eSDK Huawei Storage Kubernetes CSI Plugins V4.5.0

User Guide

Issue 02

Date 2024-11-13

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

Initial Digital Certificate

The Initial digital certificates on Huawei devices are subject to the *Rights and Responsibilities of Initial 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
01	2024-09-29	This issue is the first official release.
02	2024-11-13	This issue is the second official release.

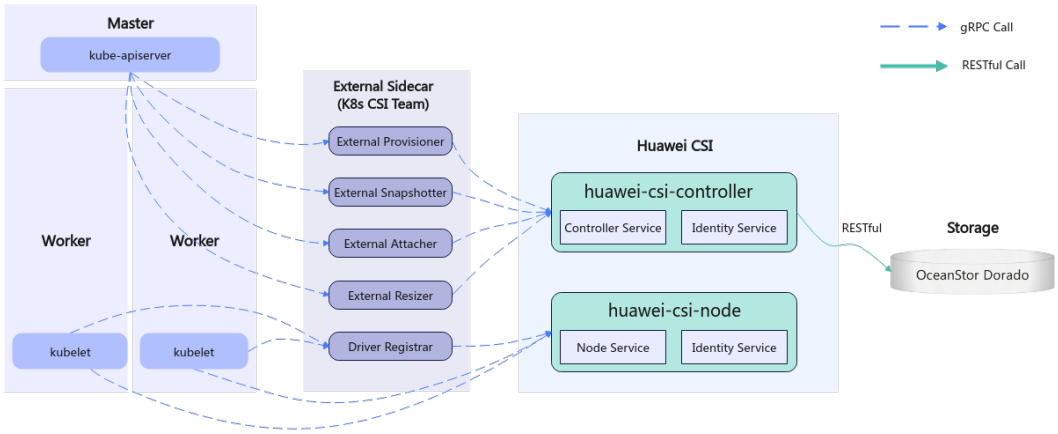
Contents

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 through gRPC 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



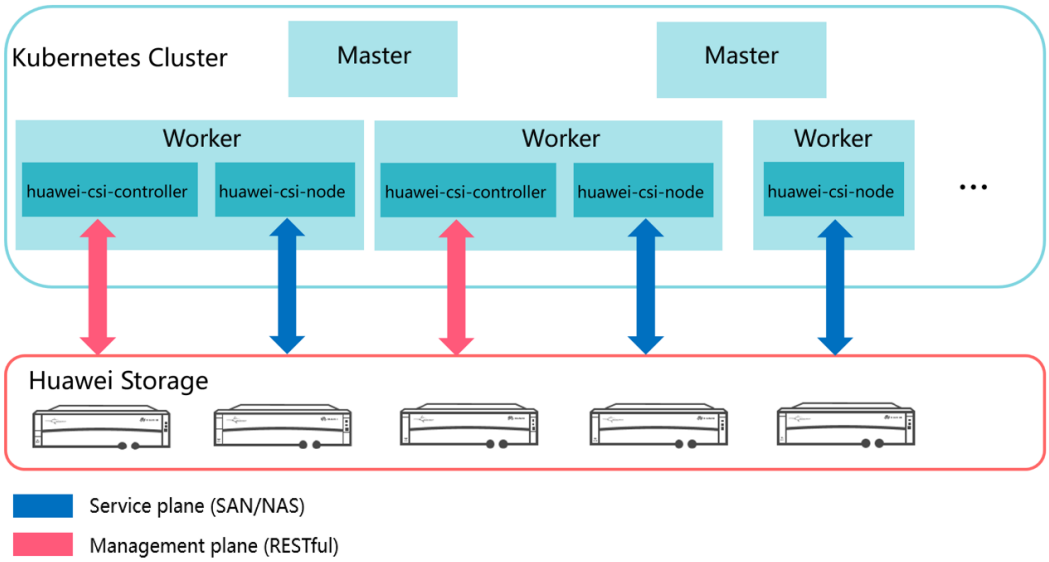
Huawei CSI consists of two components: huawei-csi-controller and huawei-csi-node.

- huawei-csi-controller: one or more Pods (including Controller Service and Identity Service) running in Deployment mode. It is used to interact with Huawei storage using RESTful. Therefore, the node running the huawei-csi-controller component must be connected to the management plane network of the storage.

- huawei-csi-node: a Pod (including Node Service and Identity Service) 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 huawei-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



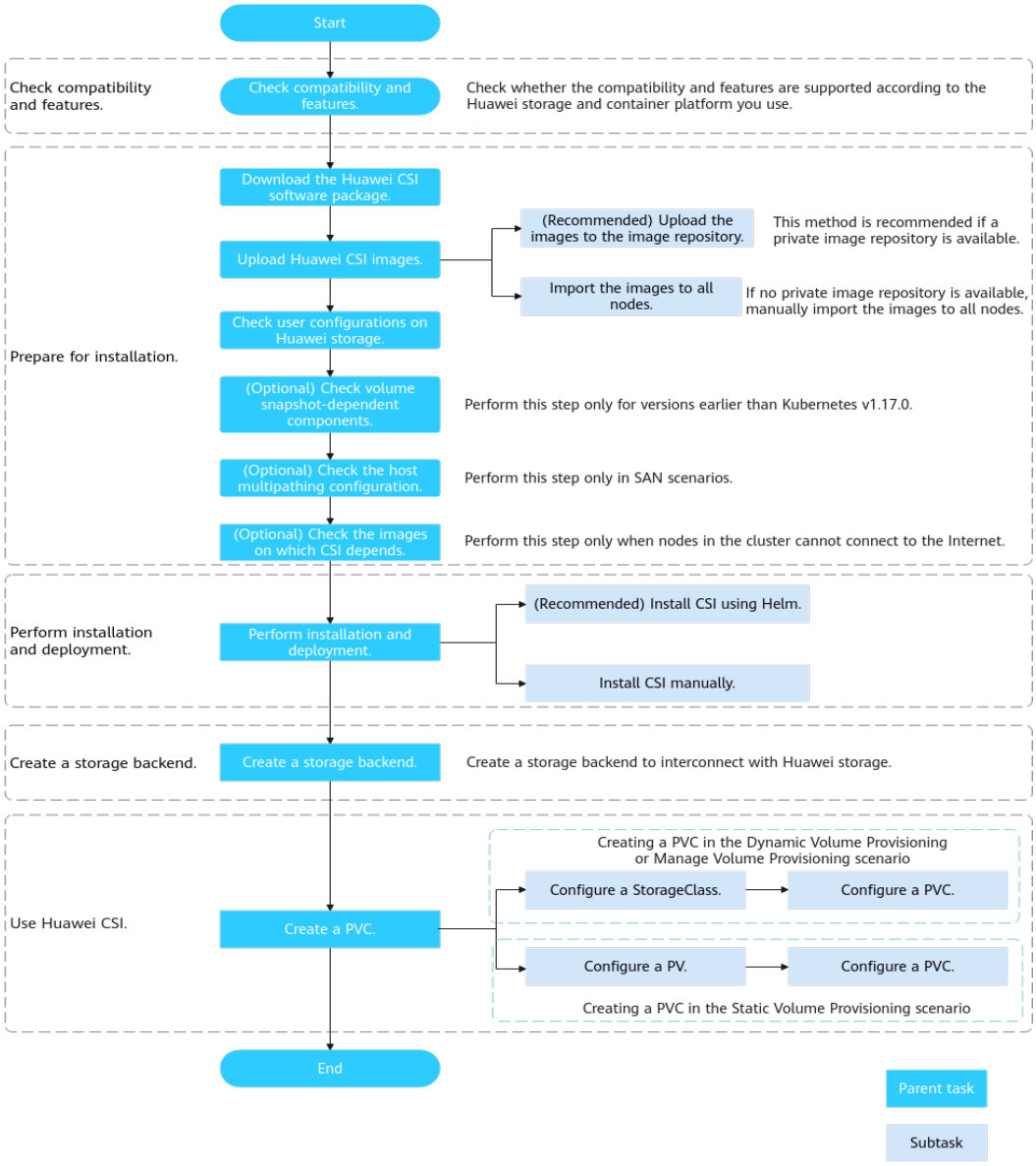
This document describes how to install, deploy, and use the Huawei CSI V4.5.0 plug-in.

2 Quick Start

This chapter describes how to quickly install and use Huawei CSI to manage Persistent Volume Claims (PVCs).

Huawei CSI Use Process

Figure 2-1 CSI installation and use process



Compatibility and Features

Before using this plug-in, learn about its compatibility with Huawei storage, container platforms, and host operating systems (OSs), as well as supported features.

Compatibility and Features

Installation Preparations

Before installing Huawei CSI, you need to prepare the configurations of related environments such as container platforms and hosts.

Installation Preparations

Installation and Deployment

Huawei CSI provides two installation modes: installation using Helm and manual installation, which are suitable for different container platforms such as Kubernetes and OpenShift.

Installation and Deployment

Creating a Storage Backend

Before using Huawei CSI, you need to create storage backend resources.

Creating a Storage Backend

Using Huawei CSI

Now, you can use Huawei CSI to manage PVCs.

Using Huawei CSI

3 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.

3.1 Kubernetes and OS Compatibility

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

Table 3-1 Supported container management platforms

Container Management Platform	Version
Kubernetes	1.16 to 1.30
Red Hat OpenShift Container Platform	4.6 EUS, 4.7, 4.8, 4.9, 4.10, 4.11, 4.12, 4.13, 4.14, 4.15
Tanzu Kubernetes	TKGI 1.14.1, TKGI 1.15, TKGI 1.16, TKGI 1.17, TKGI 1.18
CCE Agile	22.3.2
CCE	22.9.5

NOTICE

- The connection between Huawei CSI and Tanzu Kubernetes supports only the centralized storage NAS scenario. For the related FAQ, see **9.5 Common Problems and Solutions for Interconnecting with the Tanzu Kubernetes Cluster**.
- The connection between Huawei CSI and CCE or CCE Agile supports only centralized storage.

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

Table 3-2 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	UltraPath 31.2.1, supporting NVMe over RoCE
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, 4.14, 4.15	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

OS Name	OS Version	Native DM-Multipath Version	Huawei UltraPath Version
Ubuntu ARM	22.04	Delivered with the OS, supporting FC/iSCSI	Not supported
Kylin x86_64	7.6, V10 SP1, V10 SP2, V10 SP3	Delivered with the OS, supporting FC/iSCSI	UltraPath 31.2.0, supporting FC/iSCSI[1]
Kylin ARM	V10 SP1, V10 SP2, V10 SP3	Delivered with the OS, supporting FC/iSCSI	UltraPath 31.3.0, supporting iSCSI[2]
Debian x86_64	9, 11, 12	Delivered with the OS, supporting FC/iSCSI	Not supported
EulerOS x86_64	V2R9, V2R10, V2R11, V2R12	Delivered with the OS, supporting FC/iSCSI	Not supported
EulerOS ARM	V2R10, V2R12	Delivered with the OS, supporting FC/iSCSI	Not supported
UOS x86_64	V20	Delivered with the OS, supporting FC/iSCSI	Not supported
BC-Linux ARM	21.10	Delivered with the OS, supporting FC/iSCSI	Not supported
Anolis OS[3]	8.8	Delivered with the OS, supporting iSCSI	Not supported
OpenEuler x86_64	22.03 LTS SP1	Delivered with the OS, supporting iSCSI	Not supported
Red Hat Enterprise Linux x86_64	8.6, 8.7, 8.8, 9.4	Delivered with the OS, supporting FC/iSCSI	Not supported

Note 1: Only Kylin x86_64 V10 SP2 supports UltraPath 31.2.0.

Note 2: Only Kylin ARM V10 SP3 supports UltraPath 31.3.0.

Note 3: Anolis OS supports only OceanStor Pacific storage.

