.
  ```
  oceanctl get backend <backend-name> -o json
  ```

7.3 Updating a Storage Backend

NOTICE

- Currently, only the password in the storage backend information can be updated.
- If the backend account password is updated on the storage device, the CSI plug-in will retry due to login failures. As a result, the account may be locked. If the account is locked, change the password by referring to **11.20 An Account Is Locked After the Password Is Updated on the Storage Device**.

Command Description

- Run the following command to obtain the help information about updating a backend.
  ```
  oceanctl update backend -h
  ```
- Run the following command to update the specified storage backend in the default namespace.
  ```
  oceanctl update backend <backend-name> --password
  ```
- Run the following command to update a storage backend in the specified namespace.
  ```
  oceanctl update backend <backend-name> -n <namespace> --password
  ```

Example of Updating a Backend

Step 1 Run the following command to obtain the help information about updating a storage backend.

```
# oceanctl update backend -h
Update a backend for Ocean Storage in Kubernetes

Usage:
  oceanctl update backend <name> [flags]
```



```
Examples:
  # Update backend account information in default(huawei-csi) namespace
  oceanctl update backend <name>  --password

  # Update backend account information in specified namespace
  oceanctl update backend <name> -n namespace --password

Flags:
  -h, --help               help for backend
  -n, --namespace string   namespace of resources
      --password           Update account password
```

Step 2 Run the following command to update a storage backend.

```
# oceanctl update backend backend-1 --password
Please enter this backend user name:admin
Please enter this backend password:

backend/backend-1 updated
```

----End

7.4 Deleting a Storage Backend

Command Description

- Run the following command to obtain the help information about deleting a backend.
  ```
  oceanctl delete backend -h
  ```
- Run the following command to delete the specified storage backend in the default namespace.
  ```
  oceanctl delete backend <backend-name>
  ```
- Run the following command to delete all storage backends in the default namespace.
  ```
  oceanctl delete backend  --all
  ```
- Run the following command to delete a storage backend in the specified namespace.
  ```
  oceanctl delete backend <backend-name...> -n <namespace>
  ```

Example of Deleting a Backend

Step 1 Run the following command to obtain information about a storage backend.

```
# oceanctl get backend
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
| NAMESPACE  |   NAME    | PROTOCOL |   STORAGETYPE   |          SN          | STATUS | ONLINE |
URL                |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
| huawei-csi | backend-1 |   roce   | oceanstor-san   | xxxxxxxxxxxxxxxxxxx | Bound  |  true  | https://
192.168.129.157:8088   |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
| huawei-csi | backend-2 |   roce   | oceanstor-san   | xxxxxxxxxxxxxxxxxxx | Bound  |  true  | https://
192.168.129.158:8088   |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
```

Step 2 Run the following command to delete the specified storage backend.

```
# oceanctl delete backend backend-1
backend/backend-1 deleted
```


Step 3 Check the deletion result. If **not found** is displayed, the deletion is successful.

```
# oceanctl get backend backend-1
Error from server (NotFound): backend "backend-1" not found
```

----End

7.5 (Optional) Adding a Certificate to a Storage Backend

This section describes how to create a certificate for a storage backend. If certificate verification is required for logging in to the storage, you can add a certificate by referring to this section. Currently, you can create a certificate for a storage backend based on the specified .crt or .pem file.

NOTICE

Before creating a certificate for a storage backend, import the prepared certificate to the storage array.

7.5.1 Creating a Certificate for a Storage Backend

Prerequisites

A certificate has been created. Take OceanStor Dorado V6 as an example. For details about how to create a certificate, **click here**.

Command Description

- Run the following command to obtain the help information about querying a certificate.
  ```
  oceanctl create cert -h
  ```
- Run the following command to create a certificate for a single storage backend in the default namespace based on the specified .crt certificate file.
  ```
  oceanctl create cert <name> -f /path/to/cert.crt -b <backend-name>
  ```
- Run the following command to create a certificate for a single storage backend in the specified namespace based on the specified .crt certificate file.
  ```
  oceanctl create cert <name> -f /path/to/cert.crt -b <backend-name> -n <namespace>
  ```
- Run the following command to create a certificate for a single storage backend in the specified namespace based on the specified .pem certificate file.
  ```
  oceanctl create cert <name> -f /path/to/cert.pem -b <backend-name> -n <namespace>
  ```

Example of Creating a Certificate

Step 1 Prepare a certificate file in advance, for example, **cert.crt**.

Step 2 Run the following command to obtain information about a storage backend.

```
# oceanctl get backend
+-----------+-----------+----------+-----------------+----------------------+--------+--------
+-------------------------------+
| NAMESPACE |   NAME    | PROTOCOL |   STORAGETYPE   |          SN          | STATUS | ONLINE |
```



```
URL              |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+-------------------------------+
| huawei-csi | backend-1 |   roce   | oceanstor-san   | xxxxxxxxxxxxxxxxxxx | Bound  |  true  | https://
192.168.129.157:8088   |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+-------------------------------+
| huawei-csi | backend-2 |   roce   | oceanstor-san   | xxxxxxxxxxxxxxxxxxx | Bound  |  true  | https://
192.168.129.158:8088   |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+-------------------------------+
```

Step 3 Run the following command to create a certificate for the specified storage backend.

```
# oceanctl create cert cert-1 -b backend-1 -f /path/to/cert.crt
cert/cert-1 created
```

Step 4 Check the certificate creation result.

```
# oceanctl get cert -b backend-1
+------------+--------+--------------+
| NAMESPACE  |  NAME  | BOUNDBACKEND |
+------------+--------+--------------+
| huawei-csi | cert-1 |   backend-1  |
+------------+--------+--------------+
```

----End

7.6 (Optional) Querying a Storage Backend Certificate

Command Description

- Run the following command to obtain the help information about querying a certificate.
  ```
  oceanctl get cert -h
  ```
- Run the following command to query the certificate of a specified storage backend in the default namespace.
  ```
  oceanctl get cert -b <backend-name>
  ```
- Run the following command to query the certificate of a specified storage backend in the specified namespace.
  ```
  oceanctl get cert -b <backend-name> -n <namespace>
  ```

7.7 (Optional) Updating a Storage Backend Certificate

Before updating a certificate, prepare a new certificate file and update the storage backend certificate by following the instructions provided in this section. If the certificate is no longer used, delete the certificate from the storage backend by referring to **7.8 (Optional) Deleting a Storage Backend Certificate**.

Command Description

- Run the following command to obtain the help information about updating a certificate.
  ```
  oceanctl update cert -h
  ```
- Run the following command to update a certificate for a specified storage backend in the default namespace based on the specified .crt certificate file.
  ```
  oceanctl update cert -b <backend-name> -f /path/to/cert.crt
  ```


- Run the following command to update a certificate for a specified storage backend in the specified namespace based on the specified .crt certificate file.
  ```
  oceanctl update cert -b <backend-name> -n <namespace> -f /path/to/cert.crt
  ```
- Run the following command to update a certificate for a specified storage backend in the specified namespace based on the specified .pem certificate file.
  ```
  oceanctl update cert -b <backend-name> -n <namespace> -f /path/to/cert.pem
  ```

Example of Updating a Certificate

Step 1 Run the following command to obtain information about a storage backend.

```
# oceanctl get backend
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
| NAMESPACE  |   NAME    | PROTOCOL |   STORAGETYPE   |          SN          | STATUS | ONLINE |
URL                |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
| huawei-csi | backend-1 |   roce   | oceanstor-san   | xxxxxxxxxxxxxxxxxxxx | Bound  |  true  | https://
192.168.129.157:8088   |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
| huawei-csi | backend-2 |   roce   | oceanstor-san   | xxxxxxxxxxxxxxxxxxxx | Bound  |  true  | https://
192.168.129.158:8088   |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
```

Step 2 Run the following command to check whether the specified storage backend has a certificate.

```
# oceanctl get cert -b backend-1
+------------+--------+--------------+
| NAMESPACE  |  NAME  | BOUNDBACKEND |
+------------+--------+--------------+
| huawei-csi | cert-1 |   backend-1  |
+------------+--------+--------------+
```

Step 3 Run the following command to update the certificate of the specified storage backend.

```
# oceanctl update cert -b backend-1 -f /path/to/cert.crt
cert/cert-1 updated
```

----End

7.8 (Optional) Deleting a Storage Backend Certificate

Command Description

- Run the following command to obtain the help information about deleting a certificate.
  ```
  oceanctl delete cert -h
  ```
- Run the following command to delete the certificate of a specified storage backend in the default namespace.
  ```
  oceanctl delete cert -b <backend-name>
  ```
- Run the following command to delete the certificate of a specified storage backend in the specified namespace.
  ```
  oceanctl delete cert -b <backend-name> -n <namespace>
  ```


Example of Deleting a Certificate

Step 1 Run the following command to obtain information about a storage backend.

```
# oceanctl get backend
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
| NAMESPACE  |   NAME    | PROTOCOL |   STORAGETYPE   |          SN          | STATUS | ONLINE |
URL                |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
| huawei-csi | backend-1 |   roce   | oceanstor-san   | xxxxxxxxxxxxxxxxxxx | Bound  |  true  | https://
192.168.129.157:8088   |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
| huawei-csi | backend-2 |   roce   | oceanstor-san   | xxxxxxxxxxxxxxxxxxx | Bound  |  true  | https://
192.168.129.158:8088   |
+------------+-----------+----------+-----------------+----------------------+--------+--------
+--------------------------------+
```

Step 2 Run the following command to obtain information about the certificate of the specified storage backend.

```
# oceanctl get cert -b backend-1
+------------+--------+--------------+
| NAMESPACE  |  NAME  | BOUNDBACKEND |
+------------+--------+--------------+
| huawei-csi | cert-1 |   backend-1  |
+------------+--------+--------------+
```

Step 3 Run the following command to delete the certificate of the specified storage backend.

```
# oceanctl delete cert -b backend-1
cert/cert-1 deleted
```

Step 4 Check the deletion result. If **no cert found** is displayed, the deletion is successful.

```
# oceanctl get cert -b backend-1
Error from server (NotFound): no cert found on backend backend-1 in huawei-csi namespace
```

----End

8 Using Huawei CSI

This chapter describes how to use Huawei CSI to manage the lifecycle of PVs and snapshots.

NOTICE

When block volumes are mapped, Huawei CSI automatically creates associated objects, such as hosts, host groups, and LUN groups, as well as mapping views. If these objects are manually created on the storage, the mapping logic of Huawei CSI will be affected. Therefore, ensure that these objects are deleted before mapping volumes using Huawei CSI.

8.1 Managing a PV/PVC

Based on service requirements, files in containers need to be persistently stored on disks. When the containers are re-built or re-allocated to new nodes, the persistent data can still be used.

To persistently store data on storage devices, you need to use the **PersistentVolume (PV)** and **PersistentVolumeClaim (PVC)** when provisioning containers.

- PV: a piece of storage in the Kubernetes cluster that has been provisioned by an administrator or dynamically provisioned using a **StorageClass**.
- PVC: a request for storage by a user. A PVC consumes PV resources. A PVC can request specific size and access modes. For example, a PV can be mounted in ReadWriteOnce, ReadOnlyMany, or ReadWriteMany mode. For details, see **Access Modes**.

This section describes how to use Huawei CSI to create, expand the capacity of, and clone a PV/PVC, as well as create a PVC using a snapshot.

8.1.1 Creating a PVC

Huawei CSI allows storage resources (LUNs or file systems) to be created on Huawei storage and provided for containers based on user settings. For details about the supported features, see **Table 2-5** or **Table 2-9**.

A PVC can be created in dynamic volume provisioning or static volume provisioning mode.

- Dynamic volume provisioning does not require a PV to be created in advance. Huawei CSI automatically creates resources required by a PV on storage devices based on a StorageClass. In addition, you can create a PV when creating a PVC.
- Static volume provisioning requires the administrator to create required resources on a storage device in advance and use existing resources by creating a PV. In addition, you can specify the associated PV when creating a PVC.

8.1.1.1 Dynamic Volume Provisioning

Dynamic volume provisioning allows storage volumes to be created on demand. Dynamic volume provisioning depends on the StorageClass objects. The cluster administrator can define multiple StorageClass objects as required and specify a StorageClass that meets service requirements when declaring a PV or PVC. When applying for resources from Huawei storage devices, Huawei CSI creates storage resources that meet service requirements based on the preset StorageClass.

To implement dynamic volume provisioning, perform the following steps:

- Configuring a StorageClass
- Configuring a PVC

8.1.1.1.1 Configuring a StorageClass

A **StorageClass** provides administrators with methods to describe a storage "class". Different types may map to a different group of capability definitions. Kubernetes cluster users can dynamically provision volumes based on a StorageClass.

A StorageClass supports the following parameters.

If SAN storage is used, refer to example file **/examples/sc-lun.yaml**. If NAS storage is used, refer to example file **/examples/sc-fs.yaml**.

Table 8-1 StorageClass configuration parameters

Parameter	Description	Mandatory	Default Value	Remarks
metadata.name	User-defined name of a StorageClass object.	Yes	-	Take Kubernetes v1.22.1 as an example. The value can contain digits, lowercase letters, hyphens (-), and periods (.), and must start and end with a letter or digit.
provisioner	Name of the provisioner.	Yes	csi.huawei.com	Set this parameter to the driver name set during Huawei CSI installation. The value is the same as that of **driverName** in the **values.yaml** file.
reclaimPolicy	Reclamation policy. The following types are supported: • **Delete**: Resources are automatically reclaimed. • **Retain**: Resources are manually reclaimed.	No	Delete	• **Delete**: When a PV/PVC is deleted, resources on the storage device are also deleted. • **Retain**: When a PV/PVC is deleted, resources on the storage device are not deleted.
allowVolumeExpansion	Whether to allow volume expansion. If this parameter is set to **true**, the capacity of the PV that uses the StorageClass can be expanded.	No	false	This function can only be used to expand PV capacity but cannot be used to reduce PV capacity. The PV capacity expansion function is supported in Kubernetes 1.14 (alpha) and later versions.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.backend	Name of the backend where the resource to be created is located.	No	-	If this parameter is not set, Huawei CSI will randomly select a backend that meets the capacity requirements to create resources. You are advised to specify a backend to ensure that the created resource is located on the expected backend.
parameters.pool	Name of the storage resource pool where the resource to be created is located. If this parameter is set, **parameters.backend** must also be specified.	No	-	If this parameter is not set, Huawei CSI will randomly select a storage pool that meets the capacity requirements from the selected backend to create resources. You are advised to specify a storage pool to ensure that the created resource is located in the expected storage pool.
parameters.volumeType	Type of the volume to be created. The following types are supported: ● **lun**: A LUN is provisioned on the storage side. ● **fs**: A file system is provisioned on the storage side. ● **dtree**: A volume of the Dtree type is provisioned on the storage side.	Yes	-	● If NAS storage is used, this parameter must be set to **fs**. ● If SAN storage is used, this parameter must be set to **lun**. ● If NAS storage of the Dtree type is used, this parameter must be set to **dtree**.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.allocType	Allocation type of the volume to be created. The following types are supported: ● **thin**: Not all required space is allocated during creation. Instead, the space is dynamically allocated based on the usage. ● **thick**: All required space is allocated during creation.	No	-	If this parameter is not specified, **thin** will be used. Not all required space is allocated during creation. Instead, the space is dynamically allocated based on the usage. OceanStor Dorado V3/V6 does not support **thick**.
parameters.fsType	Type of a host file system. The supported types are: ● ext2 ● ext3 ● ext4 ● xfs	No	ext4	This parameter is valid only when **volumeType** of a StorageClass is set to **lun** and **volumeMode** of a PVC is set to **Filesystem**.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.authClient	IP address of the NFS client that can access the volume. This parameter is mandatory when **volumeType** is set to **fs**. You can enter the client host name (a full domain name is recommended), client IP address, or client IP address segment.	Conditionally mandatory	-	The asterisk (*) can be used to indicate any client. If you are not sure about the IP address of the access client, you are advised to use the asterisk (*) to prevent the client access from being rejected by the storage system. If the client host name is used, you are advised to use the full domain name. The IP addresses can be IPv4 addresses, IPv6 addresses, or a combination of IPv4 and IPv6 addresses. You can enter multiple host names, IP addresses, or IP address segments and separate them with semicolons (;) or spaces or by pressing **Enter**. Example: **192.168.0.10;192.168.0.0/24;myserver1.test**
parameters.cloneSpeed	Cloning speed. The value ranges from 1 to 4.	No	3	**4** indicates the highest speed. This parameter is available when you clone a PVC or create a PVC using a snapshot. For details, see **8.1.3 Cloning a PVC** or **8.1.4 Creating a PVC Using a Snapshot**.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.applicationType	Application type name for creating a LUN or NAS when the backend is OceanStor Dorado V6.	No	-	• If the value of **volumeType** is **lun**, log in to DeviceManager and choose **Services** > **Block Service** > **LUN Groups** > **LUNs** > **Create** to obtain the application type name. • If the value of **volumeType** is **fs**, log in to DeviceManager and choose **Services** > **File Service** > **File Systems** > **Create** to obtain the application type name.
parameters.qos	LUN/NAS QoS settings of the PV on the storage side. The value of the parameter is JSON character strings in dictionary format. A character string is enclosed by single quotation marks and the dictionary key by double quotation marks. Example: **'{"maxMBPS": 999, "maxIOPS": 999}'**	No	-	For details about the supported QoS configurations, see **Table 8-2**.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.storageQuota	Quota of a PV on the storage device. This parameter is valid only when NAS is used for connecting to OceanStor Pacific series storage. The value of the parameter is JSON character strings in dictionary format. A character string is enclosed by single quotation marks and the dictionary key by double quotation marks. Example: **'{"spaceQuota": "softQuota", "gracePeriod": 100}'**	No	-	For details about the supported quota configurations, see **Table 8-3**.
parameters.hyperMetro	Whether a HyperMetro volume is to be created. This parameter needs to be configured when the backend is of the HyperMetro type. • **"true"**: The created volume is a HyperMetro volume. • **"false"**: The created volume is a common volume.	Conditionally mandatory	false	Set this parameter when the used backend is a HyperMetro backend and a HyperMetro volume needs to be provisioned.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.fsPermission	Permission on the directory mounted to a container.	No	-	For details about the configuration format, refer to the Linux permission settings, for example, 777 and 755. All SAN storage devices are supported. Only the following NAS storage devices are supported: OceanStor Dorado V6, OceanStor V6, and OceanStor Pacific series 8.1.2 and later versions.
parameters.rootSquash	Controls the **root** permission of the client. The value can be: • **root_squash**: The client cannot access the storage system as user **root**. If a client accesses the storage system as user **root**, the client will be mapped as an anonymous user. • **no_root_squash** : A client can access the storage system as user **root** and has the permission of user **root**.	No	-	Only NAS storage is supported.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.allSquash	Whether to retain the user ID (UID) and group ID (GID) of a shared directory. The value can be: ● **all_squash**: The UID and GID of the shared directory are mapped to anonymous users. ● **no_all_squash**: The UID and GID of the shared directory are retained.	No	-	Only NAS storage is supported.
parameters.snapshotDirectoryVisibility	Whether the snapshot directory is visible. The value can be: ● **visible**: The snapshot directory is visible. ● **invisible**: The snapshot directory is invisible.	No	-	Only NAS storage is supported.
parameters.reservedSnapshotSpaceRatio	Configures reserved snapshot space. Value type: character string Value range: 0 to 50	No	-	OceanStor Dorado 6.1.5+ and OceanStor 6.1.5+ NAS storage devices are supported.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.description	Configures the description of the created file system or LUN. Value type: character string The value contains 0 to 255 characters.	No	-	Only enterprise storage file systems and LUNs are supported.
mountOptions.nfsvers	NFS mount option on the host. The following mount option is supported: **nfsvers**: protocol version for NFS mounting. The value can be **3**, **4**, **4.0**, or **4.1**.	No	-	This parameter is optional after the **-o** parameter when the **mount** command is executed on the host. The value is in list format. If the NFS version is specified for mounting, NFS 3, 4.0, and 4.1 protocols are supported (the protocol must be supported and enabled on storage devices). If **nfsvers** is set to **4**, the latest protocol version NFS 4 may be used for mounting due to different OS configurations, for example, 4.1. If protocol 4.0 is required, you are advised to set **nfsvers** to **4.0**.

Parameter	Description	Mandatory	Default Value	Remarks
mountOptions.acl	The DPC namespace supports the ACL function. The DPC client supports POSIX ACL, NFSv4 ACL, and NT ACL authentication.	No	-	The descriptions of **acl**, **aclonlyposix**, **cnflush**, and **cflush** are for reference only. For details about the parameters, see *OceanStor Pacific Series Product Documentation* and choose **Configuration** > **Basic Service Configuration Guide for File** > **Configuring Basic Services (DPC Scenario)** > **Accessing a DPC Share on a Client** > **Step 2**.
mountOptions.aclonlyposix	The DPC namespace supports POSIX ACL, and the DPC client supports POSIX ACL authentication. The following protocols support POSIX ACL: DPC, NFSv3, and HDFS. If NFSv4 ACL or NT ACL is used, the DPC client cannot identify the ACL of this type. As a result, the ACL of this type does not take effect.	No	-	If **aclonlyposix** and **acl** are used together, only **acl** takes effect. That is, the namespace supports the ACL function.

Parameter	Description	Mandatory	Default Value	Remarks
mountOptions.cnflush	Asynchronous disk flushing mode. That is, data is not flushed to disks immediately when files in the namespace are closed.	No	-	Asynchronous flushing mode: When a file is closed, data in the cache is not flushed to storage media in synchronous mode. Instead, data is written from the cache to the storage media in asynchronous flushing mode. After the write service is complete, data is flushed from the cache to disks periodically based on the flushing period. In a multi-client scenario, if concurrent operations are performed on the same file, the file size update is affected by the disk flushing period. That is, the file size is updated only after the disk flushing is complete. Generally, the update is completed within several seconds. Synchronous I/Os are not affected by the disk flushing period.
mountOptions.cflush	Synchronous disk flushing mode. That is, data is flushed to disks immediately when files in the namespace are closed.	No	-	By default, the synchronous disk flushing mode is used.