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

3.2 Kubernetes Feature Matrix

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

Table 3-3 Kubernetes versions and supported features

Feature	V1.16	V1.17	V1.18	V1.19	V1.20	V1.21+
Static Provisioning	√	√	√	√	√	√
Dynamic Provisioning	√	√	√	√	√	√
Manage Provisioning[1]	√	√	√	√	√	√
Expand Persistent Volume	√	√	√	√	√	√
Create VolumeSnapshot	x	√	√	√	√	√
Restore VolumeSnapshot	x	√	√	√	√	√
Delete VolumeSnapshot	x	√	√	√	√	√
Clone Persistent Volume	x	√	√	√	√	√
Modify Volume[2]	√	√	√	√	√	√
Raw Block Volume	√	√	√	√	√	√
Topology	√	√	√	√	√	√
Generic Ephemeral Inline Volumes	x	x	x	x	x	√
Volume Limits	x	√	√	√	√	√
FSGroup Support	x	x	x	x	√	√

- Note 1: Manage Provisioning is a volume management feature customized by Huawei CSI. This feature allows existing storage resources to be managed by

Kubernetes. You are not allowed to manage a storage resource for multiple times and concurrently delete or create a storage resource. When a storage resource is managed by multiple clusters, operations on the managed volume in a single cluster take effect only in the cluster and will not be synchronized to other clusters. Instead, you need to perform these operations on the managed volume in other clusters.

- Note 2: Modify Volume is a PVC change feature customized by Huawei CSI. This feature allows a common volume to be changed to a HyperMetro volume. To use this feature, ensure that the connected storage supports the volume HyperMetro feature.

3.3 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 3-4 Supported Huawei enterprise storage

Storage Product	Version
OceanStor V5	V500R007, V500R007 Kunpeng
OceanStor Dorado V3	V300R002
OceanStor	6.1.3, 6.1.5, 6.1.6, 6.1.7, 6.1.8
OceanStor Dorado	6.1.0, 6.1.2, 6.1.3, 6.1.5, 6.1.6, 6.1.7, 6.1.8

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

Table 3-5 Features supported by Huawei enterprise storage and constraints

Feature	OceanStor V5	OceanStor Dorado V3	OceanStor	OceanStor Dorado
Static Provisioning	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[5]	Volumes created in Dynamic Provisioning or Manage Provisioning mode are supported.			
Create VolumeSnapshot	Volumes created in Dynamic Provisioning or Manage Provisioning mode are supported.			

Feature	OceanStor V5	OceanStor Dorado V3	OceanStor	OceanStor Dorado
Delete VolumeSnapshot	Supported	Supported	Supported	Supported
Restore VolumeSnapshot	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 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.
Topology	Supported	Supported	Supported	Supported
Generic Ephemeral Volumes	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[6]	Supported	Supported	Supported
Application type	N/A	N/A	Supported	Supported
Volume HyperMetro[7]	Not supported	N/A	Only NAS volumes are supported.	
Storage multi-tenant	Only NAS volumes are supported.	N/A	Only NAS volumes are supported.[8]	

- Note 1: Manage Provisioning is a volume management feature customized by Huawei CSI. This feature allows existing storage resources to be managed by

Kubernetes. You are not allowed to manage a storage resource for multiple times and concurrently delete or create a storage resource.

- 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 6.1.0 and later versions support NFS. Only OceanStor Dorado 6.1.3 and later versions support NFS 4.1. OceanStor Dorado 6.1.7 and later versions support NFS over RDMA.
- Note 5: The provisioned PVC whose **volumeType** is **lun** and **accessModes** is **ReadOnlyMany** does not support capacity expansion.
- Note 6: Only system users can configure QoS.
- Note 7: Only the active-active (AA) mode is supported.
- Note 8: Only OceanStor Dorado 6.1.3 and later versions support multi-tenant.

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

Table 3-6 Features supported by Dtree

Feature	Supported
Static Provisioning	√
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 3-7 Huawei storage versions supported by Dtree

Storage Product	Version
OceanStor Dorado	6.1.0, 6.1.2, 6.1.3, 6.1.5, 6.1.6, 6.1.7, 6.1.8

3.4 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 3-8 Supported Huawei distributed storage

Storage Product	Version
FusionStorage Block	8.0.1
OceanStor Pacific series	8.1.0, 8.1.1, 8.1.2, 8.1.3, 8.1.5, 8.2.0

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

Table 3-9 Features supported by Huawei distributed storage and constraints

Feature	FusionStorage Block	OceanStor Pacific Series
Static Provisioning	SAN: iSCSI/SCSI	SAN: iSCSI/SCSI NAS: DPC[2]/NFS 3/4.1[3]
Dynamic Provisioning		
Manage Provisioning[1]		
Expand Persistent Volume[4]	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	Only SAN volume snapshots are supported.
Restore VolumeSnapshot	Supported	Only SAN volume snapshots are supported.

Feature	FusionStorage Block	OceanStor Pacific Series
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.
Topology	Supported	Supported
Generic Ephemeral Inline Volumes	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
Soft and hard quotas	Not supported	Only NAS volumes are supported.
Storage multi-tenant	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. You are not allowed to manage a storage resource for multiple times and concurrently delete or create a storage resource.
- 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.
- Note 4: The provisioned PVC whose **volumeType** is **lun** and **accessModes** is **ReadOnlyMany** does not support capacity expansion.

4 Installation and Deployment

4.1 Installation Preparations

This chapter describes the preparations for the installation.

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 **3.1 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 **3.3 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 **3.4 Compatibility with Huawei Distributed Storage**.
- 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.
- Ensure that the language of the operating system is English.

- Ensure that storage resource names, such as storage pool names and tenant names, are in English.

4.1.1 Downloading the 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 4.5.0 version based on the CPU architecture.

NOTE

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

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

Table 4-1 Component description

Component	Description
image/huawei-csi-v4.5.0-*arch*.tar	huawei-csi-driver image. *arch* is **X86** or **ARM**.
image/storage-backend-controller-v4.5.0-*arch*.tar	Back-end management controller image. *arch* is **X86** or **ARM**.
image/storage-backend-sidecar-v4.5.0-*arch*.tar	Back-end management sidecar image. *arch* is **X86** or **ARM**.
image/huawei-csi-extender-v4.5.0-*arch*.tar	huawei-csi-extender 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.
examples/backend	.yaml sample file used to create a storage backend.

----End

4.1.2 Uploading a Huawei CSI Image

Huawei provides the **huawei-csi** image for users. For details about how to obtain the image file, see **4.1.1 Downloading the 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.
- Manually import the CSI image to all nodes where Huawei CSI needs to be deployed.

Uploading an Image to the Image Repository

The installation of Huawei CSI depends on the following image files provided by Huawei. Import and upload the image files in sequence. For details about how to obtain the image files, see **4.1.1 Downloading the Huawei CSI Software Package**.

- huawei-csi-v4.5.0-arch.tar
- storage-backend-controller-v4.5.0-*arch*.tar
- storage-backend-sidecar-v4.5.0-*arch*.tar
- huawei-csi-extender-v4.5.0-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-v4.5.0-arch.tar** command to import the CSI image to the current node.

```
docker load -i huawei-csi-v4.5.0-arch.tar
```

Step 2 Run the **docker tag huawei-csi:4.5.0 repo.huawei.com/huawei-csi:4.5.0** command to add the image repository address to the image tag. **repo.huawei.com** indicates the image repository address.

```
docker tag huawei-csi:4.5.0 repo.huawei.com/huawei-csi:4.5.0
```

Step 3 Run the **docker push repo.huawei.com/huawei-csi:4.5.0** 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:4.5.0
```

----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 or CCE Agile platform, see the user manual of the platform.

Uploading an Image to a Local Node

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

Prerequisites

- The node has the corresponding Huawei CSI image file. For details about how to obtain the image file, see **4.1.1 Downloading the Huawei CSI Software Package**.
- Docker or another container engine has been installed on the node.

Procedure

Step 1 Use a remote access tool, such as PuTTY, to log in to the node where the image is to be imported through the management IP address.

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

Step 3 Run the **cd image** command to go to the **image** working directory. For details about the tool path, see **Table 4-1**.

Step 4 Run the following commands in sequence to import all Huawei CSI images in the image directory to the local node. In the commands, *name* indicates the name of a .tar image package.

Run the following command using the Docker container engine:

```
docker load -i <name>.tar
```

Run the following command using the containerd container engine:

```
ctr -n k8s.io image import <name>.tar
```

Run the following command using the Podman container engine:

```
podman load -i <name>.tar
```

NOTICE

If another container engine is installed on the node, use the image import command for the corresponding container engine.

----End

4.1.3 Checking User Configurations 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 users created on Huawei storage to communicate with Huawei storage. The following table lists the user information required for different storage devices.

Table 4-2 User requirements for connecting storage to CSI

Storage Type	User Type	Role	Level	Type
OceanStor V5	System user	Administrator	Administrator	Local user
	vStore user	vStore administrator	Administrator	Local user

Storage Type	User Type	Role	Level	Type
OceanStor Dorado V3	System user	Administrator	Administrator	Local user
OceanStor 6.1.3, 6.1.5, 6.1.6, 6.1.7, 6.1.8	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, 6.1.7, 6.1.8	System user	Administrator/ User-defined role[1]	N/A	Local user
	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 **10.5 Configuring Custom Permissions**.

NOTICE

You are advised not to use the users of the super administrator role.

4.1.4 Checking Volume Snapshot-Dependent Components

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

NOTICE

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
```

The following is an example of the command output.


```
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 4-1**.

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

----End

4.1.5 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 3-2**.

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 4-1**.

Procedure

Step 1 If you use the iSCSI/FC protocol to connect to Huawei enterprise storage, configure and check host multipathing by referring to ***OceanStor Dorado and OceanStor 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 ***OceanStor Dorado and OceanStor 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

4.1.6 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
  ```

4.1.7 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, you can 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, snapshot-controller, storage-backend-controller, storage-backend-sidecar, huawei-csi-driver, and huawei-csi-extender. 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 4-3 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	v1.16+	This image is provided by the Kubernetes community, used to monitor the health status of CSI and report 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	v1.16+	This image is provided by the Kubernetes community, used to call 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	v1.16+	This image is provided by the Kubernetes community, used to obtain CSI information and register 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.
csi-snapshotter	k8s.gcr.io/sig-storage/csi-snapshotter:v4.2.1	v1.17+	This image is provided by the Kubernetes community, used to call 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+	This image is provided by the Kubernetes community, used to listen to the VolumeSnapshot and VolumeSnapshotContent objects in the Kubernetes API and trigger csi-snapshotter to create a snapshot on the storage system when creating or deleting a VolumeSnapshot.

Container Name	Container Image	K8s Version Requirements	Feature Description
csi-provisioner	k8s.gcr.io/sig-storage/csi-provisioner:v3.0.0	v1.17+	This image is provided by the Kubernetes community, used to create or delete PVCs. • Calls the huawei-csi-controller service to create a LUN or file system on the storage system as a PV when creating a PVC. • Calls the huawei-csi-controller service to 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 huawei-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	
storage-backend-controller	storage-backend-controller:4.5.0	v1.16+	This image is provided by Huawei CSI software package, used to manage storageBackendClaim resources.
storage-backend-sidecar	storage-backend-sidecar:4.5.0	v1.16+	This image is provided by Huawei CSI software package, used to manage storageBackendContent resources.
huawei-csi-driver	huawei-csi:4.5.0	v1.16+	This image is provided by Huawei CSI software package, used to provide all features supported by Huawei CSI.
huawei-csi-extender	huawei-csi-extender:4.5.0	v1.16+	This image is provided by Huawei CSI software package, used to provide extended features of Huawei CSI.

NOTE

If the cluster is not connected to the Internet, manually download the container images and upload them to the cluster. For details, see **8.3 Downloading a Container Image**.

4.2 Installing Huawei CSI

This section describes how to install Huawei CSI.

NOTE

In the current version, resource requests and limits are added to Huawei CSI. For details, see **10.6 Huawei CSI Resource Management**.

Prerequisites

- Operations described in **4.1 Installation Preparations** have been completed.
- All worker nodes of the cluster communicate properly with the service network 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.

4.2.1 Installing Huawei CSI Using Helm

Helm Installation Description

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 **8.7 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 or delete volumes.
- Kubernetes External Attacher: used to attach or detach volumes.
- Kubernetes External Resizer: used to expand the capacity of volumes.
- Kubernetes External liveness-probe: used to determine the health status of a Pod.
- (Optional) huawei-csi-extender: Huawei CSI extender.
- (Optional) Kubernetes External Snapshotter: used to provide snapshot support (installed as CRD).
- (Optional) Kubernetes External Snapshot Controller: used to control volume snapshots.