Table 8-2 Supported QoS configurations

Storage Type	Parameter	Description	Remarks
OceanStor V3/ OceanStor V5	IOTYPE	Read/write type.	This parameter is optional. If it is not specified, the default value of the storage backend is used. For details, see related storage documents. The value can be: • **0**: read I/O • **1**: write I/O • **2**: read and write I/Os
	MAXBANDWIDTH	Maximum bandwidth. This is a restriction policy parameter.	The value is an integer greater than 0, expressed in MB/s.
	MINBANDWIDTH	Minimum bandwidth. This is a protection policy parameter.	The value is an integer greater than 0, expressed in MB/s.
	MAXIOPS	Maximum IOPS. This is a restriction policy parameter.	The value is an integer greater than 0.
	MINIOPS	Minimum IOPS. This is a protection policy parameter.	The value is an integer greater than 0.
	LATENCY	Maximum latency. This is a protection policy parameter.	The value is an integer greater than 0, expressed in ms.
OceanStor Dorado V3	IOTYPE	Read/write type.	The value can be: • **2**: read and write I/Os
	MAXBANDWIDTH	Maximum bandwidth. This is a restriction policy parameter.	The value is an integer ranging from 1 to 999999999, expressed in MB/s.
	MAXIOPS	Maximum IOPS. This is a restriction policy parameter.	The value is an integer ranging from 100 to 999999999.

Storage Type	Parameter	Description	Remarks
OceanStor Dorado V6/ OceanStor V6	IOTYPE	Read/write type.	The value can be: • **2**: read and write I/Os
	MAXBANDWIDTH	Maximum bandwidth. This is a restriction policy parameter.	The value is an integer ranging from 1 to 999999999, expressed in MB/s.
	MINBANDWIDTH	Minimum bandwidth. This is a protection policy parameter.	The value is an integer ranging from 1 to 999999999, expressed in MB/s.
	MAXIOPS	Maximum IOPS. This is a restriction policy parameter.	The value is an integer ranging from 100 to 999999999.
	MINIOPS	Minimum IOPS. This is a protection policy parameter.	The value is an integer ranging from 100 to 999999999.
	LATENCY	Maximum latency. This is a protection policy parameter.	The value can be **0.5** or **1.5**, expressed in ms.
FusionStorage/ OceanStor Pacific series	maxMBPS	Maximum bandwidth. This is a restriction policy parameter.	This parameter is mandatory. The value is an integer greater than 0, expressed in MB/s. For details about the maximum value, see the actual limit of the storage device. For example, the maximum value of OceanStor Pacific NAS is 1073741824.
	maxIOPS	Maximum IOPS. This is a restriction policy parameter.	This parameter is mandatory. The value is an integer greater than 0. For details about the maximum value, see the actual limit of the storage device. For example, the maximum value of OceanStor Pacific NAS is 1073741824000.

NOTE

- vStore users of OceanStor V3/OceanStor V5 cannot configure QoS policies.
- The QoS configuration takes effect only on the newly created PVC. QoS cannot be added automatically for PVCs with the same StorageClass name that have been provisioned.

Table 8-3 Supported quota configurations

Parameter	Description	Remarks
spaceQuota	File quota type.	This parameter is mandatory. Only **softQuota** or **hardQuota** can be configured.
gracePeriod	Grace period allowed when the soft quota is configured.	This parameter is conditionally optional only when **spaceQuota** is set to **softQuota**. The value is an integer ranging from 0 to 4294967294.

For details about how to configure a StorageClass in typical scenarios, see the following examples:

- **Setting the Backend and Storage Pool in a StorageClass**
- **Setting the NFS Access Mode in a StorageClass**
- **Setting a Dtree Type in a StorageClass**
- **Setting the Local File System Access Mode in a StorageClass**
- **Setting the DPC Access Mode in a StorageClass**
- **Setting an Application Type in a StorageClass**
- **Setting a Soft Quota in a StorageClass**
- **Setting HyperMetro in a StorageClass**
- **Setting the Permission on a Mount Directory in a StorageClass**
- **Setting QoS in a StorageClass**
- **Configuring a StorageClass on the CCE/CCE Agile Platform**

Setting the Backend and Storage Pool in a StorageClass

If multiple Huawei backends are configured in a Kubernetes cluster or a Huawei backend provides multiple storage pools, you are advised to configure the specified backend and storage pool information in the StorageClass. This prevents Huawei CSI from randomly selecting backends and storage pools and ensures that the storage device where the volume resides complies with the plan.

For details about how to set the backend and storage pool for SAN storage, see the following configuration example.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
```



```
  name: mysc
provisioner: csi.huawei.com
allowVolumeExpansion: true
parameters:
  backend: "iscsi_san_181"
  pool: "pool001"
  volumeType: lun
  allocType: thin
```

For details about how to set the backend and storage pool for NAS storage, see the following configuration example.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
allowVolumeExpansion: true
parameters:
  backend: "iscsi_nas_181"
  pool: "pool001"
  volumeType: fs
  allocType: thin
  authClient: "*"
```

Setting the NFS Access Mode in a StorageClass

When a container uses an NFS file system as a storage resource, refer to the following configuration example. In this example, NFS version 4.1 is specified for mounting.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: nfs_nas_181
  pool: pool001
  volumeType: fs
  allocType: thin
  authClient: "192.168.0.10;192.168.0.0/24;myserver1.test"  #use * for all client
mountOptions:
  - nfsvers=4.1
```

Setting a Dtree Type in a StorageClass

When a container uses a Dtree as a storage resource, refer to the following configuration example.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: nfs_dtree
  volumeType: dtree
  allocType: thin
  authClient: "*"
mountOptions:
  - nfsvers=4.1
```


Setting the Local File System Access Mode in a StorageClass

If a container uses a LUN of enterprise storage or distributed storage as a storage resource and a file system needs to be formatted as a local file system, refer to the following example.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: iscsi_lun_181
  pool: pool001
  volumeType: lun
  allocType: thin
  fsType: xfs
```

Setting the DPC Access Mode in a StorageClass

If a container uses OceanStor Pacific series storage and the storage supports DPC-based access, you can configure mounting parameters for DPC-based access in the StorageClass. In this example, **acl** is used as the authentication parameter for mounting, and **cnflush** is used to set the asynchronous disk flushing mode.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: nfs_dpc_101
  pool: pool001
  volumeType: fs
  allocType: thin
  authClient: "*"  #use * for all client
mountOptions:
  - acl
  - cnflush
```

Setting an Application Type in a StorageClass

When a container uses a LUN of OceanStor Dorado V6 as the storage, if the default application type of the storage cannot meet the I/O model requirements of some services (for example, the container provides the database OLAP service), you can configure an application type in the StorageClass to improve storage performance. For details about the application types to be used, see the product documentation of the corresponding storage product.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: iscsi_lun_181
  pool: pool001
  volumeType: lun
  allocType: thin
  fsType: xfs
  applicationType: Oracle_OLAP
```


Setting a Soft Quota in a StorageClass

If a container uses a file system of OceanStor Pacific series as the storage, you can configure a soft quota in the StorageClass. The following is a configuration example.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: nfs_pacific_101
  pool: pool001
  volumeType: fs
  allocType: thin
  authClient: "*"
  storageQuota: '{"spaceQuota": "softQuota", "gracePeriod": 100}'
mountOptions:
  - nfsvers=3
```

Setting QoS in a StorageClass

When containers use enterprise storage or distributed storage as storage resources, you can set QoS for the storage resources used by containers to ensure that the storage read and write operations of these containers meet certain service levels.

Storage devices of different models or versions support different QoS settings. For details about how to find the configuration items of the corresponding storage devices, see **Table 8-2**. In this example, the backend is OceanStor Dorado V6. For other storage devices, refer to this example.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: iscsi_qos_181
  pool: pool001
  volumeType: lun
  allocType: thin
  fsType: xfs
  qos: '{"IOTYPE": 2, "MINIOPS": 1000}'
```

Setting HyperMetro in a StorageClass

When a container uses an NFS HyperMetro file system as a storage resource, refer to the following configuration example. In this example, the used backend supports HyperMetro, and **hyperMetro** is set to **true**.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: nfs_hypermetro_dorado_181
  pool: pool001
  volumeType: fs
  hyperMetro: "true"
  allocType: thin
  authClient: "*"
```


NOTICE

Before provisioning a NAS HyperMetro volume, you need to configure the HyperMetro relationship between two storage devices, including the remote device, HyperMetro domain, and the like. The HyperMetro domain of the file system can only work in HyperMetro mode. For details about the configuration operation, see the product documentation of the corresponding storage model.

Setting the Permission on a Mount Directory in a StorageClass

To modify the permission on a mount directory in a container, you can configure the directory permission in a StorageClass. The following is a configuration example.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
allowVolumeExpansion: true
parameters:
  volumeType: fs
  allocType: thin
  authClient: "*"
  fsPermission: "777"
  rootSquash: "no_root_squash"  # This parameter supports only NAS storage.
  allSquash: "no_all_squash"  # This parameter supports only NAS storage.
```

After the StorageClass configuration is complete, perform the following steps to create a StorageClass.

Step 1 Run the following command to create a StorageClass based on the .yaml file.

```
# kubectl create -f mysc.yaml
storageclass.storage.k8s.io/mysc created
```

Step 2 Run the following command to view the information about the created StorageClass.

```
# kubectl get sc
NAME   PROVISIONER      RECLAIMPOLICY   VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
mysc   csi.huawei.com   Delete          Immediate           false                  34s
```

After creating a StorageClass, you can use the StorageClass to create a PV or PVC.

----End

CAUTION

Pay attention to the following when using a StorageClass:

- Do not delete a StorageClass that is being used by a PV. Otherwise, the StorageClass information of the PV will be missing. As a result, an error occurs during mounting on the host.
- Modifications to a StorageClass do not take effect on existing PVs. You need to delete these PVs and create them again using the modified StorageClass to apply the modified parameters.

Configuring a StorageClass on the CCE/CCE Agile Platform

Create a StorageClass of the NAS type on the CCE/CCE Agile platform. The following is a configuration example. The value of **provisioner** must be the same as that of **driverName** in the **values.yaml** file.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
  annotations:
    storageclass.kubernetes.io/storageType: file
provisioner: csi.oceanstor.com
allowVolumeExpansion: true
parameters:
  volumeType: fs
  allocType: thin
  authClient: "*"
```

Create a StorageClass of the Block type on the CCE/CCE Agile platform. The following is a configuration example. The value of **provisioner** must be the same as that of **driverName** in the **values.yaml** file.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
  annotations:
    storageclass.kubernetes.io/storageType: block
provisioner: csi.oceanstor.com
allowVolumeExpansion: true
parameters:
  volumeType: lun
  allocType: thin
```

8.1.1.1.2 Configuring a PVC

After configuring a StorageClass, you can use the StorageClass to configure a PVC. For details about the PVC configuration template, see example file **pvc*.yaml** in the **examples** directory in Huawei CSI software package.

Table 8-4 Parameters in the pvc*.yaml file

Parameter	Description	Mandatory	Default Value	Remarks
metadata.name	User-defined name of a PVC object.	Yes	-	Take Kubernetes v1.22.1 as an example. The value can contain digits, lowercase letters, hyphens (-), and periods (.), and must start and end with a letter or digit.

Parameter	Description	Mandatory	Default Value	Remarks
spec.volumeMode	Volume mode. This parameter is optional. When LUN volumes are used, the following types are supported: • **Filesystem**: local file system. • **Block**: raw device.	No	Filesystem	This parameter takes effect when a PV is mounted. The default value is **Filesystem**. • **Filesystem** indicates that a container accesses a PV using a local file system. The local file system type is specified by the **fsType** field in the specified StorageClass. Storage of the Dtree type also uses this parameter. • **Block** indicates that a PV is accessed in raw volume mode.
spec.storageClassName	Name of the StorageClass object.	Yes	-	Name of the StorageClass object required by services.
spec.resources.requests.storage	Size of the volume to be created. The format is ***Gi and the unit is GiB. The size must be an integer multiple of 512 bytes.	Yes	10Gi	The PVC capacity depends on storage specifications and host specifications. For example, OceanStor Dorado 6.1.2 or OceanStor Pacific series 8.1.0 is connected to CentOS 7. If ext4 file systems are used, see **Table 8-5**. If XFS file systems are used, see **Table 8-6**. If NFS or raw devices are used, the capacity must meet the specifications of the used Huawei storage device model and version. If the PVC capacity does not meet the specifications, a PVC or Pod may fail to be created due to the limitations of storage specifications or host file system specifications.

Parameter	Description	Mandatory	Default Value	Remarks
spec.accessModes	Access mode of the volume. • **RWO** (ReadWriteOnce): A volume can be mounted to a node in read/write mode. This mode also allows multiple Pods running on the same node to access the volume. • **ROX** (ReadOnlyMany): A volume can be mounted to multiple nodes in read-only mode. • **RWX** (ReadWriteMany): A volume can be mounted to multiple nodes in read/write mode. • **RWOP** (ReadWriteOncePod): A volume can only be mounted to a single Pod in read/write mode. Kubernetes 1.22 and later versions support this feature.	Yes	ReadWriteOnce	• RWO/ROX/RWOP: supported by all types of volumes. RWOP is supported only by Kubernetes 1.22 and later versions. Check whether this feature is enabled for your Kubernetes cluster by referring to **10.5 Enabling the ReadWriteOncePod Feature Gate**. • RWX: supported by volumes whose **volumeMode** is set to **Block** or **NFS**.

Table 8-5 ext4 capacity specifications

Storage Type	Storage Specifications	ext4 Specifications	CSI Specifications
OceanStor Dorado 6.1.2	512 Ki to 256 Ti	50 Ti	512 Ki to 50 Ti
OceanStor Pacific series 8.1.0	64 Mi to 512 Ti	50 Ti	64 Mi to 50 Ti

Table 8-6 XFS capacity specifications

Storage Type	Storage Specifications	XFS Specifications	CSI Specifications
OceanStor Dorado 6.1.2	512 Ki to 256 Ti	500 Ti	512 Ki to 500 Ti
OceanStor Pacific series 8.1.0	64 Mi to 512 Ti	500 Ti	64 Mi to 500 Ti

Step 1 Based on service requirements, modify specific parameters by referring to the description in this section and the PVC configuration file example to generate the PVC configuration file to be created, for example, the **mypvc.yaml** file in this example.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: mypvc
spec:
  accessModes:
    - ReadWriteOnce
  volumeMode: Filesystem
  storageClassName: mysc
  resources:
    requests:
      storage: 100Gi
```

Step 2 Run the following command to create a PVC using the configuration file.

```
# kubectl create -f mypvc.yaml
persistentvolumeclaim/mypvc created
```

Step 3 After a period of time, run the following command to view the information about the created PVC. If the PVC status is **Bound**, the PVC has been created and can be used by a Pod.

```
# kubectl get pvc mypvc
NAME      STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
mypvc     Bound    pvc-840054d3-1d5b-4153-b73f-826f980abf9e   100Gi      RWO            mysc           12s
```


CAUTION

- After the PVC is created, if the PVC is in the **Pending** state after a long time (for example, one minute), refer to **11.6 When a PVC Is Created, the PVC Is in the Pending State**.
- You are advised to create or delete a maximum of 100 PVCs in a batch.

----End

After a PVC is created, you can use the PVC to create a Pod. The following is a simple example of using a PVC. In this example, the created Pod uses the newly created *mypvc*.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  selector:
    matchLabels:
      app: nginx
  replicas: 2
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - image: nginx:alpine
        name: container-0
        volumeMounts:
        - mountPath: /tmp
          name: pvc-mypvc
      restartPolicy: Always
      volumes:
      - name: pvc-mypvc
        persistentVolumeClaim:
          claimName:  mypvc                 # name of PVC
```

8.1.1.2 Manage Volume Provisioning

Manage Volume Provisioning allows administrators to use resources created on storage as PVs and supports features of dynamic volumes, such as capacity expansion, snapshot, and clone. This is a custom capability of Huawei CSI. This feature applies to the following scenarios:

- In the reconstruction containerized applications, existing storage volumes need to be used.
- The Kubernetes cluster is rebuilt.
- Storage data is migrated in disaster recovery (DR) scenarios.

Prerequisites

- You have registered the storage where the volume to be managed resides with CSI.
- You have logged in to the storage device to obtain the name and capacity of the volume to be managed.

8.1.1.2.1 Configuring a StorageClass

A **StorageClass** provides administrators with methods to describe a storage "class". Different types may map to a different group of capability definitions. Kubernetes cluster users can dynamically provision volumes based on a StorageClass.

A StorageClass supports the following parameters.

If SAN storage is used, refer to example file **/examples/sc-lun.yaml**. If NAS storage is used, refer to example file **/examples/sc-fs.yaml**.

Table 8-7 StorageClass configuration parameters

Parameter	Description	Mandatory	Default Value	Remarks
metadata.name	User-defined name of a StorageClass object.	Yes	-	Take Kubernetes v1.22.1 as an example. The value can contain digits, lowercase letters, hyphens (-), and periods (.), and must start and end with a letter or digit.
provisioner	Name of the provisioner.	Yes	csi.huawei.com	Set this parameter to the driver name set during Huawei CSI installation. The value is the same as that of **driverName** in the **values.yaml** file.
reclaimPolicy	Reclamation policy. The following types are supported: • **Delete**: Resources are automatically reclaimed. • **Retain**: Resources are manually reclaimed.	No	Delete	• **Delete**: When a PV/PVC is deleted, resources on the storage device are also deleted. • **Retain**: When a PV/PVC is deleted, resources on the storage device are not deleted.

Parameter	Description	Mandatory	Default Value	Remarks
allowVolumeExpansion	Whether to allow volume expansion. If this parameter is set to **true**, the capacity of the PV that uses the StorageClass can be expanded.	No	false	This function can only be used to expand PV capacity but cannot be used to reduce PV capacity. The PV capacity expansion function is supported in Kubernetes 1.14 (alpha) and later versions.
parameters.backend	Name of the backend where the resource to be created is located.	No	-	If this parameter is not set, Huawei CSI will randomly select a backend that meets the capacity requirements to create resources. You are advised to specify a backend to ensure that the created resource is located on the expected backend.
parameters.pool	Name of the storage resource pool where the resource to be created is located. If this parameter is set, **parameters.backend** must also be specified.	No	-	If this parameter is not set, Huawei CSI will randomly select a storage pool that meets the capacity requirements from the selected backend to create resources. You are advised to specify a storage pool to ensure that the created resource is located in the expected storage pool.
parameters.volumeType	Type of the volume to be created. The following types are supported: • **lun**: A LUN is provisioned on the storage side. • **fs**: A file system is provisioned on the storage side.	Yes	-	• If NAS storage is used, this parameter must be set to **fs**. • If SAN storage is used, this parameter must be set to **lun**.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.allocType	Allocation type of the volume to be created. The following types are supported: • **thin**: Not all required space is allocated during creation. Instead, the space is dynamically allocated based on the usage. • **thick**: All required space is allocated during creation.	No	-	If this parameter is not specified, **thin** will be used. Not all required space is allocated during creation. Instead, the space is dynamically allocated based on the usage. OceanStor Dorado V3/V6 does not support **thick**.
parameters.fsType	Type of a host file system. The supported types are: • ext2 • ext3 • ext4 • xfs	No	ext4	This parameter is valid only when **volumeType** of a StorageClass is set to **lun** and **volumeMode** of a PVC is set to **Filesystem**.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.authClient	IP address of the NFS client that can access the volume. This parameter is mandatory when **volumeType** is set to **fs**. You can enter the client host name (a full domain name is recommended), client IP address, or client IP address segment.	Conditionally mandatory	*	The asterisk (*) can be used to indicate any client. If you are not sure about the IP address of the access client, you are advised to use the asterisk (*) to prevent the client access from being rejected by the storage system. If the client host name is used, you are advised to use the full domain name. The IP addresses can be IPv4 addresses, IPv6 addresses, or a combination of IPv4 and IPv6 addresses. You can enter multiple host names, IP addresses, or IP address segments and separate them with semicolons (;) or spaces or by pressing **Enter**. Example: **192.168.0.10;192.168.0.0/24;myserver1.test**
parameters.cloneSpeed	Cloning speed. The value ranges from 1 to 4.	No	-	If this parameter is not set, the default value **3** is used. **4** indicates the highest speed. This parameter is available when you clone a PVC or create a PVC using a snapshot. For details, see **8.1.3 Cloning a PVC** or **8.1.4 Creating a PVC Using a Snapshot**.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.applicationType	Application type name for creating a LUN or NAS when the backend is OceanStor Dorado V6.	No	-	• If the value of **volumeType** is **lun**, log in to DeviceManager and choose **Services** > **Block Service** > **LUN Groups** > **LUNs** > **Create** to obtain the application type name. • If the value of **volumeType** is **fs**, log in to DeviceManager and choose **Services** > **File Service** > **File Systems** > **Create** to obtain the application type name.
parameters.qos	LUN/NAS QoS settings of the PV on the storage side. The value of the parameter is JSON character strings in dictionary format. A character string is enclosed by single quotation marks and the dictionary key by double quotation marks. Example: **'{"maxMBPS": 999, "maxIOPS": 999}'**	No	-	For details about the supported QoS configurations, see **Table 8-8**.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.storageQuota	Quota of a PV on the storage device. This parameter is valid only when NAS is used for connecting to OceanStor Pacific series storage. The value of the parameter is JSON character strings in dictionary format. A character string is enclosed by single quotation marks and the dictionary key by double quotation marks. Example: **'{"spaceQuota": "softQuota", "gracePeriod": 100}'**	No	-	For details about the supported quota configurations, see **Table 8-9**.
parameters.hyperMetro	Whether a HyperMetro volume is to be created. This parameter needs to be configured when the backend is of the HyperMetro type. • **"true"**: The created volume is a HyperMetro volume. • **"false"**: The created volume is a common volume.	Conditionally mandatory	false	Set this parameter when the used backend is a HyperMetro backend and a HyperMetro volume needs to be provisioned.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.fsPermission	Permission on the directory mounted to a container.	No	-	For details about the configuration format, refer to the Linux permission settings, for example, 777 and 755. All SAN storage devices are supported. Only the following NAS storage devices are supported: OceanStor Dorado V6, OceanStor V6, and OceanStor Pacific series 8.1.2 and later versions.
parameters.rootSquash	Controls the **root** permission of the client. The value can be: • **root_squash**: The client cannot access the storage system as user **root**. If a client accesses the storage system as user **root**, the client will be mapped as an anonymous user. • **no_root_squash**: A client can access the storage system as user **root** and has the permission of user **root**.	No	-	Only NAS storage is supported.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.allSquash	Whether to retain the user ID (UID) and group ID (GID) of a shared directory. The value can be: • **all_squash**: The UID and GID of the shared directory are mapped to anonymous users. • **no_all_squash**: The UID and GID of the shared directory are retained.	No	-	Only NAS storage is supported.
parameters.snapshotDirectoryVisibility	Whether the snapshot directory is visible. The value can be: • **visible**: The snapshot directory is visible. • **invisible**: The snapshot directory is invisible.	No	-	Only NAS storage is supported.
parameters.reservedSnapshotSpaceRatio	Configures reserved snapshot space. Value type: character string Value range: 0 to 50	No	-	OceanStor Dorado 6.1.5+ and OceanStor 6.1.5+ NAS storage devices are supported.

Parameter	Description	Mandatory	Default Value	Remarks
parameters.description	Configures the description of the created file system or LUN. Value type: character string The value contains 0 to 255 characters.	No	-	Only enterprise storage file systems and LUNs are supported.
mountOptions.nfsvers	NFS mount option on the host. The following mount option is supported: **nfsvers**: protocol version for NFS mounting. The value can be **3**, **4**, **4.0**, or **4.1**.	No	-	This parameter is optional after the **-o** parameter when the **mount** command is executed on the host. The value is in list format. If the NFS version is specified for mounting, NFS 3, 4.0, and 4.1 protocols are supported (the protocol must be supported and enabled on storage devices). If **nfsvers** is set to **4**, the latest protocol version NFS 4 may be used for mounting due to different OS configurations, for example, 4.1. If protocol 4.0 is required, you are advised to set **nfsvers** to **4.0**.
mountOptions.acl	The DPC namespace supports the ACL function. The DPC client supports POSIX ACL, NFSv4 ACL, and NT ACL authentication.	No	-	The descriptions of **acl**, **aclonlyposix**, **cnflush**, and **cflush** are for reference only. For details about the parameters, see ***OceanStor Pacific Series Product Documentation*** and choose **Configuration** > **Basic Service Configuration Guide for File** > **Configuring Basic Services (DPC Scenario)** > **Accessing a DPC Share on a Client** > **Step 2**.

Parameter	Description	Mandatory	Default Value	Remarks
mountOptions.aclonlyposix	The DPC namespace supports POSIX ACL, and the DPC client supports POSIX ACL authentication. The following protocols support POSIX ACL: DPC, NFSv3, and HDFS. If NFSv4 ACL or NT ACL is used, the DPC client cannot identify the ACL of this type. As a result, the ACL of this type does not take effect.	No	-	If **aclonlyposix** and **acl** are used together, only **acl** takes effect. That is, the namespace supports the ACL function.
mountOptions.cnflush	Asynchronous disk flushing mode. That is, data is not flushed to disks immediately when files in the namespace are closed.	No	-	Asynchronous flushing mode: When a file is closed, data in the cache is not flushed to storage media in synchronous mode. Instead, data is written from the cache to the storage media in asynchronous flushing mode. After the write service is complete, data is flushed from the cache to disks periodically based on the flushing period. In a multi-client scenario, if concurrent operations are performed on the same file, the file size update is affected by the disk flushing period. That is, the file size is updated only after the disk flushing is complete. Generally, the update is completed within several seconds. Synchronous I/Os are not affected by the disk flushing period.

Parameter	Description	Mandatory	Default Value	Remarks
mountOptions.cflush	Synchronous disk flushing mode. That is, data is flushed to disks immediately when files in the namespace are closed.	No	-	By default, the synchronous disk flushing mode is used.

Table 8-8 Supported QoS configurations

Storage Type	Parameter	Description	Remarks
OceanStor V3/ OceanStor V5	IOTYPE	Read/write type.	This parameter is optional. If it is not specified, the default value of the storage backend is used. For details, see related storage documents. The value can be: • **0**: read I/O • **1**: write I/O • **2**: read and write I/Os
	MAXBANDWIDTH	Maximum bandwidth. This is a restriction policy parameter.	The value is an integer greater than 0, expressed in MB/s.
	MINBANDWIDTH	Minimum bandwidth. This is a protection policy parameter.	The value is an integer greater than 0, expressed in MB/s.
	MAXIOPS	Maximum IOPS. This is a restriction policy parameter.	The value is an integer greater than 0.
	MINIOPS	Minimum IOPS. This is a protection policy parameter.	The value is an integer greater than 0.
	LATENCY	Maximum latency. This is a protection policy parameter.	The value is an integer greater than 0, expressed in ms.

Storage Type	Parameter	Description	Remarks
OceanStor Dorado V3	IOTYPE	Read/write type.	The value can be: ● **2**: read and write I/Os
	MAXBANDWIDTH	Maximum bandwidth. This is a restriction policy parameter.	The value is an integer ranging from 1 to 999999999, expressed in MB/s.
	MAXIOPS	Maximum IOPS. This is a restriction policy parameter.	The value is an integer ranging from 100 to 999999999.
OceanStor Dorado V6/ OceanStor V6	IOTYPE	Read/write type.	The value can be: ● **2**: read and write I/Os
	MAXBANDWIDTH	Maximum bandwidth. This is a restriction policy parameter.	The value is an integer ranging from 1 to 999999999, expressed in MB/s.
	MINBANDWIDTH	Minimum bandwidth. This is a protection policy parameter.	The value is an integer ranging from 1 to 999999999, expressed in MB/s.
	MAXIOPS	Maximum IOPS. This is a restriction policy parameter.	The value is an integer ranging from 100 to 999999999.
	MINIOPS	Minimum IOPS. This is a protection policy parameter.	The value is an integer ranging from 100 to 999999999.
	LATENCY	Maximum latency. This is a protection policy parameter.	The value can be **0.5** or **1.5**, expressed in ms.
FusionStorage/ OceanStor Pacific series	maxMBPS	Maximum bandwidth. This is a restriction policy parameter.	This parameter is mandatory. The value is an integer greater than 0, expressed in MB/s.
	maxIOPS	Maximum IOPS. This is a restriction policy parameter.	This parameter is mandatory. The value is an integer greater than 0.

NOTE

- vStore users of OceanStor V3/OceanStor V5 cannot configure QoS policies.
- The QoS configuration takes effect only on the newly created PVC. QoS cannot be added automatically for PVCs with the same StorageClass name that have been provisioned.

Table 8-9 Supported quota configurations

Parameter	Description	Remarks
spaceQuota	File quota type.	This parameter is mandatory. Only **softQuota** or **hardQuota** can be configured.
gracePeriod	Grace period allowed when the soft quota is configured.	This parameter is conditionally optional only when **spaceQuota** is set to **softQuota**. The value is an integer ranging from 0 to 4294967294.

For details about how to configure a StorageClass in typical scenarios, see the following examples:

- **Setting the Backend and Storage Pool in a StorageClass**
- **Setting the NFS Access Mode in a StorageClass**
- **Setting the Local File System Access Mode in a StorageClass**
- **Setting the DPC Access Mode in a StorageClass**
- **Setting an Application Type in a StorageClass**
- **Setting a Soft Quota in a StorageClass**
- **Setting HyperMetro in a StorageClass**
- **Setting the Permission on a Mount Directory in a StorageClass**
- **Setting QoS in a StorageClass**

Setting the Backend and Storage Pool in a StorageClass

If multiple Huawei backends are configured in a Kubernetes cluster or a Huawei backend provides multiple storage pools, you are advised to configure the specified backend and storage pool information in the StorageClass. This prevents Huawei CSI from randomly selecting backends and storage pools and ensures that the storage device where the volume resides complies with the plan.

For details about how to set the backend and storage pool for SAN storage, see the following configuration example.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
allowVolumeExpansion: true
parameters:
  backend: "iscsi_san_181"
  pool: "pool001"
  volumeType: lun
  allocType: thin
```