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.

4.2.1.1 Installing Huawei CSI on Kubernetes, OpenShift, and Tanzu

Installation 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 4-1**.

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

```
cd helm/esdk
```

Step 4 Prepare the **values.yaml** file. Huawei CSI provides the **values.yaml** template file in the **helm/esdk** directory of the software package. You can also modify parameters according to **4.2.1.3 Parameters in the values.yaml File of Helm** to customize Huawei CSI.

Step 5 Perform the following configuration before the installation:

- If the container platform is Kubernetes, skip this step.
- If the container platform is OpenShift, perform the configuration in **Installation and Configuration on the OpenShift Platform**.
- If the container platform is Tanzu, perform the configuration in **Installation and Configuration on the Tanzu Platform**.

Step 6 Run the following command to update the storage backend CRD.

```
kubectl apply -f ./crds/backend/
```

Step 7 (Optional) Check snapshot-dependent components by following the instructions provided in **4.1.4 Checking Volume Snapshot-Dependent Components**. After confirming that the components are correct, run the following command to update the snapshot CRD. If **controller.snapshot.enabled** is set to **false** or the

Kubernetes version is earlier than v1.17, you can skip this step. For details, see **Table 4-5**.

```
kubectl apply -f ./crds/snapshot-crds/ --validate=false
```

Step 8 Run the following 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
```

The following is an example of the command output.

```
NAME: helm-huawei-csi
LAST DEPLOYED: Wed Jun  8 11:50:28 2022
NAMESPACE: huawei-csi
STATUS: deployed
REVISION: 1
TEST SUITE: None
```

Step 9 After the huawei-csi service is deployed, run the following command to check whether the service is started.

```
kubectl get pod -n huawei-csi
```

The following is an example of the command output. If the Pod status is **Running**, the installation is successful.

```
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

Installation and Configuration on the OpenShift Platform

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

Step 1 Run the following command to edit the **helm_scc.yaml** file.

```
vi helm_scc.yaml
```

Step 2 Modify the **helm_scc.yaml** file. In the following command output, **huawei-csi** indicates the created namespace. Replace it based on the actual situation.

```
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
```

Step 3 Run the following command to create a **SecurityContextConstraints** file.

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

----End

Installation and Configuration on the Tanzu Platform

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

Step 1 Go to the **helm/esdk** directory in the installation package, run the following command to open the configuration file, modify the file, and save the file. For details about the installation package directory, see **Table 4-1**.

```
vi values.yaml
```

Step 2 Modify the **kubeletConfigDir** parameter as follows:

```
# 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
```

----End

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

Step 1 Run the following command to create a file named **rbac.yaml**.

```
vi rbac.yaml
```

Step 2 Copy the following content to the **rbac.yaml** file, save the file, and exit.

```
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 3 Run the following command to create the RBAC permission.


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

----End

4.2.1.2 Installing Huawei CSI on the CCE or CCE Agile Platform

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

Creating a Helm Installation Package

The CCE or CCE Agile platform cannot directly install Huawei CSI using Helm. You need to manually create a Helm installation package and upload it to the chart list on the platform for installation.

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 4-1**.

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
```

Modify the following parameters:

```
# 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 following command to create a Helm installation package. This command will generate the installation package to the current path.

```
helm package ./esdk/ -d ./
```

----End

Installing Huawei CSI

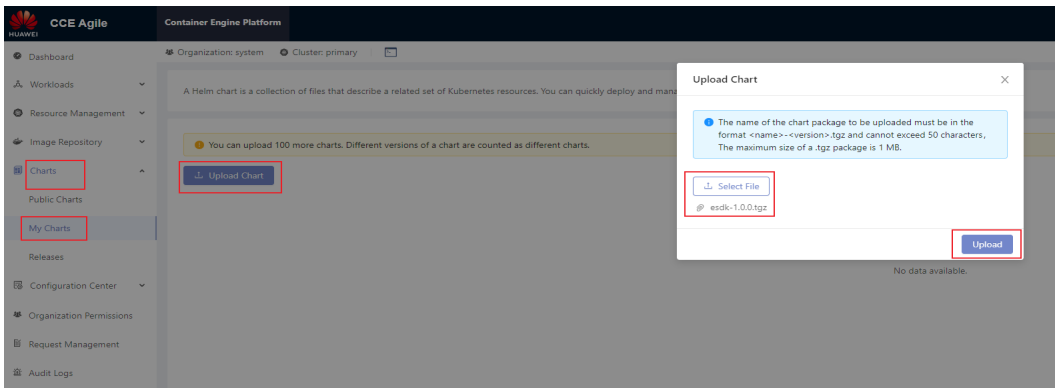
Step 1 Use a remote access tool, such as PuTTY, to log in to any master node where the CCE Agile platform is deployed through the management IP address.

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

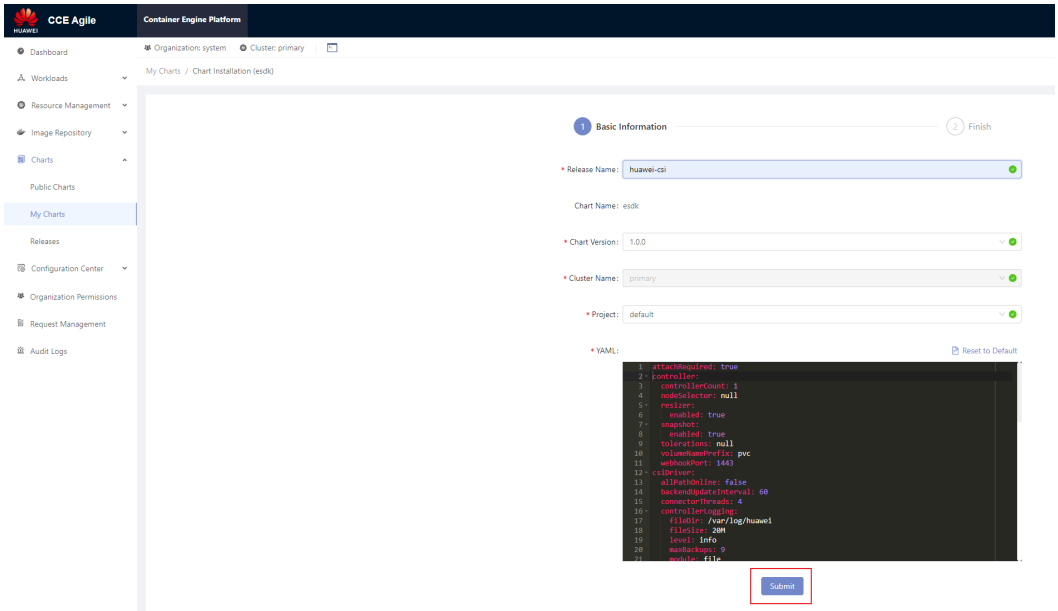
```
kubectl create namespace huawei-csi
```

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

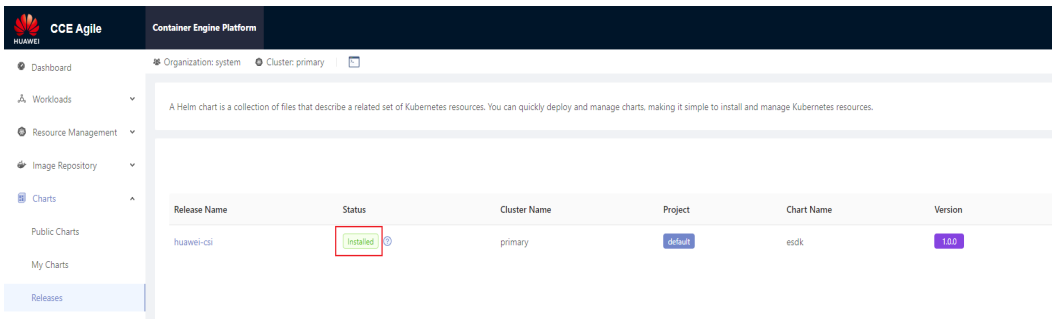
Step 4 On the home page, choose **Charts** > **My Charts** > **Upload Chart**. The **Upload Chart** dialog box is displayed. Import the exported Helm installation package to the CCE Agile platform.



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

4.2.1.3 Parameters in the values.yaml File of Helm

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.

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 4-4 images parameters

Parameter	Description	Mandatory	Default Value
images.huaweiCSIService	huawei-csi image.	Yes	huawei-csi:4.5.0
images.storageBackendSidecar	Huawei back-end management sidecar image.	Yes	storage-backend-sidecar:4.5.0
images.storageBackendController	Huawei back-end management controller image.	Yes	storage-backend-controller:4.5.0
images.huaweiCSIExtender	huawei-csi-extender image.	No	huawei-csi-extender:4.5.0
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**, **storageBackendController**, and **huaweiCSIExtender**, see **4.1.2 Uploading a Huawei CSI Image**. Use the name and version of the finally generated image.
- For details about other sidecar image parameters, see **4.1.7 Checking the Images on Which CSI Depends**. Use the name and version of the finally uploaded image.

controller Parameters

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

Table 4-5 controller parameters

Parameter	Description	Mandatory	Default Value	Remarks
controller.controllerCount	Number of huawei-csi-controller component copies.	Yes	1	If the Kubernetes version is earlier than v1.17, the huawei-csi-controller component can be deployed only in single-copy mode because the csi-provisioner sidecar image provided by the Kubernetes community does not support the **--leader-election** parameter. Therefore, if the Kubernetes version is earlier than v1.17, this parameter can only be set to **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**. It is recommended that the prefix contain no more than 20 characters. For details, see **Configuring the PV Name Prefix**. ● If the connected backend is OceanStor V5 SAN storage, it is recommended that the prefix contain a maximum of 5 characters. ● If the connected backend is OceanStor V5 NAS storage, the prefix can contain only lowercase letters, hyphens (-), and digits. ● If the connected backend is OceanStor Dorado or OceanStor storage, the prefix can contain only lowercase letters, hyphens (-), and digits. ● If the connected backend is OceanStor Pacific series storage, the prefix can

Parameter	Description	Mandatory	Default Value	Remarks
				contain a maximum of 58 characters, including only letters, digits, underscores (_), hyphens (-), and periods (.). • If the connected backend is FusionStorage Block, the prefix can contain a maximum of 58 characters, including only letters, digits, underscores (_), and hyphens (-).
controller.webhookPort	Port used by the webhook service.	Yes	4433	If a port conflict occurs, change the port number to an idle one.
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**.

Parameter	Description	Mandatory	Default Value	Remarks
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**.
controller.livenessProbePort	Liveness probe port of huawei-csi-controller, used for health check.	Yes	9808	If a port conflict occurs, change the port number to an idle one.
controller.csiExtender.volumeModify.enabled	Whether to enable the PVC change feature.	No	false	If you want to use PVC change-related functions, enable this feature.
controller.csiExtender.volumeModify.retryBaseDelay	Minimum retry interval when a PVC change fails to be created.	No	5s	The default value is recommended.
controller.csiExtender.volumeModify.retryMaxDelay	Maximum retry interval when a PVC change fails to be created.	No	5m	The default value is recommended.
controller.csiExtender.volumeModify.reconcileDelay	Interval for reconciling VolumeModifyClaim objects.	No	1s	The default value is recommended.

NOTE

If **controller.snapshot.enabled** is set to **true**, you need to install the volume snapshot CRD resource in the **helm/crd/snapshot-crds** directory.

node Parameters

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

Table 4-6 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**.

Parameter	Description	Mandatory	Default Value	Remarks
node.livenessProbePort	Liveness probe port of huawei-csi-node, used for health check.	Yes	9800	If a port conflict occurs, change the port number to an idle one.
node.kubeletVolumeDevicesDirName	Name of the directory where a block device is mounted to kubelet.	No	volumeDevices	After a block device is successfully mounted, the directory structure of the mount path is as follows: `/var/lib/kubelet/plugins/kubernetes.io/csi/{kubeletVolumeDevices-DirName}/publish/{specName}/{podUID}`

csiDriver Parameters

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

Table 4-7 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.

Parameter	Description	Mandatory	Default Value	Remarks
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.
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	-

Parameter	Description	Mandatory	Default Value	Remarks
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	-
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
csiDriver.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 mounting condition is met.	This parameter is mandatory when **csiDriver.scsiMultipathType** is set to **DM-multipath**.	false	-
csiDriver.backendUpdateInterval	Interval for updating backend capabilities. The value ranges from 60 to 600 seconds.	Yes	60	-

Parameter	Description	Mandatory	Default Value	Remarks
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.
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.
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	-

NOTICE

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.

Other Parameters

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

Table 4-8 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	-
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	-

Parameter	Description	Mandatory	Default Value	Remarks
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**.
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.

Parameter	Description	Mandatory	Default Value	Remarks
leaderElection.leaseDuration	Leader duration.	No	8s	This parameter takes effect only in the multi-controller scenario.
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
```

4.2.2 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.