For details about how to set the backend and storage pool for NAS storage, see the following configuration example.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
allowVolumeExpansion: true
parameters:
  backend: "iscsi_nas_181"
  pool: "pool001"
  volumeType: fs
  allocType: thin
  authClient: "*"
```

Setting the NFS Access Mode in a StorageClass

When a container uses an NFS file system as a storage resource, refer to the following configuration example. In this example, NFS version 4.1 is specified for mounting.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: nfs_nas_181
  pool: pool001
  volumeType: fs
  allocType: thin
  authClient: "192.168.0.10;192.168.0.0/24;myserver1.test"  #use * for all client
mountOptions:
  - nfsvers=4.1
```

Setting the Local File System Access Mode in a StorageClass

If a container uses a LUN of enterprise storage or distributed storage as a storage resource and a file system needs to be formatted as a local file system, refer to the following example.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: iscsi_lun_181
  pool: pool001
  volumeType: lun
  allocType: thin
  fsType: xfs
```

Setting the DPC Access Mode in a StorageClass

If a container uses OceanStor Pacific series storage and the storage supports DPC-based access, you can configure mounting parameters for DPC-based access in the StorageClass. In this example, **acl** is used as the authentication parameter for mounting, and **cnflush** is used to set the asynchronous disk flushing mode.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
```



```
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: nfs_dpc_101
  pool: pool001
  volumeType: fs
  allocType: thin
  authClient: "*"  #use * for all client
mountOptions:
  - acl
  - cnflush
```

Setting an Application Type in a StorageClass

When a container uses a LUN of OceanStor Dorado V6 as the storage, if the default application type of the storage cannot meet the I/O model requirements of some services (for example, the container provides the database OLAP service), you can configure an application type in the StorageClass to improve storage performance. For details about the application types to be used, see the product documentation of the corresponding storage product.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: iscsi_lun_181
  pool: pool001
  volumeType: lun
  allocType: thin
  fsType: xfs
  applicationType: Oracle_OLAP
```

Setting a Soft Quota in a StorageClass

If a container uses a file system of OceanStor Pacific series as the storage, you can configure a soft quota in the StorageClass. The following is a configuration example.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: nfs_pacific_101
  pool: pool001
  volumeType: fs
  allocType: thin
  authClient: "*"
  storageQuota: '{"spaceQuota": "softQuota", "gracePeriod": 100}'
mountOptions:
  - nfsvers=3
```

Setting QoS in a StorageClass

When containers use enterprise storage or distributed storage as storage resources, you can set QoS for the storage resources used by containers to ensure that the storage read and write operations of these containers meet certain service levels.

Storage devices of different models or versions support different QoS settings. For details about how to find the configuration items of the corresponding storage

devices, see **Table 8-8**. In this example, the backend is OceanStor Dorado V6. For other storage devices, refer to this example.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: iscsi_qos_181
  pool: pool001
  volumeType: lun
  allocType: thin
  fsType: xfs
  qos: '{"IOTYPE": 2, "MINIOPS": 1000}'
```

Setting HyperMetro in a StorageClass

When a container uses an NFS HyperMetro file system as a storage resource, refer to the following configuration example. In this example, the used backend supports HyperMetro, and **hyperMetro** is set to **true**.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
parameters:
  backend: nfs_hypermetro_dorado_181
  pool: pool001
  volumeType: fs
  hyperMetro: "true"
  allocType: thin
  authClient: "*"
```

NOTICE

Before provisioning a NAS HyperMetro volume, you need to configure the HyperMetro relationship between two storage devices, including the remote device, HyperMetro domain, and the like. The HyperMetro domain of the file system can only work in HyperMetro mode. For details about the configuration operation, see the product documentation of the corresponding storage model.

Setting the Permission on a Mount Directory in a StorageClass

To modify the permission on a mount directory in a container, you can configure the directory permission in a StorageClass. The following is a configuration example.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: mysc
provisioner: csi.huawei.com
allowVolumeExpansion: true
parameters:
  volumeType: fs
  allocType: thin
  authClient: "*"
  fsPermission: "777"
  rootSquash: "no_root_squash"  # This parameter supports only NAS storage.
  allSquash: "no_all_squash"  # This parameter supports only NAS storage.
```


After the StorageClass configuration is complete, perform the following steps to create a StorageClass.

Step 1 Run the following command to create a StorageClass based on the .yaml file.

```
# kubectl create -f mysc.yaml
storageclass.storage.k8s.io/mysc created
```

Step 2 Run the following command to view the information about the created StorageClass.

```
# kubectl get sc
NAME   PROVISIONER      RECLAIMPOLICY   VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
mysc   csi.huawei.com   Delete          Immediate           false                  34s
```

After creating a StorageClass, you can use the StorageClass to create a PV or PVC.

----End

CAUTION

Pay attention to the following when using a StorageClass:

- Do not delete a StorageClass that is being used by a PV. Otherwise, the StorageClass information of the PV will be missing. As a result, an error occurs during mounting on the host.
- Modifications to a StorageClass do not take effect on existing PVs. You need to delete these PVs and create them again using the modified StorageClass to apply the modified parameters.

8.1.1.2.2 Configuring a PVC

After configuring a StorageClass, you can use the StorageClass to configure a PVC. For details about the PVC configuration template, see example file **pvc-manager.yaml** in the **examples** directory in Huawei CSI software package.

Table 8-10 Parameters in the pvc-manager.yaml file

Parameter	Description	Mandatory	Default Value	Remarks
metadata.annotations	PVC object annotations. Set the following parameters: • *Driver name*/**manageVolumeName**: volume name on the storage. • *Driver name*/**manageBackendName**: name of the backend to which the volume belongs.	Yes	csi.huawei.com/manageVolumeName: * csi.huawei.com/manageBackendName: *	• For details about how to obtain *Driver name*, see **4.1.1.4 csiDriver Parameters**. • *Driver name*/**manageVolumeName**: name of an existing volume on the storage. Only English characters are supported. • *Driver name*/**manageBackendName**: name of the storage backend in CSI. You can run the **oceanctl get backend -n huawei-csi** command to obtain the backend name.
metadata.labels	PVC object labels.	Yes	-	Format: **provisioner:** *Driver name specified during installation* Example: **provisioner: csi.huawei.com** This parameter takes effect when a PVC is created. It is used to listen to PVC resources and obtain information about **metadata.annotations**.
metadata.name	User-defined name of a PVC object.	Yes	-	Take Kubernetes v1.22.1 as an example. The value can contain digits, lowercase letters, hyphens (-), and periods (.), and must start and end with a letter or digit.

Parameter	Description	Mandatory	Default Value	Remarks
spec.volumeMode	Volume mode. This parameter is optional. When LUN volumes are used, the following types are supported: • **Filesystem**: local file system. • **Block**: raw device.	No	Filesystem	This parameter takes effect when a PV is mounted. • **Filesystem** indicates that a container accesses a PV using a local file system. The local file system type is specified by the **fsType** field in the specified StorageClass. • **Block** indicates that a PV is accessed in raw volume mode.

Parameter	Description	Mandatory	Default Value	Remarks
	NOTE This parameter takes effect when a PV is mounted. The use method of this parameter must be the same as that of the managed volume. ● If a volume is used as a raw volume before being managed, **volumeMode** must be set to **Block**. ● If a volume is used in ext2, ext3, or ext4 mode before being managed, **volumeMode** must be set to **Filesystem** and **fsType** in the StorageClass must be set to **ext2**, **ext3**, or **ext4**. ● If a volume is used in XFS mode before being managed, **volumeMode** must be set to **Filesystem** and **fsType** in the StorageClass must be set to **xfs**.			
spec.storageClassName	Name of the StorageClass object.	Yes	-	The configuration of the StorageClass must be the same as that of the managed volume.

Parameter	Description	Mandatory	Default Value	Remarks
spec.resources.requests.storage	Size of the volume to be created. The format is ***Gi and the unit is GiB. The size must be an integer multiple of 512 bytes.	Yes	-	The PVC capacity depends on storage specifications and host specifications. For example, OceanStor Dorado 6.1.2 or OceanStor Pacific series 8.1.0 is connected to CentOS 7. If ext4 file systems are used, see **Table 8-11**. If XFS file systems are used, see **Table 8-12**. If NFS or raw devices are used, the capacity must meet the specifications of the used Huawei storage device model and version. If the PVC capacity does not meet the specifications, a PVC or Pod may fail to be created due to the limitations of storage specifications or host file system specifications.

Parameter	Description	Mandatory	Default Value	Remarks
spec.accessModes	Access mode of the volume. • **RWO** (ReadWriteOnce): A volume can be mounted to a node in read/write mode. This mode also allows multiple Pods running on the same node to access the volume. • **ROX** (ReadOnlyMany): A volume can be mounted to multiple nodes in read-only mode. • **RWX** (ReadWriteMany): A volume can be mounted to multiple nodes in read/write mode. • **RWOP** (ReadWriteOncePod): A volume can only be mounted to a single Pod in read/write mode. Kubernetes 1.22 and later versions support this feature.	Yes	ReadWriteOnce	• RWO/ROX/RWOP: supported by all types of volumes. RWOP is supported only by Kubernetes 1.22 and later versions. Check whether this feature is enabled for your Kubernetes cluster by referring to **10.5 Enabling the ReadWriteOncePod Feature Gate**. • RWX: supported by volumes whose **volumeMode** is set to **Block** or **NFS**.

Table 8-11 ext4 capacity specifications

Storage Type	Storage Specifications	ext4 Specifications	CSI Specifications
OceanStor Dorado 6.1.2	512 Ki to 256 Ti	50 Ti	512 Ki to 50 Ti
OceanStor Pacific series 8.1.0	64 Mi to 512 Ti	50 Ti	64 Mi to 50 Ti

Table 8-12 XFS capacity specifications

Storage Type	Storage Specifications	XFS Specifications	CSI Specifications
OceanStor Dorado 6.1.2	512 Ki to 256 Ti	500 Ti	512 Ki to 500 Ti
OceanStor Pacific series 8.1.0	64 Mi to 512 Ti	500 Ti	64 Mi to 500 Ti

Step 1 Based on service requirements, modify specific parameters by referring to the description in this section and the PVC configuration file example to generate the PVC configuration file to be created, for example, the **mypvc.yaml** file in this example.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: mypvc
  annotations:
    csi.huawei.com/manageVolumeName: *
    csi.huawei.com/manageBackendName: *
  labels:
    provisioner: csi.huawei.com
spec:
  accessModes:
    - ReadWriteOnce
 volumeMode: Filesystem
 storageClassName: mysc
 resources:
   requests:
     storage: 100Gi
```

Step 2 Run the following command to create a PVC using the configuration file.

```
# kubectl create -f mypvc.yaml
persistentvolumeclaim/mypvc created
```

Step 3 After a period of time, run the following command to view the information about the created PVC. If the PVC status is **Bound**, the PVC has been created and can be used by a Pod.

```
# kubectl get pvc mypvc
NAME       STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
mypvc      Bound    pvc-840054d3-1d5b-4153-b73f-826f980abf9e   100Gi      RWO            mysc           12s
```


CAUTION

- After the PVC is created, if the PVC is in the **Pending** state after a long time (for example, one minute), refer to **11.6 When a PVC Is Created, the PVC Is in the Pending State**.
- You are advised to create or delete a maximum of 100 PVCs in a batch.

----End

After a PVC is created, you can use the PVC to create a Pod. The following is a simple example of using a PVC. In this example, the created Pod uses the newly created *mypvc*.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  selector:
    matchLabels:
      app: nginx
  replicas: 2
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - image: nginx:alpine
        name: container-0
        volumeMounts:
        - mountPath: /tmp
          name: pvc-mypvc
      restartPolicy: Always
      volumes:
      - name: pvc-mypvc
        persistentVolumeClaim:
          claimName:  mypvc                 # name of PVC
```

8.1.1.3 Static Volume Provisioning

Static volume provisioning allows administrators to use a resource created on the storage side as a PV for containers in the cluster.

To implement static volume provisioning, perform the following steps:

- Configuring a PV
- Configuring a PVC

8.1.1.3.1 Configuring a PV

If static volume provisioning is used, you do not need to configure a StorageClass. Instead, you can directly create a PV. Before creating a PV, you need to configure a PV. The following example is a configuration file for static volume provisioning.

```
kind: PersistentVolume
apiVersion: v1
metadata:
  name: mypv
spec:
  volumeMode: Filesystem
```



```
storageClassName: ""
accessModes:
  - ReadWriteOnce
csi:
  driver: csi.huawei.com
  volumeHandle: iscsi-dorado-181.lun0001
  fsType: xfs
capacity:
  storage: 100Gi
```

As shown in the preceding example, in the configuration file for static volume provisioning, **storageClassName** must be set to **""**. Otherwise, Kubernetes will use the default StorageClass. For details about other parameters, see **Table 8-13**.

Table 8-13 Static volume provisioning parameters

Parameter	Description	Mandatory	Default Value	Remarks
metadata.name	User-defined name of a PVC object.	Yes	-	Take Kubernetes v1.22.1 as an example. The value can contain digits, lowercase letters, hyphens (-), and periods (.), and must start and end with a letter or digit.
spec.volumeMode	Volume mode. This parameter is optional. When LUN volumes are used, the following types are supported: • **Filesystem**: local file system. • **Block**: raw device.	No	Filesystem	This parameter takes effect when a PV is mounted. The default value is **Filesystem**. • **Filesystem** indicates that a container accesses a PV using a local file system. The local file system type is specified by the **fsType** field in the specified StorageClass. • **Block** indicates that a PV is accessed in raw volume mode.
spec.storageClassName	Name of the StorageClass object. This parameter is mandatory.	Yes	-	Set the parameter to an empty string, that is, enter **""**.

Parameter	Description	Mandatory	Default Value	Remarks
spec.accessModes	Access mode of the volume. • **RWO** (ReadWriteOnce): A volume can be mounted to a node in read/write mode. This mode also allows multiple Pods running on the same node to access the volume. • **ROX** (ReadOnlyMany): A volume can be mounted to multiple nodes in read-only mode. • **RWX** (ReadWriteMany): A volume can be mounted to multiple nodes in read/write mode. • **RWOP** (ReadWriteOncePod): A volume can only be mounted to a single Pod in read/write mode. Kubernetes 1.22 and later versions support this feature.	Yes	ReadWriteOnce	• RWO/ROX/RWOP: supported by all types of volumes. RWOP is supported only by Kubernetes 1.22 and later versions. Check whether this feature is enabled for your Kubernetes cluster by referring to **10.5 Enabling the ReadWriteOncePod Feature Gate**. • RWX: supported by volumes whose **volumeMode** is set to **Block** or **NFS**.

Parameter	Description	Mandatory	Default Value	Remarks
spec.csi.driver	CSI driver name.	Yes	csi.huawei.com	Set this parameter to the driver name set during Huawei CSI installation.
spec.csi.volumeHandle	Unique identifier of a storage resource. This parameter is mandatory. Format: *<backendName>.<volume-name>*	Yes	-	The value of this parameter consists of the following parts: ● *<backendName>*: indicates the name of the backend where the volume resides. You can run the following command to obtain the configured backend information. **oceanctl get backend** ● *<volume-name>*: indicates the name of a resource (LUN/file system) on the storage. You can obtain the value from DeviceManager.
spec.csi.fsType	Type of a host file system. This parameter is optional. The supported types are: ● ext2 ● ext3 ● ext4 ● xfs	No	-	If this parameter is not set, the default value **ext4** is used. This parameter is available only when **volumeMode** is set to **Filesystem**.

Parameter	Description	Mandatory	Default Value	Remarks
spec.capacity.storage	Volume size.	Yes	100Gi	Ensure that the size is the same as that of the corresponding resource on the storage. Kubernetes will not invoke CSI to check whether the value of this parameter is correct. Therefore, the PV can be successfully created even if its capacity is inconsistent with that of the corresponding resource on the storage.
spec.mountOptions.nfsvers	NFS mount option on the host. The following mount option is supported: **nfsvers**: protocol version for NFS mounting. The value can be **3**, **4**, **4.0**, or **4.1**.	No	-	This parameter is optional after the **-o** parameter when the **mount** command is executed on the host. The value is in list format. If the NFS version is specified for mounting, NFS 3, 4.0, and 4.1 protocols are supported (the protocol must be supported and enabled on storage devices). If **nfsvers** is set to **4**, the latest protocol version NFS 4 may be used for mounting due to different OS configurations, for example, 4.1. If protocol 4.0 is required, you are advised to set **nfsvers** to **4.0**.

Parameter	Description	Mandatory	Default Value	Remarks
spec.mountOptions.acl	The DPC namespace supports the ACL function. The DPC client supports POSIX ACL, NFSv4 ACL, and NT ACL authentication.	No	-	The descriptions of **acl**, **aclonlyposix**, **cnflush**, and **cflush** are for reference only. For details about the parameters, see ***OceanStor Pacific Series Product Documentation*** and choose **Configuration** > **Basic Service Configuration Guide for File** > **Configuring Basic Services (DPC Scenario)** > **Accessing a DPC Share on a Client** > **Step 2**.
spec.mountOptions.aclonlyposix	The DPC namespace supports POSIX ACL, and the DPC client supports POSIX ACL authentication. The following protocols support POSIX ACL: DPC, NFSv3, and HDFS. If NFSv4 ACL or NT ACL is used, the DPC client cannot identify the ACL of this type. As a result, the ACL of this type does not take effect.	No	-	If **aclonlyposix** and **acl** are used together, only **acl** takes effect. That is, the namespace supports the ACL function.

Parameter	Description	Mandatory	Default Value	Remarks
spec.mountOptions.cnflush	Asynchronous disk flushing mode. That is, data is not flushed to disks immediately when files in the namespace are closed.	No	-	Asynchronous flushing mode: When a file is closed, data in the cache is not flushed to storage media in synchronous mode. Instead, data is written from the cache to the storage media in asynchronous flushing mode. After the write service is complete, data is flushed from the cache to disks periodically based on the flushing period. In a multi-client scenario, if concurrent operations are performed on the same file, the file size update is affected by the disk flushing period. That is, the file size is updated only after the disk flushing is complete. Generally, the update is completed within several seconds. Synchronous I/Os are not affected by the disk flushing period.
spec.mountOptions.cflush	Synchronous disk flushing mode. That is, data is flushed to disks immediately when files in the namespace are closed.	No	-	By default, the synchronous disk flushing mode is used.

Prerequisites

- A storage resource, such as a LUN or file system, required by the PV to be created exists on the storage device. If the storage resource is a file system, you also need to create the share and client information of the file system.
- You have configured the PV configuration file by referring to **Table 8-13**.

Procedure

Step 1 Run the following command to create a PV based on the prepared .yaml file.

```
# kubectl create -f mypv.yaml
persistentvolume/mypv created
```

Step 2 After a period of time, run the following command to view the information about the created PV. If the PV status is **Available**, the PV is successfully created.

```
# kubectl get pv
NAME      CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS      CLAIM             STORAGECLASS
REASON   AGE
mypv      100Gi      RWO            Retain           Available                     4s
```

----End

8.1.1.3.2 Configuring a PVC

After a PV is created in static volume provisioning mode, you can create a PVC based on the PV for containers. The following example is a PVC configuration file for static volume provisioning.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: mypvc
spec:
  accessModes:
    - ReadWriteOnce
  volumeMode: Filesystem
  resources:
    requests:
      storage: 100Gi
  volumeName: mypv
```

As shown in the preceding example, set the **volumeName** parameter in the PVC configuration file to the PV created in static volume provisioning mode. For details about the parameters, see **Table 8-14**.

Table 8-14 PVC parameters

Parameter	Description	Mandatory	Default Value	Remarks
metadata.name	User-defined name of a PVC object.	Yes	-	Take Kubernetes v1.22.1 as an example. The value can contain digits, lowercase letters, hyphens (-), and periods (.), and must start and end with a letter or digit.

Parameter	Description	Mandatory	Default Value	Remarks
spec.accessModes	Access mode of the volume. • **RWO** (ReadWriteOnce): A volume can be mounted to a node in read/write mode. This mode also allows multiple Pods running on the same node to access the volume. • **ROX** (ReadOnlyMany): A volume can be mounted to multiple nodes in read-only mode. • **RWX** (ReadWriteMany): A volume can be mounted to multiple nodes in read/write mode. • **RWOP** (ReadWriteOncePod): A volume can only be mounted to a single Pod in read/write mode. Kubernetes 1.22 and later versions support this feature.	Yes	ReadWriteOnce	• RWO/ROX/RWOP: supported by all types of volumes. RWOP is supported only by Kubernetes 1.22 and later versions. Check whether this feature is enabled for your Kubernetes cluster by referring to **10.5 Enabling the ReadWriteOncePod Feature Gate**. • RWX: supported by volumes whose **volumeMode** is set to **Block** or **NFS**.

Parameter	Description	Mandatory	Default Value	Remarks
spec.volumeMode	Volume mode.	No	Filesystem	This parameter is optional. The value can be **Filesystem** or **Block**. The default value is **Filesystem**. This parameter takes effect when a Pod is created. **Filesystem** indicates that a file system is created on a PVC to access the storage. **Block** indicates that a raw volume is used to access the storage.

Parameter	Description	Mandatory	Default Value	Remarks
spec.resources.requests.storage	Size of the volume to be created.	Yes	-	Size of the volume to be created. The format is ***Gi and the unit is GiB. The PVC capacity depends on storage specifications and host specifications. For example, OceanStor Dorado 6.1.2 or OceanStor Pacific series 8.1.0 is connected to CentOS 7. If ext4 file systems are used, see **Table 8-5**. If XFS file systems are used, see **Table 8-6**. If NFS or raw devices are used, the capacity must meet the specifications of the used Huawei storage device model and version. If the PVC capacity does not meet the specifications, a PVC or Pod may fail to be created due to the limitations of storage specifications or host file system specifications. When a PVC is created using a static PV and the PVC capacity is smaller than the capacity of the bound PV, the PVC capacity is set to the capacity of the bound PV. If the PVC capacity is greater than the capacity of the bound PV, the PVC cannot be created.
spec.volumeName	Name of the PV object.	Yes	-	This parameter is mandatory when a PVC is created statically.

Procedure

Step 1 Run the following command to create a PVC based on the configured .yaml file.

```
# kubectl create -f mypvc.yaml
persistentvolumeclaim/mypvc created
```

Step 2 After a period of time, run the following command to view the information about the created PVC.

```
# kubectl get pvc
NAME      STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
mypvc     Bound    pvc-840054d3-1d5b-4153-b73f-826f980abf9e   100Gi      RWO                           12s
```

NOTE

- After the PVC is created, if the PVC is in the **Pending** state after a long time (for example, one minute), refer to **11.6 When a PVC Is Created, the PVC Is in the Pending State**.
- You are advised to create or delete a maximum of 100 PVCs in a batch.

----End

After a PVC is created, you can use the PVC to create a Pod. The following is a simple example of using a PVC. In this example, the created Pod uses the newly created *mypvc*.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  selector:
    matchLabels:
      app: nginx
  replicas: 2
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
      - image: nginx:alpine
        name: container-0
        volumeMounts:
        - mountPath: /tmp
          name: pvc-mypvc
      restartPolicy: Always
      volumes:
      - name: pvc-mypvc
        persistentVolumeClaim:
          claimName:  mypvc                 # name of PVC
```

8.1.2 Expanding the Capacity of a PVC

When the capacity of a PVC used by a container is insufficient, you need to expand the capacity of the PVC.

Prerequisites

- A PVC has been created, the backend to which it resides exists and supports capacity expansion.
- For details about the storage devices that support capacity expansion, see **Table 2-5** and **Table 2-9**. For details about the Kubernetes versions that support capacity expansion, see **Table 2-3**.

- The csi-resizer service is enabled for huawei-csi-controller.

```
# kubectl describe deploy huawei-csi-controller -n huawei-csi | grep csi-resizer
  csi-resizer:
   Image:      k8s.gcr.io/sig-storage/csi-resizer:v1.4.0
```

Procedure

Step 1 Run the **kubectl get pvc** *mypvc* command to query the StorageClass name of the PVC. In the preceding command, *mypvc* indicates the name of the PVC to be expanded.

```
# kubectl get pvc mypvc
NAME            STATUS   VOLUME                                     CAPACITY   ACCESS MODES
STORAGECLASS    AGE
mypvc           Bound    pvc-3383be36-537c-4cb1-8f32-a415fa6ba384   2Gi        RW0
mysc            145m
```

Step 2 Run the **kubectl get sc** *mysc* command to check the StorageClass supports capacity expansion. In the preceding command, *mysc* indicates the name of the StorageClass to be queried.

```
# kubectl get sc mysc
NAME            PROVISIONER      RECLAIMPOLICY   VOLUMEBINDINGMODE
ALLOWVOLUMEEXPANSION   AGE
mysc            csi.huawei.com   Delete          Immediate          true                   172m
```

If the value of **ALLOWVOLUMEEXPANSION** is **true**, the current StorageClass supports capacity expansion. In this case, go to **Step 4**.

Step 3 Run the following command to change the value of **allowVolumeExpansion** to **true**. In the preceding command, *mysc* indicates the name of the StorageClass to be modified.

```
# kubectl patch sc mysc --patch '{"allowVolumeExpansion":true}'
```

Step 4 Run the following command to expand the capacity.

```
# kubectl patch pvc mypvc -p '{"spec":{"resources":{"requests":{"storage":"120Gi"}}}}'
```

In the preceding command, *mypvc* indicates the name of the PVC to be expanded, and *120Gi* indicates the capacity after expansion. Change the values based on the site requirements.

NOTE

- The PVC capacity depends on storage specifications and host specifications. For example, OceanStor Dorado 6.1.2 or OceanStor Pacific series 8.1.0 is connected to CentOS 7. If ext4 file systems are used, see **Table 8-5**. If XFS file systems are used, see **Table 8-6**. If NFS or raw devices are used, the capacity must meet the specifications of the used Huawei storage device model and version.
- If the PVC capacity does not meet the specifications, a PVC or Pod may fail to be created due to the limitations of storage specifications or host file system specifications.
- If the capacity expansion fails because the target capacity exceeds the storage pool capacity, see **11.18 Failed to Expand the PVC Capacity Because the Target Capacity Exceeds the Storage Pool Capacity**.

Step 5 Run the following command to check whether the capacity modification takes effect.

```
# kubectl get pvc
NAME      STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
mypvc     Bound    pvc-3383be36-537c-4cb1-8f32-a415fa6ba384   120Gi      RWO            mysc           24s
```

----End

8.1.3 Cloning a PVC

This section describes how to clone a PVC.

When cloning a PVC, you need to specify the data source. The following is a simple example of cloning a PVC. In this example, **mypvc** is used as the data source and a PVC named **myclone** is created.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: myclone
spec:
  storageClassName: mysc
  dataSource:
    name: mypvc
    kind: PersistentVolumeClaim
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
```

CAUTION

- The specified **storageClassName** must be the same as the StorageClass of the source volume in **dataSource**.
- The capacity of the clone volume must be greater than or equal to that of the source volume. Equal capacity is recommended.

Prerequisites

The source PVC already exists in the system, and the backend where the source PVC resides supports cloning. For details about the storage devices that support cloning, see **Table 2-5** and **Table 2-9**. For details about the Kubernetes versions that support cloning, see **Table 2-3**.

Procedure

Step 1 Run the following command to create a PVC based on the configuration file of the clone volume.

```
# kubectl create -f myclone.yaml
persistentvolumeclaim/myclone created
```

----End

8.1.4 Creating a PVC Using a Snapshot

This section describes how to create a PVC using a snapshot.

When creating a PVC, you need to specify the data source. The following is a simple example of creating a PVC using a snapshot. In this example, **mysnapshot** is used as the data source and a PVC named **myrestore** is created.

```
apiVersion: v1
kind: PersistentVolumeClaim
```



```
metadata:
  name: myrestore
spec:
  storageClassName: mysc
  dataSource:
    name: mysnapshot
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 100Gi
```

CAUTION

- The specified **storageClassName** must be the same as the StorageClass of the snapshot source volume in **dataSource**.
- The capacity of the clone volume must be greater than or equal to that of the snapshot. Equal capacity is recommended.

Prerequisites

A snapshot already exists in the system, and the backend where the snapshot resides supports cloning. For details about the storage devices that support PVC creation using a snapshot, see **Table 2-5** and **Table 2-9**. For details about the Kubernetes versions that support PVC creation using a snapshot, see **Table 2-3**.

Procedure

Step 1 Run the following command to create a PVC based on the configuration file for creating a volume using a snapshot.

```
# kubectl create -f myrestore.yaml
persistentvolumeclaim/myrestore created
```

----End

8.2 Creating a VolumeSnapshot

In Kubernetes, a **VolumeSnapshot** is a snapshot of a volume on a storage system. The VolumeSnapshot capability provides Kubernetes users with a standard way to replicate the content of a volume at a specified point in time without creating a volume. For example, this function enables database administrators to back up the database before making changes such as editing or deleting.

This section describes how to create a VolumeSnapshot using Huawei CSI. To create a VolumeSnapshot, perform the following steps:

- Checking information about volume snapshot-dependent components
- Configuring a VolumeSnapshotClass
- Configuring a VolumeSnapshot

8.2.1 Checking Information About Volume Snapshot-dependent Components

If you need to use volume snapshots and features associated with volume snapshots in the container environment, perform the operations in **3.5 Checking Volume Snapshot-Dependent Components** to check whether volume snapshot-dependent components have been deployed in your environment and check the api-versions information about volume snapshots.

8.2.2 Configuring a VolumeSnapshotClass

VolumeSnapshotClass provides a way to describe the "classes" of storage when provisioning a VolumeSnapshot. Each VolumeSnapshotClass contains the **driver**, **deletionPolicy**, and **parameters** fields, which are used when a VolumeSnapshot belonging to the class needs to be dynamically provisioned.

The name of a VolumeSnapshotClass object is significant, and is how users can request a particular class. Administrators set the name and other parameters of a class when first creating VolumeSnapshotClass objects, and the objects cannot be updated once they are created.

The following is an example of a VolumeSnapshotClass used by Huawei CSI:

- If api-versions in your environment supports v1, use the following example:
```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: mysnapclass
driver: csi.huawei.com
deletionPolicy: Delete
```
- If api-versions in your environment supports v1beta1, use the following example:
```
apiVersion: snapshot.storage.k8s.io/v1beta1
kind: VolumeSnapshotClass
metadata:
  name: mysnapclass
driver: csi.huawei.com
deletionPolicy: Delete
```
- If api-versions in your environment supports both v1 and v1beta1, v1 is recommended.

You can modify the parameters according to **Table 8-15**. Currently, Huawei CSI does not support user-defined parameters (**parameters**) in a VolumeSnapshotClass. Therefore, you are advised to create a VolumeSnapshotClass for all snapshots.

Table 8-15 VolumeSnapshotClass parameters

Parameter	Description	Remarks
metadata.name	User-defined name of a VolumeSnapshotClass object.	Take Kubernetes v1.22.1 as an example. The value can contain digits, lowercase letters, hyphens (-), and periods (.), and must start and end with a letter or digit.

Parameter	Description	Remarks
driver	driver identifier. This parameter is mandatory.	Set this parameter to the driver name set during Huawei CSI installation. The default driver name is **csi.huawei.com**.
deletionPolicy	Snapshot deletion policy. This parameter is mandatory. The value can be: • Delete • Retain	• If the deletion policy is **Delete**, the snapshot on the storage device will be deleted together with the VolumeSnapshotContent object. • If the deletion policy is **Retain**, the snapshot and VolumeSnapshotContent object on the storage device will be retained.

Prerequisites

Huawei CSI supports snapshots, and the volume snapshot component CRD on which its running depends has been installed. For details about the CRD, see **3.5 Checking Volume Snapshot-Dependent Components**. For details about the Kubernetes versions that support VolumeSnapshot creation, see **Table 2-3**.

Procedure

Step 1 Run the following command to create a VolumeSnapshotClass using the created VolumeSnapshotClass configuration file.

```
# kubectl create -f mysnapclass.yaml
volumesnapshotclass.snapshot.storage.k8s.io/mysnapclass created
```

Step 2 Run the following command to view the information about the created VolumeSnapshotClass.

```
# kubectl get volumesnapshotclass
NAME          DRIVER          DELETIONPOLICY   AGE
mysnapclass   csi.huawei.com   Delete          25s
```

----End

8.2.3 Configuring a VolumeSnapshot

VolumeSnapshot can be provisioned in two ways: pre-provisioning and dynamic provisioning. Currently, Huawei CSI supports only dynamic provisioning. This section describes how to dynamically provision a VolumeSnapshot using Huawei CSI.

The following is an example of the VolumeSnapshot configuration file:

- If api-versions in your environment supports v1, use the following example:

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: mysnapshot
spec:
  volumeSnapshotClassName: mysnapclass
  source:
    persistentVolumeClaimName: mypvc
```


- If api-versions in your environment supports v1beta1, use the following example:

```
apiVersion: snapshot.storage.k8s.io/v1beta1
kind: VolumeSnapshot
metadata:
  name: mysnapshot
spec:
  volumeSnapshotClassName: mysnapclass
  source:
    persistentVolumeClaimName: mypvc
```

- The api-versions information in the VolumeSnapshot must be the same as the version used for creating the VolumeSnapshotClass.

You can modify the parameters according to **Table 8-16**.

Table 8-16 VolumeSnapshot parameters

Parameter	Description	Remarks
metadata.name	User-defined name of a VolumeSnapshot object.	Take Kubernetes v1.22.1 as an example. The value can contain digits, lowercase letters, hyphens (-), and periods (.), and must start and end with a letter or digit.
spec.volumeSnapshotClassName	Name of the VolumeSnapshotClass object.	--
spec.source.persistentVolumeClaimName	Name of the source PVC object.	Name of the source PVC of the snapshot

Prerequisites

- The source PVC exists, and the backend where the PVC resides supports VolumeSnapshot creation. For details about the storage devices that support VolumeSnapshot creation, see **Table 2-5** and **Table 2-9**. For details about the Kubernetes versions that support VolumeSnapshot creation, see **Table 2-3**.
- The volume snapshot component CRD on which the running of Huawei CSI depends has been installed. For details, see **3.5 Checking Volume Snapshot-Dependent Components**.
- A VolumeSnapshotClass that uses Huawei CSI exists in the system.

Procedure

Step 1 Run the following command to create a VolumeSnapshot using the created VolumeSnapshot configuration file.

```
# kubectl create -f mysnapshot.yaml
volumesnapshot.snapshot.storage.k8s.io/mysnapshot created
```

Step 2 Run the following command to view the information about the created VolumeSnapshot.

```
# kubectl get volumesnapshot
NAME         READYTOUSE   SOURCEPVC   SOURCESNAPSHOTCONTENT   RESTORESIZE
```



```
SNAPSHOTCLASS    SNAPSHOTCONTENT                                    CREATIONTIME   AGE
mysnapshot       true      mypvc                  100Gi      mysnapclass
snapcontent-1009af0a-24c2-4435-861c-516224503f2d   <invalid>      78s
```

----End

9 Advanced Features

9.1 Configuring ALUA

Asymmetric Logical Unit Access (ALUA) is a model that supports access to multiple target ports. In the multipathing state, ALUA presents active/passive volumes to the host and provides a port access status switchover interface to switch over the working controllers for volumes. For example, when a volume of a controller fails, you can set the status of ports on the controller to **Unavailable**. After the host multipathing software that supports ALUA detects the status, it switches subsequent I/Os from the failed controller to the peer controller.

9.1.1 Configuring ALUA Using Helm

9.1.1.1 Configuring ALUA Parameters for a Huawei Enterprise Storage Backend

For details about how to configure ALUA for Huawei enterprise storage, see the host connectivity guide of the corresponding product.

The ALUA configuration may vary according to the OS. Visit **Huawei Technical Support**, enter **Host Connectivity Guide** in the search box, and click the search button. In the search result, select the host connectivity guide for the desired OS. Configure ALUA according to the actual situation and the description in the guide. Huawei CSI will apply the configuration items you set to the initiator of the host on Huawei storage.

NOTE

A node with a Pod provisioned does not proactively change ALUA information. The host ALUA configuration changes only after a Pod is provisioned again to the node.

ALUA Parameters for OceanStor V3/V5 and OceanStor Dorado V3 Series

Table 9-1 lists the ALUA parameters supported by Huawei CSI for OceanStor V3/V5 and OceanStor Dorado V3 series.

Table 9-1 ALUA parameters supported by Huawei CSI for OceanStor V3/V5 and OceanStor Dorado V3 series

Parameter	Description	Remarks
HostName	Host name rule. This parameter is mandatory. You can use a regular expression.	The host name can be obtained by running the **cat /etc/hostname** command. It can be matched by using regular expressions. When **HostName** is set to *, the configuration takes effect on hosts with any name. For details, see **Regular expression**. If the host name of a compute node matches multiple ALUA configuration options, they will be sorted based on the matching accuracy and the first ALUA configuration option will be used. For details about the sorting rules, see **Rules for Matching ALUA Configuration Items with Host Names**.
MULTIPATHTYPE	Multipathing type. This parameter is mandatory. The value can be: • **0**: Third-party multipathing is not used. • **1**: Third-party multipathing is used.	--
FAILOVERMODE	Initiator switchover mode. This parameter is conditionally mandatory. The value can be: • **0**: early-version ALUA • **1**: common ALUA • **2**: ALUA not used • **3**: special ALUA	This parameter needs to be specified only when third-party multipathing is used. Configure the initiator switchover mode by referring to the connectivity guide.

Parameter	Description	Remarks
SPECIALMODETYPE	Special mode type of the initiator. This parameter is conditionally mandatory. The value can be: • **0**: special mode 0 • **1**: special mode 1 • **2**: special mode 2 • **3**: special mode 3	This parameter needs to be specified only when the initiator switchover mode is special ALUA. Configure the special mode type of the initiator by referring to the connectivity guide.
PATHTYPE	Initiator path type. This parameter is conditionally mandatory. The value can be: • **0**: preferred path • **1**: non-preferred path	This parameter needs to be specified only when third-party multipathing is used. Configure the initiator path type by referring to the connectivity guide.

The following uses OceanStor 18500 V5 as an example to describe how to connect to Red Hat. For details about the host connectivity guide, see ***Huawei SAN Storage Host Connectivity Guide for Red Hat***.

The following ALUA configuration example is recommended in the OceanStor 18500 V5 host connectivity guide for Red Hat in non-HyperMetro storage scenarios. In this example, the OS on compute node **myhost01** in the Kubernetes cluster is RHEL 5.*x*, and that on other compute nodes is RHEL 7.*x*. According to the recommendation, the switchover mode of RHEL 5.*x* should be "ALUA not used", and that of RHEL 7.*x* should be "common ALUA".