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 Run the following command to create a namespace.

```
kubectl create ns huawei-csi
```

Step 4 Go to the **manual/esdk** working directory. For details about the path, see **Table 4-1**.

```
cd manual/esdk
```

Step 5 Run the following command to update the storage backend CRD.

```
kubectl apply -f ./crds/backend/
```

Step 6 (Optional) Check snapshot-dependent components by following the instructions provided in **4.1.4 Checking Volume Snapshot-Dependent Components**. After

confirming that the components are correct, run the following command to update the snapshot CRD. If the Kubernetes version is earlier than v1.17, skip this step.

```
kubectl apply -f ./crds/snapshot-crds/ --validate=false
```

Step 7 (Optional) Run the following command to install CSIDriver. If the CSIDriver feature is not used, you can skip this step. For details, see the **CSIDriver** feature.

```
kubectl apply -f ./deploy/csidriver.yaml
```

Step 8 Run the following command to install the huawei-csi-controller service.

NOTE

If the Kubernetes version is earlier than v1.17, modify the *./deploy/huawei-csi-controller.yaml* file as follows:

- If the Kubernetes version is earlier than v1.17, the snapshot feature is not supported. In this case, delete the snapshot-related container configurations items **csi-snapshotter** and **snapshot-controller**.
- If the Kubernetes version is earlier than v1.17, the csi-provisioner sidecar image provided by the Kubernetes community does not support the **--leader-election** parameter. Therefore, the **leader-election** parameter of the csi-provisioner container is deleted and only single-copy deployment is supported.
- Modify the dependent image version based on the version requirements in **4.1.7 Checking the Images on Which CSI Depends**.

```
kubectl apply -f ./deploy/huawei-csi-controller.yaml
```

Step 9 Run the following command to install the huawei-csi-node service.

```
kubectl apply -f ./deploy/huawei-csi-node.yaml
```

Step 10 Run the following command to check whether the services are started.

```
kubectl get pod -n huawei-csi
```

The following is an example of the command output. If the Pod status is **Running**, the installation is successful.

```
NAME                                  READY   STATUS    RESTARTS   AGE
huawei-csi-controller-68745d489c-v5xkj   9/9     Running   0          13m
huawei-csi-node-4hbqp                 3/3     Running   0          13m
huawei-csi-node-f7dkf                 3/3     Running   0          13m
huawei-csi-node-xrntc                 3/3     Running   0          13m
```

----End

NOTE

In the multi-copy controller deployment scenario, you can modify the **spec.replica** field of the Deployment resource in the *./deploy/huawei-csi-controller.yaml* file to specify the number of copies. After the modification, run the following command for the modification to take effect.

```
kubectl apply -f ./deploy/huawei-csi-controller.yaml
```

4.3 Uninstalling Huawei CSI

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

NOTICE

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.

4.3.1 Uninstalling Huawei CSI Using Helm

4.3.1.1 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 following 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 uninstall the huawei-csi-controller, huawei-csi-node, and RBAC resources of Huawei CSI.

```
helm uninstall helm-huawei-csi -n huawei-csi
```

After the uninstallation command is executed, you need to check whether the uninstallation is successful. In the preceding command, *huawei-csi* indicates the namespace where the chart is located.

```
helm list -n huawei-csi
```

The following is an example of the command output. If the command output is empty, the service is successfully uninstalled.

```
NAME    NAMESPACE       REVISION        UPDATED STATUS  CHART   APP VERSION
```

Step 3 Uninstall the huawei-csi-host-info object. For details, see **Uninstalling the huawei-csi-host-info Object**.

Step 4 Uninstall the webhook resource. For details, see **Uninstalling a Webhook Resource**.

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

Step 6 (Optional) Uninstall the Lease resource. For details, see **Uninstalling a Lease Resource**.

----End

4.3.1.2 Uninstalling Huawei CSI on CCE or CCE Agile

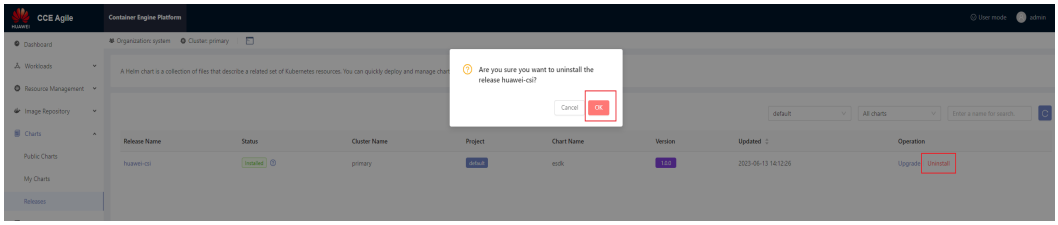
This section describes how to uninstall Huawei CSI on the CCE or CCE Agile platform. The following uses CCE Agile v22.3.2 as an example.

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 Uninstall the huawei-csi-host-info object. For details, see **Uninstalling the huawei-csi-host-info Object**.

Step 5 Uninstall the webhook resource. For details, see **Uninstalling a Webhook Resource**.

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

----End

4.3.1.3 Uninstalling CSI-Dependent Component Services

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

Uninstalling the huawei-csi-host-info Object

Secret object **huawei-csi-host-info** stores the initiator information about each node in the cluster, for example, iSCSI initiators. When you run the **helm uninstall** command, the resource will not be uninstalled. To uninstall the resource, perform the following steps:

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 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 uninstalled.

```
kubectl get secret huawei-csi-host-info -n huawei-csi
```

The following is an example of the command output. If **NotFound** is displayed in the command output, the **huawei-csi-host-info** object is successfully uninstalled.

```
Error from server (NotFound): secrets "huawei-csi-host-info" not found
```

----End

Uninstalling a Webhook Resource

The webhook resource named **storage-backend-controller.xuanwu.huawei.io** is used to verify the backend key information and connectivity with the storage. When you run the **helm uninstall** command, the resource will not be uninstalled. To uninstall the resource, perform the following steps:

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 storage-backend-controller.xuanwu.huawei.io
```

The following is an example of the command output.

```
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 storage-backend-controller.xuanwu.huawei.io
```

----End

Uninstalling the Snapshot-Dependent Component Service

NOTICE

- 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

Uninstalling a Lease Resource

If the value of the **controller.controllerCount** configuration item in the **values.yaml** file is greater than 1, huawei-csi-controller will be deployed in multi-copy mode. The multiple copies of huawei-csi-controller are implemented using the LeaderElection mechanism of Kubernetes. This mechanism creates a Lease object to store the current Holder information. When you run the **helm uninstall** command, the resource will not be uninstalled. To uninstall the resource, perform the following steps. If the value of **controller.controllerCount** is **1**, you can skip the following steps. For details about the configuration item, see **Table 4-5**.

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 Lease information.

```
kubectl get lease -n huawei-csi
```

The following is an example of the command output.

```
NAME                                        HOLDER    AGE
csi-huawei-com                              node-1    24d
external-attacher-leader-csi-huawei-com     node-1    24d
external-resizer-csi-huawei-com             node-1    24d
external-snapshotter-leader-csi-huawei-com  node-1    24d
snapshot-controller-leader                  node-1    24d
storage-backend-controller                  node-1    24d
huawei-csi-extender                         node-1    24d
```

Step 3 Run the following command to uninstall the Lease resource.

```
kubectl delete lease -n huawei-csi csi-huawei-com external-attacher-leader-csi-huawei-com external-resizer-
csi-huawei-com external-snapshotter-leader-csi-
```

Step 4 Run the following command to check whether the uninstallation is successful.

```
kubectl get lease -n huawei-csi
```

The following is an example of the command output. If the command output is empty, the uninstallation is successful.

```
No resources found in huawei-csi namespace.
```

----End

4.3.2 Manually Uninstalling Huawei CSI

This section describes how to manually uninstall Huawei CSI.

NOTICE

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.

Uninstalling the huawei-csi-node Service

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 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

Uninstalling the huawei-csi-controller Service

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 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

Uninstalling the csidriver Object

If **the CSIDriver feature is not used during installation**, you can skip the following steps.

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 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

Deleting the RBAC Permission

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

Uninstalling Other Resources

Step 1 Uninstall the huawei-csi-host-info object. For details, see **Uninstalling the huawei-csi-host-info Object**.

Step 2 Uninstall the webhook resource. For details, see **Uninstalling a Webhook Resource**.

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

Step 4 (Optional) Uninstall the Lease resource. For details, see **Uninstalling a Lease Resource**.

----End

4.4 Upgrading or Rolling Back Huawei CSI

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

NOTE

In the current version, resource requests and limits are added to Huawei CSI. For details, see **10.6 Huawei CSI Resource Management**.

4.4.1 Upgrading or Rolling Back Huawei CSI Using Helm

To upgrade Huawei CSI from 2.*x* to 4.5.0, uninstall it by referring to the user guide of the earlier version and install Huawei CSI by referring to **4.2.1 Installing Huawei CSI Using Helm**.

To upgrade Huawei CSI from 2.*x* or 3.*x* to 4.5.0, see **4.4.1.1.1 Upgrading from 2.x or 3.x to 4.x**.

To upgrade Huawei CSI from 4.*x* to 4.5.0, see **4.4.1.1.2 Upgrading Huawei CSI on Kubernetes, OpenShift, and Tanzu**.

4.4.1.1 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.

NOTICE

- 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.
- After an upgrade from 2.*x*, 3.*x*, or 4.*x* to 4.5.0, a Pod that has been provisioned in the source version may fail to be mounted again. For details, see **4.4.1.1.1 Upgrading from 2.x or 3.x to 4.x**.
- 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.

4.4.1.1.1 Upgrading from 2.x or 3.x to 4.x

NOTICE

In CSI 2.*x* or 3.*x*, when block storage is used, the mapping with storage is set up in the huawei-csi-node service. Therefore, the huawei-csi-node service needs to communicate with the storage management network. Because the huawei-csi-node service is deployed as a DaemonSet, the huawei-csi-node service is deployed on each node in the cluster. As a result, in a large-scale cluster, each huawei-csi-node service sends requests to the storage and the number of storage connections may be fully occupied. Accordingly, huawei-csi-node cannot provide services properly.

In CSI 4.*x*, the deployment model is optimized. The setup of the mapping with storage is migrated to the huawei-csi-controller service and the huawei-csi-node service does not need to communicate with the storage management network. This reduces the networking complexity of Huawei CSI. In addition, the huawei-csi-controller service is deployed as a Deployment. The number of copies is set based on the customer's reliability requirements. Generally, the number of copies ranges from 1 to 3. Therefore, the number of connections between Huawei CSI and storage is greatly reduced, so that Huawei CSI can connect to a large-scale cluster.

This change may cause a problem. That is, if a new mount process is generated after CSI is upgraded to 4.*x* but with workloads provisioned using 2.*x* or 3.*x* and the Container Orchestration (CO) system does not invoke the huawei-csi-controller service provided by Huawei CSI, the mounting will fail. For details, see **9.4.6 A Pod Fails to Be Created and Message "publishInfo doesn't exist" Is Displayed in the Events Log**.

Backing Up Storage Backend Configurations

If you have evaluated the risks mentioned in the preceding notice and need to upgrade CSI from 2.*x* or 3.*x* to 4.5.0, perform the following steps to back up storage backend configurations:

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 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
```

----End

Upgrading Huawei CSI

Perform the upgrade according to the procedure described in **Upgrading Huawei CSI**.