```
storage: oceanstor-san
name: oceanstor-iscsi-155
urls:
  - https://192.168.129.155:8088
  - https://192.168.129.156:8088
pools:
  - StoragePool001
parameters:
  protocol: iscsi
  portals:
    - 192.168.128.120
    - 192.168.128.121
  ALUA:
    ^myhost01$:
      MULTIPATHTYPE: 1
      FAILOVERMODE: 2
      PATHTYPE: 0
    "*":
      MULTIPATHTYPE: 1
      FAILOVERMODE: 1
      PATHTYPE: 0
```


ALUA Parameters for OceanStor V6 and OceanStor Dorado V6 Series

Table 9-2 lists the ALUA parameters supported by Huawei CSI for OceanStor V6 and OceanStor Dorado V6 series.

NOTE

By default, the initiator host access mode of OceanStor V6 and OceanStor Dorado V6 series storage is "balanced mode". Therefore, you are not advised to configure ALUA parameters for OceanStor V6 and OceanStor Dorado V6 series storage.

Table 9-2 ALUA parameters for OceanStor V6 and OceanStor Dorado V6 series

Parameter	Description	Remarks
HostName	Host name rule. This parameter is mandatory. You can use a regular expression.	The host name can be obtained by running the **cat /etc/hostname** command. It can be matched by using regular expressions. When **HostName** is set to *, the configuration takes effect on hosts with any name. For details, see **Regular expression**. If the host name of a compute node matches multiple ALUA configuration options, they will be sorted based on the matching accuracy and the first ALUA configuration option will be used. For details about the sorting rules, see **Rules for Matching ALUA Configuration Items with Host Names**.
accessMode	Host access mode. This parameter is mandatory. The value can be: • **0**: balanced mode • **1**: asymmetric mode	The balanced mode is recommended in non-HyperMetro scenarios. Currently, Huawei CSI does not support SAN HyperMetro scenarios. Exercise caution when using the asymmetric mode.
hyperMetroPathOptimized	Whether the path of the host on the current storage array is preferred in HyperMetro scenarios. The value can be: • **1**: yes • **0**: no	This parameter needs to be specified only when the host access mode is set to asymmetric. Currently, Huawei CSI does not support SAN HyperMetro scenarios. Exercise caution when using the asymmetric mode.

The following uses OceanStor Dorado 18000 V6 as an example to describe how to connect to Red Hat. For details about the host connectivity guide, see ***OceanStor Dorado 6.x and OceanStor 6.x Host Connectivity Guide for Red Hat***.

The following ALUA configuration example is recommended in the OceanStor Dorado 18000 V6 host connectivity guide for Red Hat in non-HyperMetro storage scenarios.

```
storage: "oceanstor-san"
name: "dorado-iscsi-155"
urls:
  - "https://192.168.129.155:8088"
  - "https://192.168.129.156:8088"
pools:
  - "StoragePool001"
parameters:
  protocol: "iscsi"
  portals:
    - "192.168.128.120"
    - "192.168.128.121"
  ALUA:
    "*":
      accessMode: 0
```

Rules for Matching ALUA Configuration Items with Host Names

- If the configured host name rule exactly matches the host name of the service node, the ALUA configuration item corresponding to the host name rule is used.

  For example, the host name rule in configuration item 1 is ***** and that in configuration item 2 is **^myhost01$**. If the host name of a compute node is **myhost01**, it exactly matches configuration item 2. In this case, Huawei CSI will apply the configuration information in configuration item 2 to the storage side.

- If the configured host name rule does not exactly match the host name of the service node, the first ALUA configuration item matched by regular expressions is used.

  For example, the host name rule in configuration item 1 is **myhost0[0-9]** and that in configuration item 2 is **myhost0[5-9]**. In this case, configuration item 1 has a higher priority than configuration item 2. If the host name of a compute node is **myhost06**, both configuration items can be matched. In this case, Huawei CSI will apply the configuration information in configuration item 1 to the storage side.

9.1.1.2 Configuring ALUA Parameters for a Distributed Storage Backend

For details about how to configure ALUA for Huawei distributed storage, see the host connectivity guide of the corresponding product.

The ALUA configuration may vary according to the OS. Visit **Huawei Technical Support**, enter **Host Connectivity Guide** in the search box, and click the search button. In the search result, select the host connectivity guide for the desired OS. Configure ALUA according to the actual situation and the description in the guide. Huawei CSI will apply the configuration items you set to the initiator of the host on Huawei storage.

NOTE

A node with a Pod provisioned does not proactively change ALUA information. The host ALUA configuration changes only after a Pod is provisioned again to the node.

In non-HyperMetro scenarios of distributed storage, you are advised to set the switchover mode to "disable ALUA" (default value). This is because the storage system is in active/active mode and "enables ALUA" is meaningless. Therefore, you are not advised to configure ALUA parameters for distributed storage.

Table 9-3 lists the ALUA parameters supported by Huawei CSI for distributed storage.

Table 9-3 ALUA parameters for distributed storage

Parameter	Description	Remarks
HostName	The value of **HostName** is the host name of a worker node, for example, **HostName1** and **HostName2**.	The host name can be obtained by running the **cat /etc/hostname** command. It can be matched by using regular expressions. When **HostName** is set to *, the configuration takes effect on hosts with any name. For details, see **Regular expression**. If the host name of a compute node matches multiple ALUA configuration options, they will be sorted based on the matching accuracy and the first ALUA configuration option will be used. For details about the sorting rules, see **Rules for Matching ALUA Configuration Items with Host Names**.
switchoverMode	Switchover mode. This parameter is mandatory. The value can be: • **Disable_alua**: disables ALUA. • **Enable_alua**: enables ALUA.	In non-HyperMetro scenario, you are advised to set the switchover mode to "disable ALUA". This is because the storage system is in active/active mode and "enables ALUA" is meaningless. Currently, Huawei CSI does not support SAN HyperMetro scenarios. Exercise caution when enabling ALUA.

Parameter	Description	Remarks
pathType	Path type. This parameter is conditionally mandatory. The value can be: • **optimal_path**: preferred path • **non_optimal_path**: non-preferred path	This parameter is mandatory when the switchover mode is set to "enables ALUA".

Rules for Matching ALUA Configuration Items with Host Names

- If the configured host name rule exactly matches the host name of the service node, the ALUA configuration item corresponding to the host name rule is used.

  For example, the host name rule in configuration item 1 is ***** and that in configuration item 2 is **^myhost01$**. If the host name of a compute node is **myhost01**, it exactly matches configuration item 2. In this case, Huawei CSI will apply the configuration information in configuration item 2 to the storage side.

- If the configured host name rule does not exactly match the host name of the service node, the first ALUA configuration item matched by regular expressions is used.

  For example, the host name rule in configuration item 1 is **myhost0[0-9]** and that in configuration item 2 is **myhost0[5-9]**. In this case, configuration item 1 has a higher priority than configuration item 2. If the host name of a compute node is **myhost06**, both configuration items can be matched. In this case, Huawei CSI will apply the configuration information in configuration item 1 to the storage side.

9.2 Configuring Storage Topology Awareness

In the Kubernetes cluster, resources can be scheduled and provisioned based on the topology labels of nodes and the topology capabilities supported by storage backends.

Prerequisites

You need to configure topology labels on worker nodes in the cluster. The method is as follows:

1. Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.
2. Run the **kubectl get node** command to view information about worker nodes in the current cluster.

```
# kubectl get node
NAME     STATUS   ROLES                  AGE   VERSION
node01   Ready    controlplane,etcd,worker   42d   v1.22.3
node02   Ready    worker                 42d   v1.22.3
node03   Ready    worker                 42d   v1.22.3
```


3. Run the **kubectl label node** *<nodename>* **topology.kubernetes.io/***<key>*=*<value>* command to configure a topology label for a worker node. In the preceding command, *<nodename>* indicates the name of a worker node. For details about the **key** and **value** parameters, see **Table 9-4**.

```
# kubectl label node node01 topology.kubernetes.io/zone=ChengDu
node/node01 labeled
```

Table 9-4 Parameter description

Parameter	Description	Remarks
<key>	Unique identifier of a topology label.	The value can be **zone**, **region**, or **protocol.***<protocol>*. *<protocol>* can be set to **iscsi**, **nfs**, **fc**, or **roce**.
<value>	Value of a topology label.	If **key** is set to **zone** or **region**, **value** is a user-defined parameter. If **key** is set to **protocol.***<protocol>*, **value** is fixed to **csi.huawei.com**.

NOTE

- A topology label must start with **topology.kubernetes.io**. Topology label examples:
  - Example 1: **topology.kubernetes.io/region=China-west**
  - Example 2: **topology.kubernetes.io/zone=ChengDu**
  - Example 3: **topology.kubernetes.io/protocol.iscsi=csi.huawei.com**
  - Example 4: **topology.kubernetes.io/protocol.fc=csi.huawei.com**
- A key in a topology label on a node can have only one value.
- If multiple protocols are configured in a topology label on a node, when you select a backend, the backend needs to meet only one of the protocols.
- If both the region and the zone are configured in a topology label on a node, when you select a backend, the backend must meet both of them.

4. Run the **kubectl get nodes -o=jsonpath='{range .items[*]}[{.metadata.name}, {.metadata.labels}]{"\n"}{end}' | grep --color "topology.kubernetes.io"** command to view the label information about all worker nodes in the current cluster.

```
# kubectl get nodes -o=jsonpath='{range .items[*]}[{.metadata.name}, {.metadata.labels}]{"\n"}{end}'
| grep --color "topology.kubernetes.io"
[node01, {"beta.kubernetes.io/arch":"amd64","beta.kubernetes.io/os":"linux","kubernetes.io/
arch":"amd64","kubernetes.io/hostname":"node01","kubernetes.io/os":"linux","node-role.kubernetes.io/
controlplane":"true","node-role.kubernetes.io/etcd":"true","node-role.kubernetes.io/
worker":"true","topology.kubernetes.io/zone":"ChengDu"}]
```

9.2.1 Configuring Storage Topology Awareness Using Helm

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Go to the directory where the Helm project is located. If the previous Helm project cannot be found, copy the **helm** directory in the component package to any directory on the master node. For details about the component package path, see **Table 3-1**.

Step 3 Go to the backend service configuration directory **/examples/backend/** and back up the **backend.yaml** file.

```
# cp backend.yaml backend.yaml.bak
```

Step 4 Run the **vi** *backend.yaml* command to open the file and configure topology awareness as required. The following is an example. After the modification is complete, press **Esc** and enter **:wq!** to save the modification.

```
storage: "oceanstor-san"
name: "dorado-iscsi-155"
namespace: "huawei-csi"
urls:
  - "https://192.168.129.155:8088"
pools:
  - "StoragePool001"
parameters:
  protocol: "iscsi"
  portals:
    - "10.10.30.20"
    - "10.10.30.21"
supportedTopologies:
  - { "topology.kubernetes.io/region": "China-west", "topology.kubernetes.io/zone": "ChengDu" }
  - { "topology.kubernetes.io/region": "China-south","topology.kubernetes.io/zone": "ShenZhen" }
maxClientThreads: "30"
```

Step 5 Run the following command to delete the storage backend to be modified. In the command, **dorado-iscsi-155** indicates the storage backend name.

```
# oceanctl delete backend dorado-iscsi-155 -n huawei-csi
backend/dorado-iscsi-155 deleted
```

Step 6 Run the following command to create a storage backend.

```
# oceanctl create backend -f ../examples/backend/backend.yaml
Please enter this backend user name:admin
Please enter this backend password:

backend/dorado-iscsi-155 created
```

Step 7 Run the **vi** *StorageClass.yaml* command to modify the .yaml file. Press **I** or **Insert** to enter the editing mode and add related parameters in the .yaml file. For details about the parameters, see **Table 9-5**. After the modification is complete, press **Esc** and enter **:wq!** to save the modification.

Add the following configuration items to the *StorageClass.yaml* file.

- Example 1: Configure zone and region information in the StorageClass.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: example-storageclass
provisioner: csi.huawei.com
parameters:
  volumeType: lun
  allocType: thin
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
- matchLabelExpressions:
  - key: topology.kubernetes.io/zone
    values:
    - ChengDu
  - key: topology.kubernetes.io/region
```



```
  values:
  - China-west
```

- Example 2: Configure protocol information in the StorageClass.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: protocol-example-storageclass
provisioner: csi.huawei.com
parameters:
  volumeType: lun
  allocType: thin
volumeBindingMode: WaitForFirstConsumer
allowedTopologies:
- matchLabelExpressions:
  - key: topology.kubernetes.io/protocol.iscsi
    values:
    - csi.huawei.com
```

Table 9-5 Parameter description

Parameter	Description	Remarks
volumeBindingMode	PersistentVolume binding mode, used to control the time when PersistentVolume resources are dynamically allocated and bound.	You can set this parameter to **WaitForFirstConsumer** or **Immediate**. **WaitForFirstConsumer**: indicates that the binding and allocation of the PersistentVolume are delayed until a Pod that uses the PVC is created. **Immediate**: The PersistentVolume is bound and allocated immediately after a PVC is created.
allowedTopologies.matchLabelExpressions	Topology information label, which is used to filter CSI backends and Kubernetes nodes. If the matching fails, PVCs or Pods cannot be created. Both **key** and **value** must be configured in a fixed format.	**key**: This parameter can be set to **topology.kubernetes.io/zone** or **topology.kubernetes.io/region**. **topology.kubernetes.io/protocol.***<protocol>*: *<protocol>* indicates the protocol type and can be **iscsi**, **fc**, or **nfs**.
		value: If **key** is **topology.kubernetes.io/zone** or **topology.kubernetes.io/region**, **value** must be the same as the **topology label set in the prerequisites**. If **key** is **topology.kubernetes.io/protocol.***<protocol>*, **value** is fixed to **csi.huawei.com**.

Step 8 Run the following command to create a StorageClass based on the .yaml file.

```
# kubectl create -f StorgeClass.yaml
```


Step 9 Use the StorageClass to create a PVC with the topology capability. For details, see **8.1.1.1.2 Configuring a PVC**.

----End

10 Common Operations

10.1 Collecting Logs

Performing Check Before Collection

Step 1 Use a remote access tool, such as PuTTY, to log in to the node where the oceanctl tool is installed in the Kubernetes cluster through the management IP address.

Step 2 Run the **oceanctl version** command. The displayed version is **v4.2.0**.

```
$ oceanctl version
+------------------+
| OCEANCTL VERSION |
+------------------+
| v4.2.0           |
+------------------+
```

Step 3 Run the **oceanctl --help** command. The following information is displayed.

```
$ oceanctl --help
A CLI tool for Ocean Storage in Kubernetes

Usage:
  oceanctl [command]

Available Commands:
  collect     collect messages in Kubernetes
  create      Create a resource to Ocean Storage in Kubernetes
  delete      Delete one or more resources from Ocean Storage in Kubernetes
  get         Get one or more resources from Ocean Storage in Kubernetes
  help        Help about any command
  update      Update a resource for Ocean Storage in Kubernetes
  version     Print the version of oceanctl
```



```
Flags:
  -h, --help   help for oceanctl

Use "oceanctl [command] --help" for more information about a command.
```

Step 4 Run the **kubectl get deploy -n** *${NAMESPACE}* command to check whether a Pod is started properly. In the preceding command, *${NAMESPACE}* indicates the namespace for installing CSI. **huawei-csi** is used as an example.

```
# kubectl get deploy -n huawei-csi
NAME                 READY   UP-TO-DATE   AVAILABLE   AGE
huawei-csi-controller   1/1     1            1           21h
```

----End

Collecting All Logs in the CSI Namespace Using oceanctl

Step 1 Use a remote access tool, such as PuTTY, to log in to the node checked in **Performing Check Before Collection** through the management IP address.

Step 2 Run the **oceanctl collect logs -n <namespace> -a** command to collect CSI logs of all nodes where CSI containers reside in the cluster.

```
# oceanctl collect logs -n huawei-csi -a
node[node-1] Collection Progress:[++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++] 2/2 Pods
```

Step 3 Check the log package generated in the **/tmp** directory. You can run the **unzip** *${zip_name}* **-d collect_logs** command to decompress the log package. In the preceding command, *${zip_name}* indicates the package name.

```
# date
Wed Sep 20 02:49:24 EDT 2023

# ls
huawei-csi-2023-09-20-02:48:22-all.zip
```

----End

Collecting the Log of a Single CSI Node Using oceanctl

Step 1 Use a remote access tool, such as PuTTY, to log in to the node checked in **Performing Check Before Collection** through the management IP address.

Step 2 Run the **oceanctl collect logs -n** *<namespace>* **-N** *<nodeName>* command to collect CSI logs of all nodes where CSI containers reside in the cluster.

```
# oceanctl collect logs -n huawei-csi -N node-1
node[node-1] Collection Progress:[++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++] 2/2 Pods
```

Step 3 Check the log package generated in the **/tmp** directory. You can run the **unzip** *${zip_name}* **-d collect_logs** command to decompress the log package. In the preceding command, *${zip_name}* indicates the package name.

```
# date
Thu Sep 21 04:08:47 EDT 2023

# ls
huawei-csi-2023-09-21-04:05:15-node-1.zip
```

----End

10.2 Updating the huawei-csi-controller Service

Perform this operation when you need to update the huawei-csi-controller service, for example, adding the snapshot or the capacity expansion function.

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Back up the **values.yaml** file used during CSI installation. You can run the **helm get values** *helm-huawei-csi* **-n** *huawei-csi* **-a > values.yaml.bak** command to back up the file. In the preceding command, *helm-huawei-csi* indicates the Helm chart name defined during installation, and *huawei-csi* indicates the Helm chart namespace defined during installation.

```
# helm get values helm-huawei-csi -n huawei-csi  -a  > values.yaml.bak
```

Step 3 Go to the **/helm/esdk** directory, run the **vi** *values.yaml* command to open the file and modify controller parameters. The following is an example. After the modification is complete, press **Esc** and enter **:wq!** to save the modification. For details about the component package path, see **Table 3-1**.

```
kubernetes:
  namespace: huawei-csi

images:
  # Images provided by Huawei
  huaweiCSIService: huawei-csi:4.2.0
  storageBackendSidecar: storage-backend-sidecar:4.2.0
  storageBackendController: storage-backend-controller:4.2.0

  # CSI-related sidecar images provided by the Kubernetes community.
  # These must match the appropriate Kubernetes version.
  sidecar:
    attacher: k8s.gcr.io/sig-storage/csi-attacher:v3.4.0
    provisioner: k8s.gcr.io/sig-storage/csi-provisioner:v3.0.0
    resizer: k8s.gcr.io/sig-storage/csi-resizer:v1.4.0
    registrar: k8s.gcr.io/sig-storage/csi-node-driver-registrar:v2.3.0
    livenessProbe: k8s.gcr.io/sig-storage/livenessprobe:v2.5.0
    snapshotter: k8s.gcr.io/sig-storage/csi-snapshotter:v4.2.1
    snapshotController: k8s.gcr.io/sig-storage/snapshot-controller:v4.2.1

  # The image name and tag for the Huawei CSI Service container
  # Replace the appropriate tag name
  huaweiCSIService: huawei-csi:4.2.0


# The CSI driver parameter configuration
csiDriver:
  # Driver name, it is strongly recommended not to modify this parameter
  # The CCE platform needs to modify this parameter, e.g. csi.oceanstor.com
  driverName: csi.huawei.com
  # Endpoint, it is strongly recommended not to modify this parameter
  endpoint: /csi/csi.sock
  # DR Endpoint, it is strongly recommended not to modify this parameter
  drEndpoint: /csi/dr-csi.sock
  # Maximum number of concurrent disk scans or detaches, support 1~10
  connectorThreads: 4
  # Flag to enable or disable volume multipath access, support [true, false]
  volumeUseMultipath: true
  # Multipath software used by fc/iscsi. support [DM-multipath, HW-UltraPath, HW-UltraPath-NVMe]
  scsiMultipathType: DM-multipath
```



```
  # Multipath software used by roce/fc-nvme. only support [HW-UltraPath-NVMe]
  nvmeMultipathType: HW-UltraPath-NVMe
  # Timeout interval for waiting for multipath aggregation when DM-multipath is used on the host. support
1~600
  scanVolumeTimeout: 3
  # Interval for updating backend capabilities. support 60~600
  backendUpdateInterval: 60
  # Huawei-csi-controller log configuration
  controllerLogging:
    # Log record type, support [file, console]
    module: file
    # Log Level, support [debug, info, warning, error, fatal]
    level: info
    # Directory for storing logs
    fileDir: /var/log/huawei
    # Size of a single log file
    fileSize: 20M
    # Maximum number of log files that can be backed up.
    maxBackups: 9
  # Huawei-csi-node log configuration
  nodeLogging:
    # Log record type, support [file, console]
    module: file
    # Log Level, support [debug, info, warning, error, fatal]
    level: info
    # Directory for storing logs
    fileDir: /var/log/huawei
    # Size of a single log file
    fileSize: 20M
    # Maximum number of log files that can be backed up.
    maxBackups: 9

controller:
  # controllerCount: Define the number of huawei-csi controller
  # Allowed values: n, where n > 0
  # Default value: 1
  # Recommended value: 2
  controllerCount: 1

  # volumeNamePrefix: Define a prefix that is prepended to volumes.
  # THIS MUST BE ALL LOWER CASE.
  # Default value: pvc
  # Examples: "volumes", "vol"
  volumeNamePrefix: pvc

  # Port used by the webhook service. The default port is 4433.
  # You can change the port to another port that is not occupied.
  webhookPort: 4433

  snapshot:
    # enabled: Enable/Disable volume snapshot feature
    # If the Kubernetes version is lower than 1.17, set this parameter to false.
    # Allowed values:
    #   true: enable volume snapshot feature(install snapshotter sidecar)
    #   false: disable volume snapshot feature(do not install snapshotter sidecar)
    # Default value: None
    enabled: true

  resizer:
    # enabled: Enable/Disable volume expansion feature
    # Allowed values:
    #   true: enable volume expansion feature(install resizer sidecar)
    #   false: disable volume snapshot feature(do not install resizer sidecar)
    # Default value: None
    enabled: true

  # nodeSelector: Define node selection constraints for controller pods.
  # For the pod to be eligible to run on a node, the node must have each
  # of the indicated key-value pairs as labels.
```



```
# Leave as blank to consider all nodes
# Allowed values: map of key-value pairs
# Default value: None
nodeSelector:
# Uncomment if nodes you wish to use have the node-role.kubernetes.io/master taint
#  node-role.kubernetes.io/master: ""
# Uncomment if nodes you wish to use have the node-role.kubernetes.io/control-plane taint
#  node-role.kubernetes.io/control-plane: ""

# tolerations: Define tolerations that would be applied to controller deployment
# Leave as blank to install controller on worker nodes
# Allowed values: map of key-value pairs
# Default value: None
tolerations:
# Uncomment if nodes you wish to use have the node-role.kubernetes.io/master taint
#  - key: "node-role.kubernetes.io/master"
# Uncomment if nodes you wish to use have the node-role.kubernetes.io/control-plane taint
#  - key: "node-role.kubernetes.io/control-plane"
#    operator: "Exists"
#    effect: "NoSchedule"
```

Step 4 Run the **helm upgrade** *helm-huawei-csi* **./ -n** *huawei-csi* command to upgrade the Helm chart.

In the preceding command, *helm-huawei-csi* indicates the name of the chart to be upgraded, **./** indicates the Helm project in the current directory, and *huawei-csi* indicates the namespace where the chart is located.