Configuring the Storage Backend

Configure the storage backend by following the instructions in **5.1 Managing Storage Backends** according to the backend information backed up in **Backing Up Storage Backend Configurations**. After the storage backend is successfully configured, perform operations according to the risk handling methods described in the preceding notice to prevent problems during Pod failover.

4.4.1.1.2 Upgrading Huawei CSI on Kubernetes, OpenShift, and Tanzu

Prerequisites

- Huawei CSI of an earlier version is installed using Helm.
- A Huawei CSI image of a new version has been created and uploaded to the image repository or imported to all nodes by following the instructions provided in **4.1.2 Uploading a Huawei CSI Image**.

Upgrading Huawei CSI

If CSI of an earlier version is deployed using Helm, perform the following steps to upgrade Huawei CSI.

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 Copy the CSI component package of the target version to any directory on the master node.

Step 3 Go to the **helm/esdk** working directory. For the directory path, see **Table 4-1**.

```
cd helm/esdk
```

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

```
kubectl apply -f ./crds/backend/
```

Step 5 (Optional) Check snapshot-dependent components by following the instructions provided in **4.1.4 Checking Volume Snapshot-Dependent Components**. After confirming that the components are correct, run the **kubectl apply -f** *./crds/snapshot-crds/* **--validate=false** command to update the snapshot CRD. If **controller.snapshot.enabled** is set to **false** or the Kubernetes version is earlier than v1.17, you can skip this step. For details, see **Table 4-5**.

```
kubectl apply -f ./crds/snapshot-crds/ --validate=false
```

Step 6 Run the following command to obtain the original service configuration file. **helm-huawei-csi** indicates the Helm chart name specified during the installation

of the earlier version, and **huawei-csi** indicates the Helm chart namespace specified during the installation of the earlier version.

```
helm get values helm-huawei-csi -n huawei-csi -a > ./update-values.yaml
```

Step 7 Run the **vi update-values.yaml** command to open the file obtained in **Step 6**, modify the **images** configuration items, and update the image to the latest version. For details about the parameters to be modified, see **Table 4-9**.

Table 4-9 images configuration items

Parameter	Description	New Value
images.huaweiCSIService	huawei-csi image.	huawei-csi:4.5.0
images.storageBackendSidecar	Image used by Huawei backends to manage storageBackendContent resources.	storage-backend-sidecar:4.5.0
images.storageBackendController	Image used by Huawei backends to manage storageBackendClaim resources.	storage-backend-controller:4.5.0
images.huaweiCSIExtender	huawei-csi-extender image.	huawei-csi-extender:4.5.0
images.sidecar.livenessProbe	**livenessprobe** sidecar image.	k8s.gcr.io/sig-storage/livenessprobe:v2.5.0
images.sidecar.provisioner	**csi-provisioner** sidecar image.	k8s.gcr.io/sig-storage/csi-provisioner:v3.0.0
images.sidecar.attacher	**csi-attacher** sidecar image.	k8s.gcr.io/sig-storage/csi-attacher:v3.4.0
images.sidecar.resizer	**csi-resizer** sidecar image.	k8s.gcr.io/sig-storage/csi-resizer:v1.4.0
images.sidecar.snapshotter	**csi-snapshotter** sidecar image.	k8s.gcr.io/sig-storage/csi-snapshotter:v4.2.1
images.sidecar.snapshotController	**snapshot-controller** sidecar image.	k8s.gcr.io/sig-storage/snapshot-controller:v4.2.1
images.sidecar.registrar	**csi-node-driver-registrar** sidecar image.	k8s.gcr.io/sig-storage/csi-node-driver-registrar:v2.3.0

Step 8 (Optional) If you need to update configuration items or add configuration information during the upgrade, modify the configuration information in the **update-values.yaml** file by referring to **4.2.1.3 Parameters in the values.yaml File of Helm**.

> **NOTE**
>
> During the upgrade, if the **update-values.yaml** and **values.yaml** configuration files contain the same configuration item, the configuration in the **update-values.yaml** file takes effect preferentially.

Step 9 Run the following command to upgrade Huawei CSI. In the following command, **helm-huawei-csi** indicates the specified Helm chart name, **huawei-csi** indicates the specified Helm chart namespace, and **update-values.yaml** indicates the file obtained in **Step 6**.

```
helm upgrade helm-huawei-csi ./ -n huawei-csi -f ./values.yaml -f ./update-values.yaml
```

Step 10 After the huawei-csi service is deployed, run the following command to check whether the service is started.

```
kubectl get pod -n huawei-csi
```

The following is an example of the command output. If the Pod status is **Running**, the service is started successfully.

```
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

4.4.1.1.3 Upgrading Huawei CSI on CCE or CCE Agile

Prerequisites

You have downloaded the CSI software package of a new version.

Procedure

Step 1 Uninstall CSI. For details, see **4.3.1.2 Uninstalling Huawei CSI on CCE or CCE Agile**.

Step 2 Install CSI of the new version. For details, see **4.2.1.2 Installing Huawei CSI on the CCE or CCE Agile Platform**.

----End

4.4.1.2 Rolling Back Huawei CSI

If CSI fails to be upgraded from 2.*x* or 3.*x* to 4.5.0 and needs to be rolled back, uninstall CSI by referring to **4.3.1 Uninstalling Huawei CSI Using Helm** and then download and install CSI of the source version.

NOTICE

- 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.

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

Prerequisites

- CSI has been updated using Helm 3.

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 **helm/esdk** working directory. For the directory path, see **Table 4-1**.

```
cd helm/esdk
```

Step 3 Run the following command to query the historical versions of the CSI services deployed using Helm.

```
helm history helm-huawei-csi -n huawei-csi
```

The following is an example of the command output.

```
REVISION        UPDATED                         STATUS          CHART           APP VERSION     DESCRIPTION
1               Mon Jan  8 04:15:40 2024        superseded      esdk-4.4.0      4.4.0           Install complete
2               Mon Jan  8 04:16:12 2024        deployed        esdk-4.5.0      4.5.0           Upgrade complete
```

Step 4 Run the following command to roll back the CSI services to the specified version.

In the preceding command, *revision-number* indicates a version number queried in **Step 3**. For example, the version is **1**.

```
helm rollback helm-huawei-csi -n huawei-csi 1
```

The following is an example of the command output. If **Rollback was a success** is displayed in the command output, the CSI services are successfully rolled back to the specified version.

```
Rollback was a success! Happy Helming!
```

----End

4.4.1.2.2 Rolling Back Huawei CSI on CCE or CCE Agile

NOTICE

- 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 **Procedure**.

Step 3 Reinstall CSI of the source version. For details, see **4.2.1.2 Installing Huawei CSI on the CCE or CCE Agile Platform**.

----End

4.4.2 Manual Upgrade/Rollback

4.4.2.1 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.

NOTICE

- 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 4.5.0

To upgrade CSI from 2.*x* or 3.*x* to 4.5.0, 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 following 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 **4.3.2 Manually Uninstalling Huawei CSI**.

Step 4 Install CSI of the current version. For details, see **4.2.2 Manually Installing Huawei CSI**.

Step 5 Install the backend information backed up in **Step 2** according to **5.1 Managing Storage Backends**.

----End

Upgrading CSI from 4.x to 4.5.0

To upgrade CSI from 4.*x* to 4.5.0, 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 **4.3.2 Manually Uninstalling Huawei CSI**.

Step 3 Install CSI of the current version. For details, see **4.2.2 Manually Installing Huawei CSI**.

----End

4.4.2.2 Rolling Back Huawei CSI

Uninstall CSI by referring to **4.3.2 Manually Uninstalling Huawei CSI**, and then download and install CSI of the source version.

NOTICE

- 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 **4.3.2 Manually Uninstalling Huawei CSI**.

Step 3 Reinstall CSI of the source version. For details, see **4.2.2 Manually Installing Huawei CSI**.

----End

5 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 chapter describes how to use the oceanctl tool to manage storage backends, including creating, querying, updating, and deleting backends.

Description of the oceanctl Tool

- 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 depends on **kubectl** (for the Kubernetes platform) or **oc** (for the OpenShift platform) commands. Therefore, you need to run the tool on a node where **kubectl** or **oc** commands can be executed.
- By default, the user who runs **oceanctl** commands must have the read and write permissions on the **/var/log** directory. If you do not have the permissions on the directory, run the **--log-dir=/path/to/custom** command to specify a directory on which you have the permissions as the log file directory.
- **huawei-csi** is the default namespace used by oceanctl to create a backend.
- For details about **oceanctl** commands, see **5.3 Description of oceanctl Commands**.

5.1 Managing Storage Backends

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 **5.1.1.1 Examples of Storage Backend Configuration Files in Typical Scenarios**.

If the exported configmap.json file exists, create a storage backend by referring to **5.1.1 Creating a Storage Backend**.

5.1.1 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.
3. If a storage backend is connected to a vStore, the vStore name cannot be changed after the storage backend is created.

Procedure

Step 1 Prepare the backend configuration file, for example, **backend.yaml**. For details, see **5.1.1.1 Examples of Storage Backend Configuration Files in Typical Scenarios**. 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
```

The following is an example of the command output.

```
NUMBER  CONFIGURED    NAME        STORAGE             URLS
1       false         backend-1   oceanstor-san       https://192.168.129.157:8088
2       false         backend-2   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-san       https://192.168.129.157:8088
2       false         backend-2   oceanstor-san       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
```

The following is an example of the command output. If the backend status is **Bound**, the creation is successful.

```
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

5.1.1.1 Examples of Storage Backend Configuration Files in Typical Scenarios

For details about the backend configuration in typical scenarios, see the following examples. For details about the parameter configuration, see **5.1.1.2 Storage Backend Parameters**.

- **Configuring a Storage Backend of the iSCSI Type**
- **Configuring a Storage Backend of the FC Type**
- **Configuring a Storage Backend of the NVMe over RoCE Type**
- **Configuring a Storage Backend of the NVMe over FC Type**
- **Configuring a Storage Backend of the NFS Type**
- **Configuring a Storage Backend of the SCSI Type**
- **Configuring a Storage Backend of the DPC Type**
- **Configuring Storage Backends of the Dtree Type**
- **Configuring Storage Backends of the HyperMetro Type**

Configuring a Storage Backend of the iSCSI Type

NOTE

If you want to use the iSCSI protocol, ensure that the iSCSI client has been installed on the host before installing Huawei CSI. You can check whether the client has been installed on the host by referring to **4.1.6 Checking the Status of Host-Dependent Software**. If the iSCSI client is not installed, restart the huawei-csi-node service after installing the iSCSI client. During the restart, do not use Huawei CSI to create new resources or mount or unmount an existing PVC. The following command is used as an example:

```
kubectl delete pods -n huawei-csi -l app=huawei-csi-node
```

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

NOTE

If you want to use the FC protocol, ensure that the FC network between the host and the storage device is connected before installing Huawei CSI. If the FC network is not connected, connect the FC network and then restart the huawei-csi-node service. During the restart, do not use Huawei CSI to create new resources or mount or unmount an existing PVC. The following command is used as an example:

```
kubectl delete pods -n huawei-csi -l app=huawei-csi-node
```

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

NOTE

If you want to use the NVMe over RoCE protocol, ensure that the NVMe over RoCE network between the host and the storage device is connected before installing Huawei CSI. If the NVMe over RoCE network is not connected, connect the NVMe over RoCE network and then restart the huawei-csi-node service. During the restart, do not use Huawei CSI to create new resources or mount or unmount an existing PVC. The following command is used as an example:

```
kubectl delete pods -n huawei-csi -l app=huawei-csi-node
```

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

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 active-active (AA) 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.
- Except NAS HyperMetro backends, the management URLs of other backends cannot be the URL of a logical management port of a vStore that has established the HyperMetro relationship.
- When a HyperMetro storage backend is used, do not provision common file systems. Otherwise, services may be interrupted in logical port failover scenarios.

CSI allows you to connect to OceanStor or OceanStor Dorado and provision HyperMetro volumes of the NFS type on the storage side. You need to configure storage backends that work in HyperMetro mode. The procedure is as follows: Create two configuration files and create backends one by one.

This example shows how to configure backends of the HyperMetro type for Huawei OceanStor or OceanStor Dorado. 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"
```

5.1.1.2 Storage Backend Parameters

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

Table 5-1 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.