```
# helm upgrade helm-huawei-csi ./ -n huawei-csi
Release "helm-huawei-csi" has been upgraded. Happy Helming!
NAME: helm-huawei-csi
LAST DEPLOYED: Thu Jun  9 07:58:15 2022
NAMESPACE: huawei-csi
STATUS: deployed
REVISION: 2
TEST SUITE: None
```

----End

10.3 Updating the huawei-csi-node Service

Perform this operation when you need to update the huawei-csi-node service.

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Back up the **values.yaml** file used during CSI installation. You can run the **helm get values** *helm-huawei-csi* **-n** *huawei-csi* **-a > values.yaml.bak** command to back up the file. In the preceding command, *helm-huawei-csi* indicates the Helm chart name defined during installation, and *huawei-csi* indicates the Helm chart namespace defined during installation.

```
# helm get values helm-huawei-csi -n huawei-csi  -a  > values.yaml.bak
```

Step 3 Go to the **/helm/esdk** directory, run the **vi** *values.yaml* command to open the file and modify node parameters. The following is an example. After the modification is complete, press **Esc** and enter **:wq!** to save the modification. For details about the component package path, see **Table 3-1**.

```
kubernetes:
  namespace: huawei-csi
```



```
images:
  # Images provided by Huawei
  huaweiCSIService: huawei-csi:4.2.0
  storageBackendSidecar: storage-backend-sidecar:4.2.0
  storageBackendController: storage-backend-controller:4.2.0

  # CSI-related sidecar images provided by the Kubernetes community.
  # These must match the appropriate Kubernetes version.
  sidecar:
    attacher: k8s.gcr.io/sig-storage/csi-attacher:v3.4.0
    provisioner: k8s.gcr.io/sig-storage/csi-provisioner:v3.0.0
    resizer: k8s.gcr.io/sig-storage/csi-resizer:v1.4.0
    registrar: k8s.gcr.io/sig-storage/csi-node-driver-registrar:v2.3.0
    livenessProbe: k8s.gcr.io/sig-storage/livenessprobe:v2.5.0
    snapshotter: k8s.gcr.io/sig-storage/csi-snapshotter:v4.2.1
    snapshotController: k8s.gcr.io/sig-storage/snapshot-controller:v4.2.1

  # The image name and tag for the Huawei CSI Service container
  # Replace the appropriate tag name
  huaweiCSIService: huawei-csi:4.2.0


# The CSI driver parameter configuration
csiDriver:
  # Driver name, it is strongly recommended not to modify this parameter
  # The CCE platform needs to modify this parameter, e.g. csi.oceanstor.com
  driverName: csi.huawei.com
  # Endpoint, it is strongly recommended not to modify this parameter
  endpoint: /csi/csi.sock
  # DR Endpoint, it is strongly recommended not to modify this parameter
  drEndpoint: /csi/dr-csi.sock
  # Maximum number of concurrent disk scans or detaches, support 1~10
  connectorThreads: 4
  # Flag to enable or disable volume multipath access, support [true, false]
  volumeUseMultipath: true
  # Multipath software used by fc/iscsi. support [DM-multipath, HW-UltraPath, HW-UltraPath-NVMe]
  scsiMultipathType: DM-multipath
  # Multipath software used by roce/fc-nvme. only support [HW-UltraPath-NVMe]
  nvmeMultipathType: HW-UltraPath-NVMe
  # Timeout interval for waiting for multipath aggregation when DM-multipath is used on the host. support
1~600
  scanVolumeTimeout: 3
  # Interval for updating backend capabilities. support 60~600
  backendUpdateInterval: 60
  # Huawei-csi-controller log configuration
  controllerLogging:
    # Log record type, support [file, console]
    module: file
    # Log Level, support [debug, info, warning, error, fatal]
    level: info
    # Directory for storing logs
    fileDir: /var/log/huawei
    # Size of a single log file
    fileSize: 20M
    # Maximum number of log files that can be backed up.
    maxBackups: 9
  # Huawei-csi-node log configuration
  nodeLogging:
    # Log record type, support [file, console]
    module: file
    # Log Level, support [debug, info, warning, error, fatal]
    level: info
    # Directory for storing logs
    fileDir: /var/log/huawei
    # Size of a single log file
    fileSize: 20M
    # Maximum number of log files that can be backed up.
    maxBackups: 9
```



```
node:
  # maxVolumesPerNode: Defines the maximum number of volumes that can be used by a node.
  # Examples: 100
  # Uncomment if you want to limit the number of volumes that can be used in a Node.
  # maxVolumesPerNode: 100

  # nodeSelector: Define node selection constraints for node pods.
  # For the pod to be eligible to run on a node, the node must have each
  # of the indicated key-value pairs as labels.
  # Leave as blank to consider all nodes
  # Allowed values: map of key-value pairs
  # Default value: None
  nodeSelector:
  # Uncomment if nodes you wish to use have the node-role.kubernetes.io/master taint
  #  node-role.kubernetes.io/master: ""
  # Uncomment if nodes you wish to use have the node-role.kubernetes.io/control-plane taint
  #  node-role.kubernetes.io/control-plane: ""

  # tolerations: Define tolerations that would be applied to node daemonset
  # Add/Remove tolerations as per requirement
  # Leave as blank if you wish to not apply any tolerations
  # Allowed values: map of key-value pairs
  # Default value: None
  tolerations:
    - key: "node.kubernetes.io/memory-pressure"
      operator: "Exists"
      effect: "NoExecute"
    - key: "node.kubernetes.io/disk-pressure"
      operator: "Exists"
      effect: "NoExecute"
    - key: "node.kubernetes.io/network-unavailable"
      operator: "Exists"
      effect: "NoExecute"
#    - key: "node-role.kubernetes.io/control-plane"
#      operator: "Exists"
#      effect: "NoSchedule"
#    - key: "node-role.kubernetes.io/master"
#      operator: "Exists"
#      effect: "NoSchedule"
```

Step 4 Run the **helm upgrade** *helm-huawei-csi* **./ -n** *huawei-csi* command to upgrade the Helm chart.

In the preceding command, *helm-huawei-csi* indicates the name of the chart to be upgraded, **./** indicates the Helm project in the current directory, and *huawei-csi* indicates the namespace where the chart is located.

```
# helm upgrade helm-huawei-csi ./ -n huawei-csi
Release "helm-huawei-csi" has been upgraded. Happy Helming!
NAME: helm-huawei-csi
LAST DEPLOYED: Thu Jun  9 07:58:15 2022
NAMESPACE: huawei-csi
STATUS: deployed
REVISION: 2
TEST SUITE: None
```

----End

10.4 Modifying the Log Output Mode

huawei-csi supports two log output modes: **file** and **console**. **file** indicates that logs are output to the fixed directory (**/var/log/huawei**), and **console** indicates that logs are output to the standard directory of the container. You can set the log output mode as required. The default mode is **file**.

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Go to the directory where the Helm project is located. If the previous Helm project cannot be found, copy the **helm** directory in the component package to any directory on the master node. For details about the component package path, see **Table 3-1**.

Step 3 Back up the **values.yaml** file used during CSI installation. If the **values.yaml** file used during the last installation cannot be found, run the **helm get values** *helm-huawei-csi* **-n** *huawei-csi* **-a** command to query the file.

```
# cp values.yaml values.yaml.bak
```

Step 4 Run the **vi** *values.yaml* command to open the file and modify controller or node log parameters. The following is an example. After the modification is complete, press **Esc** and enter **:wq!** to save the modification.

```
# The CSI driver parameter configuration
csiDriver:
  # Driver name, it is strongly recommended not to modify this parameter
  # The CCE platform needs to modify this parameter, e.g. csi.oceanstor.com
  driverName: csi.huawei.com
  # Endpoint, it is strongly recommended not to modify this parameter
  endpoint: /csi/csi.sock
  # DR Endpoint, it is strongly recommended not to modify this parameter
  drEndpoint: /csi/dr-csi.sock
  # Maximum number of concurrent disk scans or detaches, support 1~10
  connectorThreads: 4
  # Flag to enable or disable volume multipath access, support [true, false]
  volumeUseMultipath: true
  # Multipath software used by fc/iscsi. support [DM-multipath, HW-UltraPath, HW-UltraPath-NVMe]
  scsiMultipathType: DM-multipath
  # Multipath software used by roce/fc-nvme. only support [HW-UltraPath-NVMe]
  nvmeMultipathType: HW-UltraPath-NVMe
  # Timeout interval for waiting for multipath aggregation when DM-multipath is used on the host. support
1~600
  scanVolumeTimeout: 3
  # Interval for updating backend capabilities. support 60~600
  backendUpdateInterval: 60
  # Huawei-csi-controller log configuration
  controllerLogging:
    # Log record type, support [file, console]
    module: file
    # Log Level, support [debug, info, warning, error, fatal]
    level: info
    # Directory for storing logs
    fileDir: /var/log/huawei
    # Size of a single log file
    fileSize: 20M
    # Maximum number of log files that can be backed up.
    maxBackups: 9
  # Huawei-csi-node log configuration
  nodeLogging:
    # Log record type, support [file, console]
    module: file
    # Log Level, support [debug, info, warning, error, fatal]
    level: info
    # Directory for storing logs
    fileDir: /var/log/huawei
    # Size of a single log file
    fileSize: 20M
    # Maximum number of log files that can be backed up.
    maxBackups: 9
```


Step 5 Run the **helm upgrade** *helm-huawei-csi* **./ -n** *huawei-csi* command to upgrade the Helm chart.

In the preceding command, *helm-huawei-csi* indicates the name of the chart to be upgraded, **./** indicates the Helm project in the current directory, and *huawei-csi* indicates the namespace where the chart is located.

```
# helm upgrade helm-huawei-csi ./ -n huawei-csi
Release "helm-huawei-csi" has been upgraded. Happy Helming!
NAME: helm-huawei-csi
LAST DEPLOYED: Thu Jun  9 07:58:15 2022
NAMESPACE: huawei-csi
STATUS: deployed
REVISION: 2
TEST SUITE: None
```

----End

10.5 Enabling the ReadWriteOncePod Feature Gate

The ReadWriteOnce access mode is the fourth access mode introduced by Kubernetes v1.22 for PVs and PVCs. If you create a Pod using a PVC in ReadWriteOncePod access mode, Kubernetes ensures that the Pod is the only Pod in the cluster that can read or write the PVC.

The ReadWriteOncePod access mode is an alpha feature in Kubernetes v1.22/1.23/1.24. Therefore, you need to enable the ReadWriteOncePod feature in **feature-gates** of kube-apiserver, kube-scheduler, and kubelet before using the access mode.

NOTE

Currently, the CCE/CCE Agile platform does not support the ReadWriteOncePod feature gate.

Procedure

Step 1 Enable the ReadWriteOncePod feature gate for kube-apiserver.

1. Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.
2. Run the **vi /etc/kubernetes/manifests/kube-apiserver.yaml** command, press **I** or **Insert** to enter the editing mode, and add **--feature-gates=ReadWriteOncePod=true** to the kube-apiserver container. After the modification is complete, press **Esc** and enter **:wq!** to save the modification.

```
...
spec:
  containers:
  - command:
    - kube-apiserver
    - --feature-gates=ReadWriteOncePod=true
    ...
```

NOTE

After the editing is complete, Kubernetes will automatically apply the updates.

Step 2 Enable the ReadWriteOncePod feature gate for kube-scheduler.

1. Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

2. Run the **vi /etc/kubernetes/manifests/kube-scheduler.yaml** command, press **I** or **Insert** to enter the editing mode, and add **--feature-gates=ReadWriteOncePod=true** to the kube-scheduler container. After the modification is complete, press **Esc** and enter **:wq!** to save the modification.

```
...
spec:
  containers:
  - command:
    - kube-scheduler
    - --feature-gates=ReadWriteOncePod=true
    ...
```

NOTE

After the editing is complete, Kubernetes will automatically apply the updates.

Step 3 Enable the ReadWriteOncePod feature gate for kubelet.

NOTICE

The dynamic Kubelet configuration function is not used since v1.22 and deleted in v1.24. Therefore, you need to perform the following operations on kubelet on each worker node in the cluster.

1. Use a remote access tool, such as PuTTY, to log in to any worker node in the Kubernetes cluster through the management IP address.
2. Run the **vi /var/lib/kubelet/config.yaml** command, press **I** or **Insert** to enter the editing state, and add **ReadWriteOncePod: true** to the **featureGates** field of the KubeletConfiguration object. If the **featureGates** field does not exist, add it at the same time. After the modification is complete, press **Esc** and enter **:wq!** to save the modification.

```
apiVersion: kubelet.config.k8s.io/v1beta1
featureGates:
  ReadWriteOncePod: true
  ...
```

NOTE

The default path of the kubelet configuration file is **/var/lib/kubelet/config.yaml**. Enter the path based on site requirements.

3. After the configuration is complete, run the **systemctl restart kubelet** command to restart kubelet.

----End

10.6 Configuring Access to the Kubernetes Cluster as a Non-root User

Procedure

Step 1 Copy the authentication file of the Kubernetes cluster and modify **/etc/kubernetes/admin.conf** to be the actual authentication file.

```
$ mkdir -p $HOME/.kube
$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
```


Step 2 Change the user and user group of the authentication file.

```
$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

Step 3 Configure the **KUBECONFIG** environment variable of the current user. The following uses Ubuntu 20.04 as an example.

```
$ echo "export KUBECONFIG=$HOME/.kube/config" >> ~/.bashrc
$ source ~/.bashrc
```

----End

11 FAQ

11.1 How Do I View Huawei CSI Logs?

Viewing Logs of the huawei-csi-controller Service

Step 1 Run the following command to obtain the node where huawei-csi-controller is located.

```
# kubectl get pod -A -o wide | grep huawei
huawei-csi   huawei-csi-controller-695b84b4d8-tg64l   9/9   Running   0   14s   <host1-ip>   <host1-name>   <none>   <none>
```

Step 2 Use a remote access tool, such as PuTTY, to log in to the huawei-csi-controller node in the Kubernetes cluster through the management IP address.

Step 3 Run the **cd /var/log/huawei** command to go to the log directory.

```
# cd /var/log/huawei
```

Step 4 Run the following command to view the customized output logs of the container.

```
# vi huawei-csi-controller
```

Step 5 Run the **cd /var/log/containers** command to go to the container directory.

```
# cd /var/log/containers
```

Step 6 Run the following command to view the standard output logs of the container.

```
# vi huawei-csi-controller-<name>_huawei-csi_huawei-csi-driver-<contrainer-id>.log
```

----End

Viewing Logs of the huawei-csi-node Service

Step 1 Run the following command to obtain the node where huawei-csi-node is located.

```
# kubectl get pod -A -o wide | grep huawei
huawei-csi   huawei-csi-node-g6f7z   3/3   Running   0   14s   <host2-ip>   <host2-name>   <none>   <none>
```

Step 2 Use a remote access tool, such as PuTTY, to log in to the huawei-csi-node node in the Kubernetes cluster through the management IP address.

Step 3 Run the **cd /var/log/huawei** command to go to the log directory.

```
# cd /var/log/huawei
```

Step 4 Run the following command to view the customized output logs of the container.

```
# vi huawei-csi-node
```

Step 5 Run the **cd /var/log/containers** command to go to the container directory.

```
# cd /var/log/containers
```

Step 6 Run the following command to view the standard output logs of the container.

```
# vi huawei-csi-node-<name>_huawei-csi_huawei-csi-driver-<contrainer-id>.log
```

----End

11.2 Failed to Create a Pod Because the iscsi_tcp Service Is Not Started Properly When the Kubernetes Platform Is Set Up for the First Time

Symptom

When you create a Pod, error **Cannot connect ISCSI portal *.*.*.*: libkmod: kmod_module_insert_module: could not find module by name='iscsi_tcp'** is reported in the **/var/log/huawei-csi-node** log.

Root Cause Analysis

The iscsi_tcp service may be stopped after the Kubernetes platform is set up and the iscsi service is installed. You can run the **lsmod | grep iscsi | grep iscsi_tcp** command to check whether the service is stopped.

```
# lsmod | grep iscsi | grep iscsi_tcp
iscsi_tcp              18333  6
libiscsi_tcp           25146  1 iscsi_tcp
libiscsi               57233  2 libiscsi_tcp,iscsi_tcp
scsi_transport_iscsi   99909  3 iscsi_tcp,libiscsi
```

Solution or Workaround

Run the following command to manually load the iscsi_tcp service.

```
# modprobe iscsi_tcp
# lsmod | grep iscsi | grep iscsi_tcp
iscsi_tcp              18333  6
libiscsi_tcp           25146  1 iscsi_tcp
```

11.3 Failed to Start the huawei-csi-node Service with Error Message "/var/lib/iscsi is not a directory" Reported

Symptom

The huawei-csi-node service cannot be started. When you run the **kubectl describe daemonset huawei-csi-node -n huawei-csi** command, error message "/var/lib/iscsi is not a directory" is reported.

Root Cause Analysis

The **/var/lib/iscsi** directory does not exist in the huawei-csi-node container.

Solution or Workaround

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Go to the directory where the Helm project is located. If the previous Helm project cannot be found, copy the **helm** directory in the component package to any directory on the master node. For details about the component package path, see **Table 3-1**.

Step 3 Go to the **templates** directory and find the **huawei-csi-node.yaml** file.

```
# cd /templates
```

Step 4 Run the following command to set **path** in **huawei-csi-node.yaml** > **volumes** > **iscsi-dir** > **hostPath** to **/var/lib/iscsi**, save the file, and exit.

```
# vi huawei-csi-node.yaml
```

Step 5 Run the **helm upgrade** *helm-huawei-csi* ./ **-n** *huawei-csi* **-f** *values.yaml* command to upgrade the Helm chart. The upgrade command will update the Deployment, DaemonSet, and RBAC resources. In the preceding command, *helm-huawei-csi* indicates the custom chart name and *huawei-csi* indicates the custom namespace.

```
# helm upgrade helm-huawei-csi ./  -n huawei-csi
Release "helm-huawei-csi" has been upgraded. Happy Helming!
NAME: helm-huawei-csi
LAST DEPLOYED: Thu Jun  9 07:58:15 2022
NAMESPACE: huawei-csi
STATUS: deployed
REVISION: 2
TEST SUITE: None
```

----End

11.4 After a Worker Node in the Cluster Breaks Down and Recovers, Pod Failover Is Complete but the Source Host Where the Pod Resides Has Residual Drive Letters

Symptom

A Pod is running on worker node A, and an external block device is mounted to the Pod through CSI. After worker node A is powered off abnormally, the Kubernetes platform detects that the node is faulty and switches the Pod to worker node B. After worker node A recovers, the drive letters on worker node A change from normal to faulty.

Environment Configuration

Kubernetes version: 1.18 or later

Storage type: block storage

Root Cause Analysis

After worker node A recovers, Kubernetes initiates an unmapping operation on the storage, but does not initiate a drive letter removal operation on the host. After Kubernetes completes the unmapping, residual drive letters exist on worker node A.

Solution or Workaround

Currently, you can only manually clear the residual drive letters on the host. Alternatively, restart the host again and use the disk scanning mechanism during the host restart to clear the residual drive letters. The specific method is as follows:

Step 1 Check the residual drive letters on the host.

1. Run the **multipath -ll** command to check whether a DM multipathing device with abnormal multipathing status exists.

   As shown in the following figure, the path status is **failed faulty running**, the corresponding DM multipathing device is **dm-12**, and the associated SCSI disks are **sdi** and **sdj**. If multiple paths are configured, multiple SCSI disks exist. Record these SCSI disks.

   ```
   # multipath -ll
   mpathb (3618cf24100f8f457014a764c000001f6) dm-12 HUAWEI  ,XSG1
   size=100G features='0' hwhandler='0' wp=rw
   `-+- policy='service-time 0' prio=-1 status=active
     |- 39:0:0:1        sdi 8:48  failed faulty running
     `- 38:0:0:1        sdj 8:64  failed faulty running
   ```

   - If yes, go to **Step 1.2**.
   - If no, no further action is required.

2. Check whether the residual DM multipathing device is readable.

   Run the **dd if=/dev/*dm-xx* of=/dev/null count=1 bs=1M iflag=direct** command.

   dm-xx indicates the device ID obtained in **Step 1.1**.

   If the returned result is **Input/output error** and the read data is **0 bytes (0 B) copied**, the device is unreadable.

   ```
   #dd if=/dev/dm-12 of=/dev/null count=1 bs=1M iflag=direct
   dd: error reading '/dev/dm-12': Input/output error
   0+0 records in
   0+0 records out
   0 bytes (0 B) copied, 0.0236862 s, 0.0 kB/s
   ```

   - If yes, record the residual *dm-xx* device and associated disk IDs (for details, see **Step 1.1**) and perform the clearing operation.
   - If the command execution is suspended, go to **Step 1.3**.
   - If other cases, contact technical support engineers.

3. Log in to the node again in another window.

   a. Run the following command to view the suspended process.

   ```
   # ps -ef | grep dm-12 | grep -w dd
   root     21725  9748  0 10:33 pts/10   00:00:00 dd if=/dev/dm-12 of=/dev/null count=1 bs=10M
   iflag=direct
   ```

   b. Kill the pid.

   ```
   # kill -9 pid
   ```

   c. Record the residual *dm-xx* device and associated disk IDs (for details, see **Step 1.1**) and perform the clearing operation.

Step 2 Clear the residual drive letters on the host.

1. Run the **multipath -f /dev/*dm-**** command to delete residual multipathing aggregation device information according to the DM multipathing device obtained in **Step 1**.

   ```
   # multipath -f /dev/dm-12
   ```


If an error is reported, contact technical support engineers.

2. Run the following command to clear the residual SCSI disks according to the drive letters of the residual disks obtained in **Step 1**.

```
echo 1 > /sys/block/xxxx/device/delete
```

When multiple paths are configured, clear the residual disks based on the drive letters. The residual paths are **sdi** and **sdj**.

```
# echo 1 > /sys/block/sdi/device/delete
# echo 1 > /sys/block/sdj/device/delete
```

If an error is reported, contact technical support engineers.

3. Check whether the DM multipathing device and SCSI disk information has been cleared.

Run the **multipath -ll**, **ls -l /sys/block/**, and **ls -l /dev/disk/by-id/** commands in sequence to query the path and disk information. If the residual **dm-12** device and SCSI disks **sdi** and **sdj** are cleared, the clearing is complete.

```
# multipath -ll
mpathb (3618cf24100f8f457014a764c000001f6) dm-3 HUAWEI  ,XSG1
size=100G features='0' hwhandler='0' wp=rw
`-+- policy='service-time 0' prio=-1 status=active
  |- 39:0:0:1        sdd 8:48  active ready running
  `- 38:0:0:1        sde 8:64  active ready running
mpathn (3618cf24100f8f457315a764c000001f6) dm-5 HUAWEI  ,XSG1
size=100G features='0' hwhandler='0' wp=rw
`-+- policy='service-time 0' prio=-1 status=active
  |- 39:0:0:2        sdc 8:32  active ready running
  `- 38:0:0:2        sdb 8:16  active ready running
# ls -l /sys/block/
total 0
lrwxrwxrwx 1 root root 0 Aug 11 19:56 dm-0 -> ../devices/virtual/block/dm-0
lrwxrwxrwx 1 root root 0 Aug 11 19:56 dm-1 -> ../devices/virtual/block/dm-1
lrwxrwxrwx 1 root root 0 Aug 11 19:56 dm-2 -> ../devices/virtual/block/dm-2
lrwxrwxrwx 1 root root 0 Aug 11 19:56 dm-3 -> ../devices/virtual/block/dm-3
lrwxrwxrwx 1 root root 0 Aug 11 19:56 sdb -> ../devices/platform/host35/session2/
target35:0:0/35:0:0:1/block/sdb
lrwxrwxrwx 1 root root 0 Aug 11 19:56 sdc -> ../devices/platform/host34/
target34:65535:5692/34:65535:5692:0/block/sdc
lrwxrwxrwx 1 root root 0 Aug 11 19:56 sdd -> ../devices/platform/host39/session6/
target39:0:0/39:0:0:1/block/sdd
lrwxrwxrwx 1 root root 0 Aug 11 19:56 sde -> ../devices/platform/host38/session5/
target38:0:0/38:0:0:1/block/sde
lrwxrwxrwx 1 root root 0 Aug 11 19:56 sdh -> ../devices/platform/host39/session6/
target39:0:0/39:0:0:3/block/sdh
lrwxrwxrwx 1 root root 0 Aug 11 19:56 sdi -> ../devices/platform/host38/session5/target38:0:0/38:0:0:3/
block/sdi
ls -l /dev/disk/by-id/
total 0
lrwxrwxrwx 1 root root 10 Aug 11 19:57 dm-name-mpathb -> ../../dm-3
lrwxrwxrwx 1 root root 10 Aug 11 19:58 dm-name-mpathn -> ../../dm-5
lrwxrwxrwx 1 root root 10 Aug 11 19:57 dm-uuid-mpath-3618cf24100f8f457014a764c000001f6 -> ../../
dm-3
lrwxrwxrwx 1 root root 10 Aug 11 19:58 dm-uuid-mpath-3618cf24100f8f457315a764c000001f6 -> ../../
dm-5
lrwxrwxrwx 1 root root  9 Aug 11 19:57 scsi-3618cf24100f8f457014a764c000001f6 -> ../../sdd
lrwxrwxrwx 1 root root  9 Aug 11 19:57 scsi-3618cf24100f8f45712345678000103e8 -> ../../sdi
lrwxrwxrwx 1 root root  9 Aug  3 15:17 scsi-3648435a10058805278654321ffffffff -> ../../sdb
lrwxrwxrwx 1 root root  9 Aug  2 14:49 scsi-368886030000020aff44cc0d060c987f1 -> ../../sdc
lrwxrwxrwx 1 root root  9 Aug 11 19:57 wwn-0x618cf24100f8f457014a764c000001f6 -> ../../sdd
lrwxrwxrwx 1 root root  9 Aug 11 19:57 wwn-0x618cf24100f8f45712345678000103e8 -> ../../sdi
lrwxrwxrwx 1 root root  9 Aug  3 15:17 wwn-0x648435a10058805278654321ffffffff -> ../../sdb
lrwxrwxrwx 1 root root  9 Aug  2 14:49 wwn-0x68886030000020aff44cc0d060c987f1 -> ../../sdc
```

----End

11.5 Failed to Start huawei-csi Services with the Status Displayed as InvalidImageName

Symptom

The huawei-csi services (huawei-csi-controller or huawei-csi-node) cannot be started. After the **kubectl get pod -A | grep huawei** command is executed, the command output shows that the service status is **InvalidImageName**.

```
# kubectl get pod -A | grep huawei
huawei-csi     huawei-csi-controller-fd5f97768-qlldc     6/9     InvalidImageName     0          16s
huawei-csi     huawei-csi-node-25txd                     2/3     InvalidImageName     0          15s
```

Root Cause Analysis

In the .yaml configuration files of the controller and node, the Huawei CSI image version number is incorrect. For example:

```
...
- name: huawei-csi-driver
  image: huawei-csi:4.2.0
...
```

Solution or Workaround

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the following command to modify the configuration file of the huawei-csi-node service. Press **I** or **Insert** to enter the editing mode and modify related parameters. After the modification is complete, press **Esc** and enter **:wq!** to save the modification.

```
# kubectl edit daemonset huawei-csi-node -o yaml -n=huawei-csi
```

NOTE

- In **huawei-csi-driver** in the sample .yaml file, modify **image** to Huawei CSI image **huawei-csi:4.2.0**.

  ```
  containers:
    ...
    - name: huawei-csi-driver
      image: huawei-csi:4.2.0
  ```

Step 3 Run the following command to modify the configuration file of the huawei-csi-controller service: Press **I** or **Insert** to enter the editing mode and modify related parameters. After the modification is complete, press **Esc** and enter **:wq!** to save the modification.

```
# kubectl edit deployment huawei-csi-controller -o yaml -n=huawei-csi
```

NOTE

- In **huawei-csi-driver** in the sample .yaml file, modify **image** to Huawei CSI image **huawei-csi:4.2.0**.

  ```
  containers:
    ...
    - name: huawei-csi-driver
      image: huawei-csi:4.2.0
  ```


Step 4 Wait until the huawei-csi-node and huawei-csi-controller services are started.

Step 5 Run the following command to check whether the huawei-csi services are started.

```
# kubectl get pod -A  | grep huawei
huawei-csi   huawei-csi-controller-58799449cf-zvhmv   9/9     Running      0          2m29s
huawei-csi   huawei-csi-node-7fxh6                    3/3     Running      0          12m
```

----End

11.6 When a PVC Is Created, the PVC Is in the Pending State

Symptom

A PVC is created. After a period of time, the PVC is still in the **Pending** state.

Root Cause Analysis

Cause 1: A StorageClass with the specified name is not created in advance. As a result, Kubernetes cannot find the specified StorageClass name when a PVC is created.

Cause 2: The storage pool capability does not match the StorageClass capability. As a result, huawei-csi fails to select a storage pool.

Cause 3: An error code (for example, 50331651) is returned by a RESTful interface of the storage. As a result, huawei-csi fails to create a PVC.

Cause 4: The storage does not return a response within the timeout period set by huawei-csi. As a result, huawei-csi returns a timeout error to Kubernetes.

Cause 5: Other causes.

Solution or Workaround

When a PVC is created, if the PVC is in the **Pending** state, you need to take different measures according to the following causes.

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the following command to view details about the PVC.

```
# kubectl describe pvc mypvc
```

Step 3 Perform the corresponding operation according to the **Events** information in the detailed PVC information.

- If the PVC is in the **Pending** state due to cause 1, perform the following steps.

  ```
  Events:
    Type     Reason              Age                From                         Message
    ----     ------              ----               ----                         -------
    Warning  ProvisioningFailed  0s (x15 over 3m24s)  persistentvolume-controller
  storageclass.storage.k8s.io "mysc" not found
  ```

  a. Delete the PVC.

  b. Create a StorageClass. For details, see **8.1.1.1.1 Configuring a StorageClass**.

c. Create a PVC. For details, see **8.1.1.1.2 Configuring a PVC**.

- If the PVC is in the **Pending** state due to cause 2, perform the following steps.

```
Events:
  Type     Reason              Age
From                                                                                                    Message
  ----     ------              ----             ----
-------
  Normal   Provisioning        63s (x3 over 64s)  csi.huawei.com_huawei-csi-controller-b59577886-qqzm8_58533e4a-884c-4c7f-92c3-6e8a7b327515  External provisioner is provisioning volume for claim "default/mypvc"
  Warning  ProvisioningFailed  63s (x3 over 64s)  csi.huawei.com_huawei-csi-controller-b59577886-qqzm8_58533e4a-884c-4c7f-92c3-6e8a7b327515  failed to provision volume with StorageClass "mysc": rpc error: code = Internal desc = failed to select pool, the capability filter failed, error: failed to select pool, the final filter field: replication, parameters map[allocType:thin replication:True size:1099511627776 volumeType:lun]. please check your storage class
```

a. Delete the PVC.

b. Delete the StorageClass.

c. Modify the **StorageClass.yaml** file based on the **Events** information.

d. Create a StorageClass. For details, see **8.1.1.1.1 Configuring a StorageClass**.

e. Create a PVC. For details, see **8.1.1.1.2 Configuring a PVC**.

- If the PVC is in the **Pending** state due to cause 3, contact Huawei engineers.

```
Events:
  Type     Reason              Age
From                                                                                                    Message
  ----     ------              ----             ----
-------
  Normal   Provisioning        63s (x4 over 68s)  csi.huawei.com_huawei-csi-controller-b59577886-qqzm8_58533e4a-884c-4c7f-92c3-6e8a7b327515  External provisioner is provisioning volume for claim "default/mypvc"
  Warning  ProvisioningFailed  62s (x4 over 68s)  csi.huawei.com_huawei-csi-controller-b59577886-qqzm8_58533e4a-884c-4c7f-92c3-6e8a7b327515  failed to provision volume with StorageClass "mysc": rpc error: code = Internal desc = Create volume map[ALLOCTYPE:1 CAPACITY:20 DESCRIPTION:Created from Kubernetes CSI NAME:pvc-63ebfda5-4cf0-458e-83bd-ecc PARENTID:0] error: 50331651
```

- If the PVC is in the **Pending** state due to cause 4, perform the following steps.

```
Events:
  Type     Reason              Age
From                                                                                                    Message
  ----     ------              ----             ----
-------
  Normal   Provisioning        63s (x3 over 52s)  csi.huawei.com_huawei-csi-controller-b59577886-qqzm8_58533e4a-884c-4c7f-92c3-6e8a7b327515  External provisioner is provisioning volume for claim "default/mypvc"
  Warning  ProvisioningFailed  63s (x3 over 52s)  csi.huawei.com_huawei-csi-controller-b59577886-qqzm8_58533e4a-884c-4c7f-92c3-6e8a7b327515  failed to provision volume with StorageClass "mysc": rpc error: code = Internal desc = context deadline exceeded (Client.Timeout exceeded while awaiting headers)
```

a. Wait for 10 minutes and check the PVC details again by referring to this section.

b. If it is still in the **Pending** state, contact Huawei engineers.

- If the PVC is in the **Pending** state due to cause 5, contact Huawei engineers.

----End

11.7 Before a PVC Is Deleted, the PVC Is in the Pending State

Symptom

Before a PVC is deleted, the PVC is in the **Pending** state.

Root Cause Analysis

Cause 1: A StorageClass with the specified name is not created in advance. As a result, Kubernetes cannot find the specified StorageClass name when a PVC is created.

Cause 2: The storage pool capability does not match the StorageClass capability. As a result, huawei-csi fails to select a storage pool.

Cause 3: An error code (for example, 50331651) is returned by a RESTful interface of the storage. As a result, huawei-csi fails to create a PVC.

Cause 4: The storage does not return a response within the timeout period set by huawei-csi. As a result, huawei-csi returns a timeout error to Kubernetes.

Cause 5: Other causes.

Solution or Workaround

To delete a PVC in the **Pending** state, you need to take different measures according to the following causes.

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the following command to view details about the PVC.

```
# kubectl describe pvc mypvc
```

Step 3 Perform the corresponding operation according to the **Events** information in the detailed PVC information.

- If the PVC is in the **Pending** state due to cause 1, run the **kubectl delete pvc** *mypvc* command to delete the PVC.

```
Events:
  Type     Reason              Age                From                         Message
  ----     ------              ----               ----                         -------
  Warning  ProvisioningFailed  0s (x15 over 3m24s)  persistentvolume-controller
storageclass.storage.k8s.io "mysc" not found
```

- If the PVC is in the **Pending** state due to cause 2, run the **kubectl delete pvc** *mypvc* command to delete the PVC.

```
Events:
  Type     Reason              Age
From                                                                                                  Message
  ----     ------              ----               ----
-------
  Normal   Provisioning        63s (x3 over 64s)  csi.huawei.com_huawei-csi-controller-b59577886-
qqzm8_58533e4a-884c-4c7f-92c3-6e8a7b327515  External provisioner is provisioning volume for
claim "default/mypvc"
  Warning  ProvisioningFailed  63s (x3 over 64s)  csi.huawei.com_huawei-csi-controller-b59577886-
```



```
qqzm8_58533e4a-884c-4c7f-92c3-6e8a7b327515  failed to provision volume with StorageClass
"mysc": rpc error: code = Internal desc = failed to select pool, the capability filter failed, error: failed
to select pool, the final filter field: replication, parameters map[allocType:thin replication:True
size:1099511627776 volumeType:lun]. please check your storage class
```

- If the PVC is in the **Pending** state due to cause 3, run the **kubectl delete pvc** *mypvc* command to delete the PVC.

```
Events:
  Type     Reason              Age
From                                                                                    Message
  ----     ------              ----             ----
-------
  Normal   Provisioning        63s (x4 over 68s)  csi.huawei.com_huawei-csi-controller-b59577886-
qqzm8_58533e4a-884c-4c7f-92c3-6e8a7b327515  External provisioner is provisioning volume for
claim "default/mypvc"
  Warning  ProvisioningFailed  62s (x4 over 68s)  csi.huawei.com_huawei-csi-controller-b59577886-
qqzm8_58533e4a-884c-4c7f-92c3-6e8a7b327515  failed to provision volume with StorageClass
"mysc": rpc error: code = Internal desc = Create volume map[ALLOCTYPE:1 CAPACITY:20
DESCRIPTION:Created from Kubernetes CSI NAME:pvc-63ebfda5-4cf0-458e-83bd-ecc PARENTID:0]
error: 50331651
```

- If the PVC is in the **Pending** state due to cause 4, contact Huawei engineers.

```
Events:
  Type     Reason              Age
From                                                                                    Message
  ----     ------              ----             ----
-------
  Normal   Provisioning        63s (x3 over 52s)  csi.huawei.com_huawei-csi-controller-b59577886-
qqzm8_58533e4a-884c-4c7f-92c3-6e8a7b327515  External provisioner is provisioning volume for
claim "default/mypvc"
  Warning  ProvisioningFailed  63s (x3 over 52s)  csi.huawei.com_huawei-csi-controller-b59577886-
qqzm8_58533e4a-884c-4c7f-92c3-6e8a7b327515  failed to provision volume with StorageClass
"mysc": rpc error: code = Internal desc = context deadline exceeded (Client.Timeout exceeded
while awaiting headers)
```

- If the PVC is in the **Pending** state due to cause 5, contact Huawei engineers.

----End

11.8 When a Pod Is Created, the Pod Is in the ContainerCreating State

Symptom

A Pod is created. After a period of time, the Pod is still in the **ContainerCreating** state. Check the log information (for details, see **11.1 How Do I View Huawei CSI Logs?**). The error message "Fibre Channel volume device not found" is displayed.

Root Cause Analysis

This problem occurs because residual disks exist on the host node. As a result, disks fail to be found when a Pod is created next time.

Solution or Workaround

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the following command to query information about the node where the Pod resides.


```
# kubectl get pod -o wide
NAME      READY   STATUS              RESTARTS   AGE    IP            NODE    NOMINATED NODE
READINESS GATES
mypod     0/1     ContainerCreating   0          51s    10.244.1.224  node1   <none>           <none>
```

Step 3 Delete the Pod.

Step 4 Use a remote access tool, such as PuTTY, to log in to the *node1* node in the Kubernetes cluster through the management IP address. *node1* indicates the node queried in **Step 2**.

Step 5 Clear the residual drive letters. For details, see **Solution or Workaround**.

----End

11.9 A Pod Is in the ContainerCreating State for a Long Time When It Is Being Created

Symptom

When a Pod is being created, the Pod is in the **ContainerCreating** state for a long time. Check the huawei-csi-node log (for details, see **11.1 How Do I View Huawei CSI Logs?**). No Pod creation information is recorded in the huawei-csi-node log. After the **kubectl get volumeattachment** command is executed, the name of the PV used by the Pod is not displayed in the **PV** column. After a long period of time (more than ten minutes), the Pod is normally created and the Pod status changes to **Running**.

Root Cause Analysis

The kube-controller-manager component of Kubernetes is abnormal.

Solution or Workaround

Contact container platform engineers to rectify the fault.

11.10 A Pod Fails to Be Created and the Log Shows That the Execution of the mount Command Times Out

Symptom

When a Pod is being created, the Pod keeps in the **ContainerCreating** status. In this case, check the log information of huawei-csi-node (for details, see **11.1 How Do I View Huawei CSI Logs?**). The log shows that the execution of the mount command times out.

Root Cause Analysis

Cause 1: The configured service IP address is disconnected. As a result, the **mount** command execution times out and fails.

Cause 2: For some operating systems, such as Kylin V10 SP1 and SP2, it takes a long time to run the **mount** command in a container using NFSv3. As a result, the

mount command may time out. The possible cause is that the value of **LimitNOFILE** of container runtime containerd is too large (1+ billion).

Cause 3: The mounting may fail due to network problems. The default mounting timeout period of CSI is 30 seconds. If the mounting still fails after 30 seconds, logs show that the execution of the **mount** command times out.

Solution or Workaround

Step 1 Run the **ping** command to check whether the service IP network is connected. If the ping fails, the fault is caused by cause 1. In this case, configure an available service IP address. If the ping succeeds, go to **Step 2**.

Step 2 Go to any container where the **mount** command can be executed and use NFSv3 to run the **mount** command. If the command times out, the fault may be caused by cause 2. Run the **systemctl status containerd.service** command to check the configuration file path, and then run the **cat** */xxx/containerd.service* command to check the configuration file. If the file contains **LimitNOFILE=infinity** or the value of **LimitNOFILE** is 1 billion, go to **Step 3**. Otherwise, contact Huawei technical support engineers.

Step 3 For cause 2, perform the following operations:

- Try using NFSv4.0 or NFSv4.1.
- Change the value of **LimitNOFILE** to a proper one by referring to **change solution provided by the community**. This solution will restart the container runtime. Evaluate the impact on services.

Step 4 Manually mount the file system on the host machine where the mounting fails. If the required time exceeds 30 seconds, check whether the network between the host machine and the storage node is normal. An example of the **mount** command is as follows.

- Run the following command to create a test directory.
  ```
  # mkdir /tmp/test_mount
  ```
- Run the **mount** command to mount the file system and observe the time consumed. The value of *ip:nfs_share_path* can be obtained from the huawei-csi-node log. For details, see **11.1 How Do I View Huawei CSI Logs?**.
  ```
  # time mount ip:nfs_share_path /tmp/test_mount
  ```
- After the test is complete, run the following command to unmount the file system.
  ```
  # umount /tmp/test_mount
  ```

----End

11.11 A Pod Fails to Be Created and the Log Shows That the mount Command Fails to Be Executed

Symptom

In NAS scenarios, when a Pod is being created, the Pod keeps in the **ContainerCreating** status. In this case, check the log information of huawei-csi-node (for details, see **11.1 How Do I View Huawei CSI Logs?**). The log shows that the mount command fails to be executed.

Root Cause Analysis

The possible cause is that the NFS 4.0/4.1 protocol is not enabled on the storage side. After the NFS v4 protocol fails to be used for mounting, the host does not negotiate to use the NFS v3 protocol for mounting.