Parameter	Description	Mandatory	Default Value	Remarks
namespace	Namespace.	No	-	The storage backend must be in the same namespace as Huawei CSI.
vstoreName	vStore name on the storage side. This parameter needs to be specified when the connected backend is OceanStor 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 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 or OceanStor Dorado storage 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 has been installed on the connected compute node. The tool version must be 1.*x* and not earlier than 1.9. ● 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 **7.1.1 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 **7.2.1 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.

5.1.2 Querying a Storage Backend

Run the **oceanctl** commands in **Querying a Storage Backend** to query the storage backend information.

5.1.3 Updating a Storage Backend

NOTICE

- When oceanctl is used to update storage backend information, only the storage backend password 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 **9.2.3 An Account Is Locked After the Password Is Updated on the Storage Device**.

5.1.3.1 Updating the Password of a Storage Backend Using oceanctl

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
```

The following is an example of the command output.

```
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
```

Enter the user name and new password as prompted:

```
Please enter this backend user name:admin
Please enter this backend password:

backend/backend-1 updated
```

----End

5.1.3.2 Manually Updating a Storage Backend

NOTE

- PVC provisioning must be based on a configured storage backend. Therefore, if a PVC has been provisioned on a storage backend, do not change the storage backend.
- The name uniquely identifies a storage backend. The name of a storage backend with a PVC provisioned cannot be changed.
- After a storage backend is modified, the new configuration applies only to volumes to be provisioned.
- Do not perform volume management operations during the modification of a storage backend.

Procedure

Step 1 Delete the storage backend to be modified. For details, see **5.1.4 Deleting a Storage Backend**.

Step 2 Create a storage backend with the same name. For details, see **5.1.1 Creating a Storage Backend**. The storage backend name cannot be changed.

----End

5.1.4 Deleting a Storage Backend

NOTICE

Do not delete a storage backend when a volume management operation is being performed on it.

Example of Deleting a Backend

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

```
oceanctl get backend
```

The following is an example of the command output.

```
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
```

Step 3 Run the following command to check the deletion result.

```
oceanctl get backend backend-1
```

The following is an example of the command output. If **not found** is displayed, the deletion is successful.

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

----End

5.2 (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.

5.2.1 Creating a Certificate for a Storage Backend

Prerequisites

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

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
```

The following is an example of the command output.

```
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
```

Step 4 Check the certificate creation result.

```
oceanctl get cert -b backend-1
```

The following is an example of the command output.

```
NAMESPACE    NAME    BOUNDBACKEND
huawei-csi   cert-1  backend-1
```

----End

5.2.2 Querying a Storage Backend Certificate

Query storage backend certificates using the commands in **Querying a Storage Backend Certificate**.

5.2.3 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 **5.2.4 Deleting a Storage Backend Certificate**.

Procedure

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

```
oceanctl get backend
```

The following is an example of the command output.

```
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 check whether the specified storage backend has a certificate.

```
oceanctl get cert -b backend-1
```

The following is an example of the command output.

```
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
```

----End

5.2.4 Deleting a Storage Backend Certificate

Procedure

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

```
oceanctl get backend
```

The following is an example of the command output.

```
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
```

The following is an example of the command output.

```
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
```

Step 4 Check the deletion result.

```
oceanctl get cert -b backend-1
```

The following is an example of the command output. If **no cert found** is displayed, the deletion is successful.

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

----End

5.3 Description of oceanctl Commands

Obtaining Help Information

- Obtain the oceanctl help information.
  ```
  oceanctl --help
  ```
- Check the oceanctl version.
  ```
  oceanctl version
  ```
- Specify the custom log file directory. The following example describes how to check the oceanctl version.
  ```
  oceanctl version --log-dir=/path/to/custom
  ```

Creating a Storage Backend

- 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 underscores (_). Do not run this command unless necessary.
  ```
  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 or CCE Agile platform.

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

Querying a Storage Backend

- 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
  ```

Updating a Storage Backend

- 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
  ```

Deleting a Storage Backend

- 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>
  ```

Creating a Storage Backend Certificate

- 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>
  ```

Querying a Storage Backend Certificate

- 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>
  ```

Updating a Storage Backend Certificate

- 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
  ```

Deleting a Storage Backend Certificate

- 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>
  ```

6 Using Huawei CSI

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

NOTICE

- Do not delete a storage backend when using Huawei CSI to manage volumes.
- 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.

6.1 Managing a 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.

6.1.1 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 3-5** or **Table 3-9**.

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.

6.1.1.1 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

Configuring a StorageClass

Step 1 Create a StorageClass configuration file, for example, **mysc.yaml**, based on service requirements by referring to **6.1.1.1.1 StorageClass Configuration Examples in Typical Dynamic Volume Provisioning Scenarios** and **6.1.1.1.2 StorageClass Parameters for Dynamic Volume Provisioning**.

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

```
kubectl apply -f mysc.yaml
```

Step 3 Run the following command to view the information about the created StorageClass.

```
kubectl get sc mysc
```

The following is an example of the command output.

```
NAME   PROVISIONER      RECLAIMPOLICY   VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
mysc   csi.huawei.com   Delete          Immediate           true                   8s
```

----End

Configuring a PVC

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
```

Step 3 After a period of time, run the following command to view the information about the created PVC.

```
kubectl get pvc mypvc
```

The following is an example of the command output. If the PVC status is **Bound**, the PVC has been created and can be used by a Pod.

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

NOTICE

- After the PVC is created, if the PVC is in the **Pending** state after a long time (for example, one minute), refer to **9.3.1 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

Using a PVC

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
```

6.1.1.1.1 StorageClass Configuration Examples in Typical Dynamic Volume Provisioning Scenarios

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.

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**.

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 or 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: "san-181" # Enter the storage backend name.
  pool: "pool001" # Enter the storage pool name
  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: "san-181" # Enter the storage backend name.
  pool: "pool001" # Enter the storage pool name
  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"
mountOptions:
  - nfsvers=4.1 # Specify the version 4.1 for NFS mounting.
```

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 # Set the volume type to 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: "*"
mountOptions:
  - acl # Set the authentication parameter.
  - cnflush # Set the asynchronous disk flushing mode.
```

Setting an Application Type in a StorageClass

When a container uses a LUN of OceanStor Dorado 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 # Set the application type.
```


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}' # Configure the soft quota.
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 6-2**. In this example, the backend is OceanStor Dorado. 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}' # Configure QoS.
```

NOTE

- vStore users of 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.

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" # Provision HyperMetro volumes.
  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 active-active (AA) mode. For details about the configuration operation, see the product documentation of the corresponding storage model.
- If a storage device is faulty, the logical management port may fail over. In this case, you need to manually clear the corresponding storage resources after deleting the NAS HyperMetro volume.

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"  # Only NAS storage supports this parameter.
  allSquash: "no_all_squash"    # Only NAS storage supports this parameter.
```

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
```

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

```
kubectl get sc
```

The following is an example of the command output.

```
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

NOTICE

Pay attention to the following when using a StorageClass:

- 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 or CCE Agile Platform

Create a StorageClass of the NAS type on the CCE or 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 or 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
```

6.1.1.1.2 StorageClass Parameters for Dynamic Volume Provisioning

Table 6-1 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.

Parameter	Description	Mandatory	Default Value	Remarks
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.
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.

Parameter	Description	Mandatory	Default Value	Remarks
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**.
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/ OceanStor Dorado V3 does not support **thick**.

Parameter	Description	Mandatory	Default Value	Remarks
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**.
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**

Parameter	Description	Mandatory	Default Value	Remarks
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 **6.1.3 Cloning a PVC** or **6.1.4 Creating a PVC Using a Snapshot**.
parameters.applicationType	Application type name for creating a LUN or NAS when the backend is OceanStor Dorado.	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.

Parameter	Description	Mandatory	Default Value	Remarks
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 6-2**.
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 6-3**.

Parameter	Description	Mandatory	Default Value	Remarks
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. If the storage backend is a HyperMetro backend, the value must be **true**. • **"false"**: The created volume is a common volume.	Conditionally mandatory	false	When the used backend is a HyperMetro backend and a HyperMetro volume needs to be provisioned, set this parameter to **true**. If this parameter is set to **false**, services may be interrupted if the logical management port connected to the backend fails over.
parameters.metroPairSyncSpeed	Data synchronization speed of a HyperMetro pair. The value ranges from 1 to 4. The value can be: • **1**: low • **2**: medium • **3**: high • **4**: highest	No	-	The configuration takes effect when a HyperMetro volume is created. Note: • If this parameter is not configured, the storage speed of the HyperMetro pair is determined by the storage device. • The highest synchronization speed may increase the host latency.

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, OceanStor, and OceanStor Pacific 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.accesskrb5	Configures the krb5 security protocol. • **read_only**: read-only • **read_write**: read and write • **none**: no permission	No	-	During mounting, you can specify the **sec** parameter in **mountOptions**.
parameters.accesskrb5i	Configures the krb5i security protocol. • **read_only**: read-only • **read_write**: read and write • **none**: no permission	No	-	During mounting, you can specify the **sec** parameter in **mountOptions**.
parameters.accesskrb5p	Configures the krb5p security protocol. • **read_only**: read-only • **read_write**: read and write • **none**: no permission	No	-	During mounting, you can specify the **sec** parameter in **mountOptions**.

Parameter	Description	Mandatory	Default Value	Remarks
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.
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.

Parameter	Description	Mandatory	Default Value	Remarks
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**, **4.1**, or **4.2**.	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, 4.1, and 4.2 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, 4.2. If the 4.0 protocol 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.

Parameter	Description	Mandatory	Default Value	Remarks
mountOptions.sec	Kerberos 5 protocol for mounting NFS file systems.	No	-	• If Kerberos 5 is used, set this parameter to **krb5**. • If Kerberos 5i is used, set this parameter to **krb5i**. • If Kerberos 5p is used, set this parameter to **krb5p**. • Kerberos supports only NFSv4.0 or NFSv4.1.
mountOptions.proto	Transmission protocol used for NFS mounting. The value can be **rdma**.	No	-	• Ensure that NFS over RDMA is enabled on the storage system. • NAS storage of OceanStor Dorado 6.1.7 or later is supported.
mountOptions.port	Protocol port used for NFS mounting.	Conditionally mandatory	-	If the transmission protocol is **rdma**, set this parameter to **20049**.
mountOptions.discard	Automatically triggers the Trim or Discard operation when a file system is mounted. This operation instructs a block device to release unused blocks.	No	-	The xfs and ext4 file systems are supported.

Table 6-2 Supported QoS configurations

Storage Type	Parameter	Description	Remarks
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.
OceanStor Dorado/ OceanStor	IOTYPE	Read/write type.	The value can be: ● **2**: read and write I/Os

Storage Type	Parameter	Description	Remarks
	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.

Table 6-3 Supported quota configurations

Parameter	Description	Remarks
spaceQuota	File quota type.	This parameter is mandatory. Only **softQuota** or **hardQuota** can be configured.

Parameter	Description	Remarks
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.

6.1.1.1.3 PVC Parameters for Dynamic Volume Provisioning

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 6-4 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.
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.

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	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 6-5**. If XFS file systems are used, see **Table 6-6**. 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 **8.6 Enabling the ReadWriteOncePod Feature Gate**. • The support for RWX is as follows: – **NAS storage**: supported by all volumes – **SAN storage**: supported only by volumes whose **volumeMode** is set to **Block**

Table 6-5 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 6-6 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

6.1.1.2 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.

NOTE

In scenarios where multiple Kubernetes clusters are deployed, when Manage Volume Provisioning is used to manage the same storage resource, management operations performed on the PVC corresponding to the resource in any cluster will not be synchronized to other clusters.

For example, when you expand the capacity of a PVC in a cluster, the capacity of the corresponding PVC in other clusters will not be automatically expanded. In this case, you need to manually expand the capacity in other clusters by running the expansion commands in **6.1.2 Expanding the Capacity of a PVC**.

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.

Configuring a StorageClass

Step 1 Create a StorageClass configuration file, for example, **mysc.yaml**, based on service requirements by referring to **6.1.1.2.1 StorageClass Configuration Examples in Typical Manage Volume Provisioning Scenarios** and **6.1.1.2.2 StorageClass Parameters for Manage Volume Provisioning**.

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

```
kubectl apply -f mysc.yaml
```

Step 3 Run the following command to view the information about the created StorageClass.

```
kubectl get sc mysc
```

The following is an example of the command output.

```
NAME   PROVISIONER      RECLAIMPOLICY   VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
mysc   csi.huawei.com   Delete          Immediate           true                   8s
```

----End

Configuring a PVC

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: "*"  # Enter the storage resource name.
    csi.huawei.com/manageBackendName: "*" # Enter the storage backend name.
  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
```

Step 3 After a period of time, run the following command to view the information about the created PVC.

```
kubectl get pvc mypvc
```

The following is an example of the command output. If the PVC status is **Bound**, the PVC has been created and can be used by a Pod.

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

NOTICE

- After the PVC is created, if the PVC is in the **Pending** state after a long time (for example, one minute), refer to **9.3.1 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

Using a PVC

The use method is the same as that for dynamic volume provisioning in **Using a PVC**.

6.1.1.2.1 StorageClass Configuration Examples in Typical Manage Volume Provisioning Scenarios

For details about how to configure a StorageClass in typical Manage Volume Provisioning 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 the Permission on a Mount Directory 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
mountOptions:
  - nfsvers=4.1 # Specify the version 4.1 for NFS mounting.
```

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: "*"
```



```
mountOptions:
  - acl # Set the authentication parameter.
  - cnflush # Set the asynchronous disk flushing mode.
```

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" # Set the directory permission.
```

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
```

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

```
kubectl get sc
```

The following is an example of the command output.

```
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

NOTICE

In the Manage Volume Provisioning mode, pay attention to the following when using a StorageClass:

- 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.

6.1.1.2.2 StorageClass Parameters for Manage Volume Provisioning

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 6-7 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.	Yes	-	• **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.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**.
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.applicationType	Application type name for creating a LUN or NAS when the backend is OceanStor Dorado. **NOTE** If an application type has been configured before a volume is managed, the value of **applicationType** must be the same as the configured application type.	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.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. This parameter is available when **volumeType** is set to **lun**.

Parameter	Description	Mandatory	Default Value	Remarks
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**, **4.1**, or **4.2**.	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, 4.1, and 4.2 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, 4.2. If the 4.0 protocol is required, you are advised to set **nfsver:ws** 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.

Parameter	Description	Mandatory	Default Value	Remarks
mountOptions.sec	Kerberos 5 protocol for mounting NFS file systems.	No	-	• If Kerberos 5 is used, set this parameter to **krb5**. • If Kerberos 5i is used, set this parameter to **krb5i**. • If Kerberos 5p is used, set this parameter to **krb5p**. • Kerberos supports only NFSv4.0 or NFSv4.1.
mountOptions.proto	Transmission protocol used for NFS mounting. The value can be **rdma**.	No	-	• Ensure that NFS over RDMA is enabled on the storage system. • NAS storage of OceanStor Dorado 6.1.7 or later is supported.
mountOptions.port	Protocol port used for NFS mounting.	Conditionally mandatory	-	If the transmission protocol is **rdma**, set this parameter to **20049**.
mountOptions.discard	Automatically triggers the Trim or Discard operation when a file system is mounted. This operation instructs a block device to release unused blocks.	No	-	The xfs and ext4 file systems are supported.

6.1.1.2.3 PVC Parameters for Manage Volume Provisioning

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 6-8 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 **Table 4-7**. • *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.	No	-	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 6-9**. If XFS file systems are used, see **Table 6-10**. 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 **8.6 Enabling the ReadWriteOncePod Feature Gate**. • The support for RWX is as follows: – **NAS storage**: supported by all volumes – **SAN storage**: supported only by volumes whose **volumeMode** is set to **Block**

Table 6-9 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 6-10 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

6.1.1.3 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

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.

Configuring a PV

Step 1 Prepare the PV configuration file **mypv.yaml**. The following is an example. For details about other parameters, see **6.1.1.3.1 PV Parameters for Static Volume Provisioning**.

```
kind: PersistentVolume
apiVersion: v1
metadata:
  name: mypv
spec:
  volumeMode: Filesystem
  storageClassName: "" # The value must be to "".
  accessModes:
    - ReadWriteOnce
```



```
csi:
  driver: csi.huawei.com # Enter the CSI driver name.
  volumeHandle: iscsi-dorado-181.lun0001 # Enter the volume name.
  fsType: xfs # Set the file system type.
capacity:
  storage: 100Gi
```

NOTE

In the configuration file for static volume provisioning, **storageClassName** must be set to **""**. Otherwise, Kubernetes will use the default StorageClass.

Step 2 Run the following command to create a PV based on the prepared .yaml file.

```
kubectl create -f mypv.yaml
```

Step 3 After a period of time, run the following command to view the information about the created PV.

```
kubectl get pv
```

The following is an example of the command output. If the PV status is **Available**, the PV is successfully created.

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

----End

Configuring a PVC

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

Step 1 Prepare the PVC configuration file. The following example is a PVC configuration file for static volume provisioning.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: mypvc
spec:
  storageClassName: ""
  accessModes:
    - ReadWriteOnce
  volumeMode: Filesystem
  resources:
    requests:
      storage: 100Gi
  volumeName: mypv # Enter the name of the corresponding PV.
```

Step 2 Run the following command to create a PVC based on the configured .yaml file.

```
kubectl create -f mypvc.yaml
```

Step 3 After a period of time, run the following command to view the information about the created PVC.

```
kubectl get pvc
```

The following is an example of the command output. If the PVC status is **Bound**, the PVC is successfully created.

```
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 **9.3.1 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

Using a PVC

The use method is the same as that for dynamic volume provisioning in **Using a PVC**.

6.1.1.3.1 PV Parameters for Static Volume Provisioning

Table 6-11 Static volume provisioning parameters

Parameter	Description	Mandatory	Default Value	Remarks
metadata.name	User-defined name of a PV 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 **8.6 Enabling the ReadWriteOncePod Feature Gate**. • The support for RWX is as follows: – **NAS storage**: supported by all volumes – **SAN storage**: supported only by volumes whose **volumeMode** is set to **Block**

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**, **4.1**, or **4.2**.	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, 4.1, and 4.2 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, 4.2. If the 4.0 protocol 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.

6.1.1.3.2 PVC Parameters for Static Volume Provisioning

Table 6-12 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 **8.6 Enabling the ReadWriteOncePod Feature Gate**. • The support for RWX is as follows: – **NAS storage**: supported by all volumes – **SAN storage**: supported only by volumes whose **volumeMode** is set to **Block**

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 6-5**. If XFS file systems are used, see **Table 6-6**. 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.

Parameter	Description	Mandatory	Default Value	Remarks
spec.storageClassName	Name of the StorageClass object.	Yes	-	When a PVC is created, an empty character string is transferred. If this parameter is not set, the default StorageClass object name will be used.

6.1.2 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 3-5** and **Table 3-9**. For details about the Kubernetes versions that support capacity expansion, see **3.2 Kubernetes Feature Matrix**.
- The csi-resizer service is enabled for huawei-csi-controller.

  ```
  kubectl describe deploy huawei-csi-controller -n huawei-csi | grep csi-resizer
  ```

  If the following information is displayed, the csi-resizer service is enabled.

  ```
  csi-resizer:
    Image:      k8s.gcr.io/sig-storage/csi-resizer:v1.4.0
  ```

Procedure

Step 1 Run the following command to check whether the StorageClass supports capacity expansion. In the preceding command, *mysc* indicates the name of the StorageClass to be queried.

```
kubectl get sc mysc
```

The following is an example of the command output.

```
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 3**.

Step 2 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 3 Run the following 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
```


The following is an example of the command output.

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

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 6-5**. If XFS file systems are used, see **Table 6-6**. 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 **9.3.4 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
```

The following is an example of the command output. If the value of **CAPACITY** is changed to the specified capacity, the capacity expansion is successful.

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

----End

6.1.3 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
```


NOTICE

- 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 3-5** and **Table 3-9**. For details about the Kubernetes versions that support cloning, see **3.2 Kubernetes Feature Matrix**.

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
```

----End

6.1.4 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
```

NOTICE

- 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 3-5** and **Table 3-9**. For details about the Kubernetes versions that support PVC creation using a snapshot, see **3.2 Kubernetes Feature Matrix**.

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
```

----End

6.2 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

6.2.1 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 **4.1.4 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.

6.2.2 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 6-13**. 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 6-13 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.
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 **4.1.4 Checking Volume Snapshot-Dependent Components**. For details about the Kubernetes versions that support VolumeSnapshot creation, see **Table 3-3**.

Procedure

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

```
kubectl create -f mysnapclass.yaml
```

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

```
kubectl get volumesnapshotclass
```

The following is an example of the command output.

```
NAME          DRIVER           DELETIONPOLICY   AGE
mysnapclass   csi.huawei.com   Delete           25s
```

----End

6.2.3 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 6-14**.

Table 6-14 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 3-5** and **Table 3-9**. For details about the Kubernetes versions that support VolumeSnapshot creation, see **Table 3-3**.
- The volume snapshot component CRD on which the running of Huawei CSI depends has been installed. For details, see **4.1.4 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
```

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

```
kubectl get volumesnapshot
```

The following is an example of the command output.

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

----End

7 Advanced Features

7.1 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.

7.1.1 Configuring ALUA Using Helm

7.1.1.1 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 V5 and OceanStor Dorado V3 Series

Table 7-1 lists the ALUA parameters supported by Huawei CSI for OceanStor V5 and OceanStor Dorado V3 series.

Table 7-1 ALUA parameters supported by Huawei CSI for OceanStor 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 and OceanStor Dorado Series

Table 7-2 lists the ALUA parameters supported by Huawei CSI for OceanStor and OceanStor Dorado series.

NOTE

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

Table 7-2 ALUA parameters for OceanStor and OceanStor Dorado 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 as an example to describe how to connect to Red Hat. For details about the host connectivity guide, see ***OceanStor Dorado and OceanStor Host Connectivity Guide for Red Hat***.

The following ALUA configuration example is recommended in the OceanStor Dorado 18000 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.

7.1.1.2 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 advised not to configure ALUA parameters for distributed storage.

Table 7-3 lists the ALUA parameters supported by Huawei CSI for distributed storage.

Table 7-3 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.

7.2 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 following command to view information about worker nodes in the current cluster.

   ```
   kubectl get node
   ```

   The following is an example of the command output.

   ```
   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 following 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 7-4**.

```
kubectl label node <nodename> <key>=<value>
```

Table 7-4 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 at **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 configure a StorageClass, the StorageClass 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 configure a StorageClass, the StorageClass must meet all filter criteria.

4. Run the following 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"
```

The following is an example of the command output.

```
[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"**}]
```

7.2.1 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 4-1**.

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
```

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

```
oceanctl create backend -f ../examples/backend/backend.yaml -i yaml
```

Enter the storage user name and password as prompted.

```
Please enter this backend user name:admin
Please enter this backend password:
```

Step 7 Run the **vi StorageClass.yaml** command to modify the .yaml file. Press **I** or **Insert** to enter the insert mode and add related parameters in the .yaml file. For details about the parameters, see **Table 7-5**. 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 7-5 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 at **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 **6.1.1.1.3 PVC Parameters for Dynamic Volume Provisioning**.

----End

7.3 PVC Change

This section describes how to use Huawei CSI to complete a PVC change.

7.3.1 Enabling the PVC Change Feature

The PVC change feature is disabled by default during Huawei CSI installation. To use this feature, perform the following steps.

7.3.1.1 Enabling the PVC Change Feature Using Helm

Prerequisites

- You have installed Huawei CSI using Helm.
- Huawei CSI v4.5.0 or later is used.

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 check whether the PVC change feature is enabled.

helm-huawei-csi indicates the Helm chart name specified during installation, and **huawei-csi** indicates the Helm chart namespace specified during installation. For details about the component package path, see **Table 4-1**.

```
helm get values helm-huawei-csi -n huawei-csi -a | grep volumeModify -A 1
```

The following is an example of the command output.