Solution or Workaround

- Enable the NFS 3/4.0/4.1 protocol on the storage side and retry the default mounting.
- Specify an available NFS protocol for mounting. For details, see **8.1.1.1.1 Configuring a StorageClass**.

11.12 After a Pod Fails to Be Created or kubelet Is Restarted, Logs Show That the Mount Point Already Exists

Symptom

When a Pod is being created, the Pod is always in the **ContainerCreating** state. Alternatively, after kubelet is restarted, logs show that the mount point already exists. Check the log information of huawei-csi-node (for details, see **11.1 How Do I View Huawei CSI Logs?**). The error information is: **The mount /var/lib/kubelet/pods/xxx/mount is already exist, but the source path is not /var/lib/kubelet/plugins/kubernetes.io/xxx/globalmount**

Root Cause Analysis

The root cause of this problem is that Kubernetes performs repeated mounting operations.

Solution or Workaround

Run the following command to unmount the existing path. In the command, **/var/lib/kubelet/pods/xxx/mount** indicates the existing mount path displayed in the logs.

```
umount /var/lib/kubelet/pods/xxx/mount
```

11.13 How Do I Download a Container Image to the Local PC?

Download a Container Image Using containerd

Step 1 Run the **ctr image pull** *image:tag* command to download an image to the local PC. In the preceding command, *image:tag* indicates the image to be pulled and its tag.

```
# ctr image pull k8s.gcr.io/sig-storage/livenessprobe:v2.5.0
```


Step 2 Run the **ctr image export** *image*.**tar** *image:tag* command to export the image to a file. In the preceding command, *image:tag* indicates the image to be exported, and *image.tar* indicates the name of the exported image file.

```
# ctr image export livenessprobe.tar k8s.gcr.io/sig-storage/livenessprobe:v2.5.0
```

----End

Download a Container Image Using Docker

Step 1 Run the **docker pull** *image:tag* command to download an image to the local PC. In the preceding command, *image:tag* indicates the image to be pulled.

```
# docker pull k8s.gcr.io/sig-storage/livenessprobe:v2.5.0
```

Step 2 Run the **docker save** *image:tag* **-o** *image.tar* command to export the image to a file. In the preceding command, *image:tag* indicates the image to be exported, and *image.tar* indicates the name of the exported image file.

```
# docker save k8s.gcr.io/sig-storage/livenessprobe:v2.5.0 -o livenessprobe.tar
```

----End

Download a Container Image Using Podman

Step 1 Run the **podman pull** *image:tag* command to download an image to the local PC. In the preceding command, *image:tag* indicates the image to be pulled.

```
# podman pull k8s.gcr.io/sig-storage/livenessprobe:v2.5.0
```

Step 2 Run the **podman save** *image:tag* **-o** *image.tar* command to export the image to a file. In the preceding command, *image:tag* indicates the image to be exported, and *image.tar* indicates the name of the exported image file.

```
# podman save k8s.gcr.io/sig-storage/livenessprobe:v2.5.0 -o livenessprobe.tar
```

----End

11.14 How Do I Obtain CSI Version Information?

This section describes how to view the CSI version.

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the following command to query information about the node where huawei-csi-node resides.

```
# kubectl get pod -A  -owide | grep huawei-csi-node
NAMESPACE     NAME                                 READY   STATUS    RESTARTS       AGE     IP
NODE           NOMINATED NODE   READINESS GATES
huawei-csi    huawei-csi-node-87mss                   3/3     Running   0              6m41s
192.168.129.155      node-1          <none>           <none>
huawei-csi    huawei-csi-node-xp8cc                   3/3     Running   0              6m41s   192.168.129.156
node-2          <none>           <none
```

Step 3 Use a remote access tool, such as PuTTY, to log in to any node where huawei-csi-node resides through the node IP address.

Step 4 Run the following command to view the CSI version.


```
# cat /var/lib/kubelet/plugins/csi.huawei.com/version
4.2.0
```

----End

11.15 Common Problems and Solutions for Interconnecting with the Tanzu Kubernetes Cluster

This section describes the common problems and solutions for interconnecting with the Tanzu Kubernetes cluster. Currently, the following problems occur during interconnection with the Tanzu Kubernetes cluster:

- A Pod cannot be created because the PSP permission is not created.
- The mount point of the host is different from that of the native Kubernetes. As a result, a volume fails to be mounted.
- The livenessprobe container port conflicts with the Tanzu vSphere port. As a result, the container restarts repeatedly.

11.15.1 A Pod Cannot Be Created Because the PSP Permission Is Not Created

Symptom

When huawei-csi-controller and huawei-csi-node are created, only the Deployment and DaemonSet resources are successfully created, and no Pod is created for the controller and node.

Root Cause Analysis

The service account used for creating resources does not have the "use" permission of the PSP policy.

Solution or Workaround

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run the **vi** *psp-use.yaml* command to create a file named **psp-use.yaml**

```
# vi psp-use.yaml
```

Step 3 Configure the **psp-use.yaml** file.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: huawei-csi-psp-role
rules:
- apiGroups: ['policy']
  resources: ['podsecuritypolicies']
  verbs: ['use']
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: huawei-csi-psp-role-cfg
roleRef:
```



```
  kind: ClusterRole
  name: huawei-csi-psp-role
  apiGroup: rbac.authorization.k8s.io
subjects:
- kind: Group
  apiGroup: rbac.authorization.k8s.io
  name: system:serviceaccounts:huawei-csi
- kind: Group
  apiGroup: rbac.authorization.k8s.io
  name: system:serviceaccounts:default
```

Step 4 Run the **kubectl create -f** *psp-use.yaml* command to create the PSP permission.

```
# kubectl create -f psp-use.yaml
```

----End

11.15.2 Changing the Mount Point of a Host

Symptom

A Pod fails to be created, and error message "mount point does not exist" is recorded in Huawei CSI logs.

Root Cause Analysis

The native Kubernetes cluster in the **pods-dir** directory of huawei-csi-node is inconsistent with the Tanzu Kubernetes cluster.

Solution or Workaround

Step 1 Go to the **helm/esdk/** directory and run the **vi values.yaml** command to open the configuration file.

```
# vi values.yaml
```

Step 2 Change the value of **kubeletConfigDir** to the actual installation directory of kubelet.

```
#
 Specify kubelet config dir path.
# kubernetes and openshift is usually /var/lib/kubelet
# Tanzu is usually /var/vcap/data/kubelet
kubeletConfigDir: /var/vcap/data/kubelet
```

----End

11.15.3 Changing the Default Port of the livenessprobe Container

Symptom

The livenessprobe container of the huawei-csi-controller component keeps restarting.

Root Cause Analysis

The default port (9808) of the livenessprobe container of huawei-csi-controller conflicts with the existing vSphere CSI port of Tanzu.

Solution or Workaround

Change the default port of the livenessprobe container to an idle port.

Step 1 Go to the **helm/esdk/templates** directory and run the **vi huawei-csi-controller.yaml** command to open the controller configuration file.

```
# vi huawei-csi-controller.yaml
```

Step 2 Change the default port (9808) of the livenessprobe container to an idle port.

----End

11.16 Common Problems and Solutions for Using the Tanzu Kubernetes Cluster

This section describes the common problems and solutions for using the Tanzu Kubernetes cluster.

11.16.1 Failed to Create an Ephemeral Volume

Symptom

A **generic ephemeral volume** fails to be created, and the error message **PodSecurityPolicy: unable to admit pod: [spec.volumes[0]: Invalid value: "ephemeral": ephemeral volumes are not allowed to be used spec.volumes[0]** is displayed.

Root Cause Analysis

The current PSP policy does not contain the permission to use ephemeral volumes.

Solution or Workaround

Add the permission to use ephemeral volumes to the default PSP **pks-privileged** and **pks-restricted**. The following is an example of modifying **pks-privileged**:

Step 1 Use a remote access tool, such as PuTTY, to log in to any master node in the Kubernetes cluster through the management IP address.

Step 2 Run **kubectl edit psp pks-privileged** to modify the **pks-privileged** configuration.

```
# kubectl edit psp pks-privileged
```

Step 3 Add **ephemeral** to **spec.volumes**. The following is an example.

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  annotations:
    apparmor.security.beta.kubernetes.io/allowedProfileName: '*'
    seccomp.security.alpha.kubernetes.io/allowedProfileNames: '*'
  creationTimestamp: "2022-10-11T08:07:00Z"
  name: pks-privileged
  resourceVersion: "1227763"
```



```
  uid: 2f39c44a-2ce7-49fd-87ca-2c5dc3bfc0c6
spec:
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  supplementalGroups:
    rule: RunAsAny
  volumes:
  - glusterfs
  - hostPath
  - iscsi
  - nfs
  - persistentVolumeClaim
  - ephemeral
```

Step 4 Run the **kubectl get psp pks-privileged -o yaml** command to check whether the addition is successful.

```
# kubectl get psp pks-privileged -o yaml
```

----End

11.17 Failed to Expand the Capacity of a Generic Ephemeral Volume

Symptom

In an environment where the Kubernetes version is earlier than 1.25, the capacity of a **generic ephemeral volume** of the LUN type fails to be expanded. The system displays a message indicating that the PV capacity has been expanded, but the PVC capacity fails to be updated.

Root Cause Analysis

This problem is caused by a Kubernetes **bug**, which has been resolved in Kubernetes 1.25.

11.18 Failed to Expand the PVC Capacity Because the Target Capacity Exceeds the Storage Pool Capacity

Symptom

In a Kubernetes environment earlier than 1.23, PVC capacity expansion fails when the target capacity exceeds the storage pool capacity.

Root Cause Analysis

This is a known issue in the Kubernetes community. For details, see **Recovering from Failure when Expanding Volumes**.

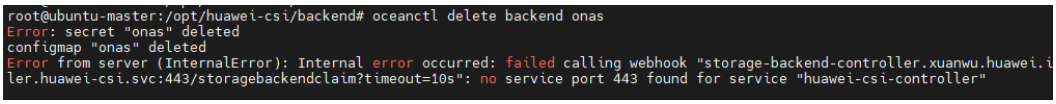
Solution or Workaround

For details, see **Recovering from Failure when Expanding Volumes**.

11.19 A webhook Fails to Be Called When the oceanctl Tool Is Used to Manage Backends

Symptom

After the webhook configuration is changed, for example, the value of the **webhookPort** parameter is changed, an error is reported indicating that a webhook fails to be called when the oceanctl tool is used to manage backends, as shown in the following figure.



Root Cause Analysis

After the webhook configuration changes, the **validatingwebhookconfiguration** resource becomes invalid.

Solution or Workaround

Step 1 Run the following command to delete the **validatingwebhookconfiguration** resource.

```
# kubectl delete validatingwebhookconfiguration storage-backend-controller.xuanwu.huawei.io
validatingwebhookconfiguration.admissionregistration.k8s.io "storage-backend-controller.xuanwu.huawei.io"
deleted
```

Step 2 Run the following command to restart CSI Controller. Run the **--replicas=**** command to set the number of CSI Controller copies to be restored. In the following example, the number of copies to be restored is 1. Change it based on site requirements.

```
# kubectl scale deployment huawei-csi-controller -n huawei-csi --replicas=0
deployment.apps/huawei-csi-controller scaled
# kubectl scale deployment huawei-csi-controller -n huawei-csi --replicas=1
deployment.apps/huawei-csi-controller scaled
```

Step 3 Run the following command to check whether CSI Controller is successfully started.

```
# kubectl get pod -n huawei-csi
NAME                                  READY   STATUS    RESTARTS   AGE
huawei-csi-controller-58d5b6b978-s2dsq   9/9     Running   0          19s
huawei-csi-node-dt6nd                    3/3     Running   0          77m
```

----End

11.20 An Account Is Locked After the Password Is Updated on the Storage Device

Symptom

After a user changes the password on the storage device, the account is locked.

Root Cause Analysis

CSI uses the account and password configured on the storage device to log in to the storage device. After the account password is changed on the storage device, CSI attempts to log in to the storage device again after the login fails. Take Dorado V6 as an example. The default login policy is that an account will be locked after three consecutive password verification failures. Therefore, when CSI retries for more than three times, the account will be locked.

Solution or Workaround

Step 1 If the backend account is **admin**, run the following command to set the number of huawei-csi-controller service copies to 0. If an account other than **admin** is used, skip this step.

```
kubectl scale deployment huawei-csi-controller -n huawei-csi --replicas=0
```

Step 2 Log in to the storage device as user **admin** and modify the login policy. Take Dorado V6 as an example. On DeviceManager, choose **Settings** > **User and Security** > **Security Policies** > **Login Policy**, click **Modify**, and disable **Account Lockout**.

Step 3 If the backend account is admin, run the following command to restore the number of CSI Controller copies using --replicas=*. In the following example, the number of copies is restored to 1. Change it based on the site requirements. If an account other than **admin** is used, skip this step.

```
kubectl scale deployment huawei-csi-controller -n huawei-csi --replicas=1
```

Step 4 Use the oceanctl tool to change the storage backend password. For details about how to change the backend password, see **7.3 Updating a Storage Backend**.

Step 5 Log in to the storage device as user admin and modify the login policy. Take Dorado V6 as an example. On DeviceManager, choose **Settings** > **User and Security** > **Security Policies** > **Login Policy**, click **Modify**, and enable **Account Lockout**.

----End

12 Appendix

12.1 Example ALUA Configuration Policy of OceanStor V3/V5 and OceanStor Dorado V3

Example 1: The configuration file content is as follows:

```
parameters:
  ALUA:
    "*":
      MULTIPATHTYPE: 1
      FAILOVERMODE: 3
      SPECIALMODETYPE: 0
      PATHTYPE: 0
    node1:
      MULTIPATHTYPE: 1
      FAILOVERMODE: 3
      SPECIALMODETYPE: 0
      PATHTYPE: 1
```

If the host name is **node1**, both of the preceding ALUA configuration sections can be used to configure initiators. According to the configuration policy rules in **9.1.1.1 Configuring ALUA Parameters for a Huawei Enterprise Storage Backend**, the priority of the second configuration section (where **HostName** is **node1**) is higher than that of the first configuration section (where **HostName** is *****).

Example 2: The configuration file content is as follows:

```
parameters:
  ALUA:
    node[0-9]:
      MULTIPATHTYPE: 1
```



```
    FAILOVERMODE: 3
    SPECIALMODETYPE: 0
    PATHTYPE: 0
  node[5-7]:
    MULTIPATHTYPE: 1
    FAILOVERMODE: 3
    SPECIALMODETYPE: 0
    PATHTYPE: 1
```

If the host name is **node6**, both of the preceding ALUA configuration sections can be used to configure initiators. According to the configuration policy rules in **9.1.1.1 Configuring ALUA Parameters for a Huawei Enterprise Storage Backend**, select the first ALUA configuration section to configure initiators.

Example 3: The configuration file content is as follows:

```
parameters:
  ALUA:
   node$:
     MULTIPATHTYPE: 1
     FAILOVERMODE: 3
     SPECIALMODETYPE: 0
     PATHTYPE: 0
   node10$:
     MULTIPATHTYPE: 1
     FAILOVERMODE: 3
     SPECIALMODETYPE: 0
     PATHTYPE: 1
```

According to the configuration policy rules in **9.1.1.1 Configuring ALUA Parameters for a Huawei Enterprise Storage Backend**: For host **node1**, select the first ALUA configuration section to configure initiators. For host **node10**, select the second ALUA configuration section to configure initiators. ^ matches the beginning of a character string, and **$** matches the end of a character string.

12.2 Example ALUA Configuration Policy of OceanStor Dorado 6.*x*

Example 1: The configuration file content is as follows:

```
parameters:
  ALUA:
    "*":
     accessMode: 1
     hyperMetroPathOptimized: 1
   node1:
     accessMode: 1
     hyperMetroPathOptimized: 0
```

If the host name is **node1**, both of the preceding ALUA configuration sections can be used to configure initiators. According to the configuration policy rules in **9.1.1.1 Configuring ALUA Parameters for a Huawei Enterprise Storage Backend**, the priority of the second configuration section (where **HostName** is **node1**) is higher than that of the first configuration section (where **HostName** is *****).

Example 2: The configuration file content is as follows:

```
parameters:
  ALUA:
   node[0-9]:
     accessMode: 1
```



```
    hyperMetroPathOptimized: 1
  node[5-7]:
    accessMode: 1
    hyperMetroPathOptimized: 0
```

If the host name is **node6**, both of the preceding ALUA configuration sections can be used to configure initiators. According to the configuration policy rules in **9.1.1.1 Configuring ALUA Parameters for a Huawei Enterprise Storage Backend**, select the first ALUA configuration section to configure initiators.

Example 3: The configuration file content is as follows:

```
parameters:
  node1$:
    node[0-9]:
    accessMode: 1
    hyperMetroPathOptimized: 1
  node10$:
    accessMode: 1
    hyperMetroPathOptimized: 0
```

According to the configuration policy rules in **9.1.1.1 Configuring ALUA Parameters for a Huawei Enterprise Storage Backend**: For host **node1**, select the first ALUA configuration section to configure initiators. For host **node10**, select the second ALUA configuration section to configure initiators. ^ matches the beginning of a character string, and **$** matches the end of a character string.

12.3 Example ALUA Configuration Policy of Distributed Storage

Example 1: The configuration file content is as follows:

```
parameters:
  ALUA:
    "*":
      switchoverMode: Enable_alua
      pathType: optimal_path
    node1:
      switchoverMode: Enable_alua
      pathType: non_optimal_path
```

If the host name is **node1**, both of the preceding ALUA configuration sections can be used to configure initiators. According to the configuration policy rules in **9.1.1.2 Configuring ALUA Parameters for a Distributed Storage Backend**, the priority of the second configuration section (where **HostName** is **node1**) is higher than that of the first configuration section (where **HostName** is *****).

Example 2: The configuration file content is as follows:

```
parameters:
  ALUA:
    node[0-9]:
      switchoverMode: Enable_alua
      pathType: optimal_path
    node[5-7]:
      switchoverMode: Enable_alua
      pathType: non_optimal_path
```

If the host name is **node6**, both of the preceding ALUA configuration sections can be used to configure initiators. According to the configuration policy rules in

9.1.1.2 Configuring ALUA Parameters for a Distributed Storage Backend, select the first ALUA configuration section to configure initiators.

Example 3: The configuration file content is as follows:

```
parameters:
  ALUA:
    node1$:
      switchoverMode: Enable_alua
      pathType: optimal_path
    node10$:
      switchoverMode: Enable_alua
      pathType: non_optimal_path
```

According to the configuration policy rules in **9.1.1.2 Configuring ALUA Parameters for a Distributed Storage Backend**: For host **node1**, select the first ALUA configuration section to configure initiators. For host **node10**, select the second ALUA configuration section to configure initiators. **^** matches the beginning of a character string, and **$** matches the end of a character string.

12.4 Installing Helm 3

This section describes how to install Helm 3.

For details, see **https://helm.sh/docs/intro/install/**.

Prerequisites

Ensure that the master node in the Kubernetes cluster can access the Internet.

Procedure

Step 1 Run the following command to download the Helm 3 installation script.

```
# curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3
```

Step 2 Run the following command to modify the permission on the Helm 3 installation script.

```
# chmod 700 get_helm.sh
```

Step 3 Determine the Helm version to be installed based on the version mapping between Helm and Kubernetes. For details about the version mapping, see **Helm Version Support Policy**. Then run the following command to change the **DESIRED_VERSION** environment variable to the Helm version to be installed and run the installation command.

```
# DESIRED_VERSION=v3.9.0 ./get_helm.sh
```

Step 4 Run the following command to check whether Helm 3 of the specified version is successfully installed.

```
# helm version
version.BuildInfo{Version:"v3.9.0", GitCommit:"7ceeda6c585217a19a1131663d8cd1f7d641b2a7",
GitTreeState:"clean", GoVersion:"go1.17.5"}
```

----End

12.5 Configuring Custom Permissions

User-defined Role Configurations

For different storage resources, refer to the following configurations:

- For NAS resources, configure the minimum permissions by referring to **Table 12-1**.
- For SAN resources, configure the minimum permissions by referring to **Table 12-2**.

NOTE

For details about how to configure permissions for user-defined roles, see **OceanStor Dorado 6000, Dorado 18000 Series Product Documentation**.

Table 12-1 Minimum permissions for NAS resources

Permission Object	Parent Object	Read/Write Permission	Function
workload_type	file_storage_service	Read-only	Queries the workload type.
file_system	file_storage_service	Read and write	Manages file systems.
fs_snapshot	file_storage_service	Read and write	Manages file system snapshots.
quota	file_storage_service	Read and write	Manages file system quotas.
nfs_service	file_storage_service	Read-only	Queries NFS services.
share	file_storage_service	Read and write	Manages NFS shares.
dtree	file_storage_service	Read and write	Manages dtrees.
hyper_metro_pair	hyper_metro	Read and write	Creates file system HyperMetro pairs.
hyper_metro_domain	hyper_metro	Read-only	Queries information about file system HyperMetro domains.
remote_device	local_data_protection	Read-only	Queries remote device information.
storage_pool	pool	Read-only	Queries storage pool information.

Permission Object	Parent Object	Read/Write Permission	Function
smart_qos	resource_performance_tuning	Read and write	Manages SmartQoS policies.
system	system	Read-only	Queries storage device information (this object needs to be configured only when the owning group is the system group).
vstore	vstore	Read-only	Queries vStore information.

Table 12-2 Minimum permissions for SAN resources

Permission Object	Parent Object	Read/Write Permission	Function
remote_device	local_data_protection	Read-only	Queries remote device information.
hyper_clone	local_data_protection	Read and write	Manages clone pairs.
lun_snapshot	local_data_protection	Read and write	Manages LUN snapshots.
workload_type	lun	Read-only	Queries the workload type.
lun	lun	Read and write	Manages LUNs.
host	mapping_view	Read and write	Manages hosts.
host_group	mapping_view	Read and write	Manages host groups.
initiator	mapping_view	Read and write	Manages initiators.
lun_group	mapping_view	Read and write	Manages LUN groups.
mapping_view	mapping_view	Read and write	Manages mapping views.
target	mapping_view	Read-only	Queries iSCSI initiators.
port	network	Read-only	Queries logical ports.

Permission Object	Parent Object	Read/Write Permission	Function
storage_pool	pool	Read-only	Queries storage pool information.
smart_qos	resource_performance_tuning	Read and write	Manages SmartQoS policies.
system	system	Read-only	Queries storage device information (this object needs to be configured only when the owning group is the system group).
vstore	vstore	Read-only	Queries vStore information.