- If **enabled: true** is displayed in the command output, the feature is enabled. In this case, skip the following steps.
- If **enabled: false** is displayed in the command output, perform the following steps to enable the PVC change feature.

```
volumeModify:
  enabled: false
```

Step 3 Go to the **/helm/esdk** directory and run the following command to configure the volume change CRD.

```
# kubectl apply -f ./crds/volume-modify/
customresourcedefinition.apiextensions.k8s.io/volumemodifyclaims.xuanwu.huawei.io configured
customresourcedefinition.apiextensions.k8s.io/volumemodifycontents.xuanwu.huawei.io configured
```

NOTE

If the command output contains **Warning: resource customresourcedefinitions/volumemodifycontents.xuanwu.huawei.io is missing the kubectl.kubernetes.io/last-applied-configuration...**, you can ignore it. This message is displayed because the **kubectl create** command instead of the **kubectl apply** command is used for installation by Helm.

Step 4 Run the following command to obtain the original service configuration file.

```
helm get values helm-huawei-csi -n huawei-csi -a > ./update-values.yaml
```


Step 5 Run the **vi update-values.yaml** command to open the file obtained in **Step 4** and modify the following configuration. After the modification is complete, press **Esc** and enter **:wq!** to save the modification.

```
csiExtender:
  volumeModify:
    enabled: true
```

Step 6 Run the following command to update Huawei CSI services.

```
helm upgrade helm-huawei-csi ./ -n huawei-csi  -f ./update-values.yaml
```

Step 7 Run the following command to check whether the services are started.

```
kubectl get pod -n huawei-csi
```

The following is an example of the command output. In the preceding command, **huawei-csi** indicates the namespace for deploying Huawei CSI.

```
NAME                                  READY   STATUS    RESTARTS   AGE
huawei-csi-controller-6dfcc4b79f-9vjtq   10/10   Running   0          24m
huawei-csi-node-tqs87                    3/3     Running   0          20m
```

----End

7.3.1.2 Enabling the PVC Change Feature Manually

Prerequisites

Huawei CSI has been manually installed.

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 **manual/esdk** working directory and run the following command to configure the volume change CRD.

```
kubectl apply -f ./crds/volume-modify/
```

Step 3 Run the following command. For details about the component package path, see **Table 4-1**.

```
kubectl apply -f ./deploy/huawei-csi-controller-extender.yaml
```

Step 4 Run the following command to check whether the services are started.

```
kubectl get pod -n huawei-csi
```

The following is an example of the command output. In the preceding command, **huawei-csi** indicates the namespace for deploying Huawei CSI.

```
NAME                                  READY   STATUS    RESTARTS   AGE
huawei-csi-controller-6dfcc4b79f-9vjtq   10/10   Running   0          24m
huawei-csi-node-tqs87                    3/3     Running   0          24m
```

----End

7.3.2 Configuring PVC Changes

The PVC change feature is implemented using CRD. Related resources are described as follows.

Table 7-6 Resource description

NAME	APIVERSION	NAMESPACED	KIND
volumemodifyclaims	xuanwu.huawei.io/v1	false	VolumeModifyClaim
volumemodifycontents	xuanwu.huawei.io/v1	false	VolumeModifyContent

NOTE

- VolumeModifyClaim resources can be created, deleted, and queried, but cannot be updated.
- VolumeModifyContent resources can only be queried and are used to display the change details of a single PVC. Do not manually create, delete, or modify the resources.
- VolumeModifyContent resources are managed by VolumeModifyClaim. Do not manually manage VolumeModifyContent resources.

7.3.2.1 Creating a PVC Change

Prerequisites

The storage backends associated with the PVC to be changed are HyperMetro storage backends. If they are not HyperMetro storage backends, configure them by following the instructions in **5.1.3.2 Manually Updating a Storage Backend**.

7.3.2.1.1 Preparing a PVC Change File

PVC Change File Description

The sample template of the PVC change file is **/examples/volumemodifyclaim.yaml**. The following table lists the configuration items.

Table 7-7 Parameter description

Parameter	Description	Mandatory	Default Value	Remarks
apiVersion	API group, which is of the string type.	Yes	xuanwu.huawei.io/v1	The value is fixed at **xuanwu.huawei.io/v1**.
kind	Resource type, which is of the string type.	Yes	VolumeModifyClaim	The value is fixed at **VolumeModifyClaim**.

Parameter	Description	Mandatory	Default Value	Remarks
metadata.name	Name of a cluster resource object, which is of the string type.	Yes	-	The name must comply with the naming rules of a **DNS subdomain name**. The value can contain a maximum of 63 characters, including digits, lowercase letters, hyphens (-), and periods (.). It must start and end with a lowercase letter or digit.
spec.source.kind	Data source type, which is of the string type.	Yes	StorageClass	This parameter can only be set to **StorageClass**.
spec.source.name	Data source name, which is of the string type.	Yes	-	Only a StorageClass name can be configured.
spec.parameters.hyperMetro	Whether to change a common volume to a HyperMetro volume. Currently, the value can only be **"true"**.	Yes	-	Only common storage volumes at the primary site can be changed to HyperMetro storage volumes.

Parameter	Description	Mandatory	Default Value	Remarks
spec.parameters.metroPairSyncSpeed	Data synchronization speed of a HyperMetro pair. The value ranges from 1 to 4. The value can be: • **1**: low • **2**: medium • **3**: high • **4**: highest	No	-	This parameter is available only when **spec.parameters.hyperMetro** is set to **"true"**. Note: • If this parameter is not configured, the storage speed of the HyperMetro pair is determined by the storage device. • The highest synchronization speed may increase the host latency.

NOTE

- The **spec.source.kind** and **spec.source.name** parameters are used to specify the volume change scope. For example, if they are set to a StorageClass and the corresponding name respectively, all PVCs in the **Bound** state provisioned using the target StorageClass will be changed.
- After all associated PVCs are changed, Huawei CSI will replace the original StorageClass and add the **spec.parameters** parameter of the VolumeModifyClaim so that the PVCs meet the StorageClass definition.

For details about the configuration in typical scenarios, see the following example:

Changing a Common Volume to a HyperMetro Volume

The following is an example of changing a common volume to a HyperMetro volume:

```
apiVersion: xuanwu.huawei.io/v1
kind: VolumeModifyClaim
metadata:
  name: myvmc
spec:
  source:
    kind: StorageClass
    name: mysc
  parameters:
    hyperMetro: "true"
```

7.3.2.1.2 Creating a PVC Change Resource

This section describes how to create a PVC change resource based on a configured PVC change file.

NOTE

- Only the HyperMetro active-active (AA) mode is supported.
- When a common volume is changed to a HyperMetro volume, only the storage volume at the primary site can be changed.
- Do not use Huawei CSI to manage a PVC during PVC change resource creation.
- Multiple VolumeModifyClaim resources cannot be created for the same PVC. If the target PVC needs to be changed for multiple times, perform the changes one by one.

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 create a PVC change.

```
kubectl create -f volumemodifyclaim.yaml
```

Step 3 Query the creation result by following the instructions in **7.3.2.2 Querying a PVC Change**.

----End

7.3.2.2 Querying a PVC Change

This section describes how to use Kubectl to query the PVC change status. Currently, Huawei CSI provides the following APIs through CRD.

Querying a VolumeModifyClaim

To query a VolumeModifyClaim using kubectl, perform the following steps.

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 a PVC change. In the command, *vmc-name* indicates the name of the VolumeModifyClaim resource.

```
kubectl get volumemodifyclaims <vmc-name> -owide
```

The following is an example of the command output.

```
NAME    STATUS      READY   SOURCEKIND     SOURCENAME   STARTEDAT             COMPLETEDAT           AGE
myvmc   Completed   1/1     StorageClass   mysc         2024-06-06T03:19:13Z  2024-06-06T03:19:16Z  2m2s
```

Table 7-8 Command output description

Parameter	Description
NAME	VolumeModifyClaim resource name.

Parameter	Description
STATUS	VolumeModifyClaim resource status. The value can be: • **Pending**: initial status. • **Creating**: The VolumeModifyClaim has completed basic verification and the server has received the change task, but the task has not been completed. • **Completed**: All associated PVCs are changed. • **Rollback**: When associated PVCs are partially changed, a user deletes PVCs. • **Deleting**: When all associated PVCs are changed, a user deletes PVCs.
READY	Ratio of the number of changed PVCs to the total number of PVCs that need to be changed.
SOURCEKIND	Data source type, for example, StorageClass.
SOURCENAME	Data source name, for example, StorageClass name.
STARTEDAT	Change start time, that is, the timestamp when the server receives the task and starts to process the task.
COMPLETEDAT	Change completion time, that is, the timestamp when the changes of all associated PVCs are complete. This parameter exists only when **STATUS** is **Completed**.
AGE	Lifetime of a VolumeModifyClaim from the time when it is created to the current time.

----End

NOTE

You can use kubectl to view the **Events** information of a VolumeModifyClaim. If a VolumeModifyClaim cannot meet the creation requirements or an error occurs during the creation, the server will record the **Events** information. The following command is used as an example:

```
kubectl describe volumemodifyclaims local-to-hypermetro
```

Querying a VolumeModifyContent

A VolumeModifyContent is created using a VolumeModifyClaim and records the change details of a single PVC. To query a VolumeModifyContent using kubectl, perform the following steps:

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 a PVC change. In the command, *myvmc-uid* indicates the VolumeModifyContent resource name.

```
kubectl get volumemodifycontents myvmc-uid -owide
```

The following is an example of the command output.


```
NAME        STATUS      MODIFYCLAIMNAME     SOURCEVOLUME    STARTEDAT
COMPLETEDAT           AGE
myvmc-uid   Completed   myvmc               default/mypvc   2024-06-06T03:19:07Z   2024-06-06T03:19:09Z
36m
```

Table 7-9 Command output description

Parameter	Description
NAME	VolumeModifyContent resource name. The format is *VolumeModifyClaim name*-*UID of the associated PVC*.
STATUS	VolumeModifyContent resource status. The value can be: ● **Pending**: initial status. ● **Creating**: The VolumeModifyContent has completed basic verification and the server has received the change task, but the task has not been completed. ● **Completed**: The associated PVC is changed. ● **Rollback**: The PVC change is being rolled back.
MODIFYCLAIMNAME	Name of the associated VolumeModifyClaim.
SOURCEVOLUME	Information about the associated PVC. The format is *Namespace name*/*PVC name*.
STARTEDAT	PVC change start time, that is, the timestamp when the server receives the task and starts to process the task.
COMPLETEDAT	PVC change completion time, that is, the timestamp when the changes of all associated PVCs are complete. This parameter exists only when **STATUS** is **Completed**.
AGE	Lifetime of a VolumeModifyContent from the time when it is created to the current time.

----End

NOTE

You can use kubectl to view the **Events** information of a VolumeModifyContent. If a VolumeModifyContent cannot meet the creation requirements or an error occurs during the PVC change, the server will record the **Events** information. The following command is used as an example:

```
kubectl describe volumemodifycontents myvmc-uid
```


7.3.2.3 Deleting a PVC Change

NOTICE

- If **STATUS** of a VolumeModifyClaim is **Creating**, deleting the VolumeModifyClaim resource will delete the created resource on the storage side and then remove the cluster resource. After the deletion, if you continue to use the original StorageClass for PVC management, you need to restore the associated storage backend to a non-HyperMetro storage backend.
- If **STATUS** of a VolumeModifyClaim is **Pending** or **Completed**, deleting the VolumeModifyClaim resource will only remove the cluster resource and will not delete the created resource on the storage side (that is, there is not interaction with the storage side).
- VolumeModifyContent resources are managed by VolumeModifyClaim. Do not manually manage VolumeModifyContent resources.
- If some PVCs among the PVCs to be changed meet the change requirements and the batch change fails, all PVC changes will be removed. As a result, the PVCs that meet the change requirements will not meet the change requirements.
- If a PVC to be changed has been manually managed on the storage side, the change may fail. Do not manually manage storage volumes when using the change feature.

This section describes how to use kubectl to delete a PVC change. The procedure is as follows.

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 delete a PVC change. In the command, *vmc-name* indicates the name of the VolumeModifyClaim resource.

```
kubectl delete volumemodifyclaims <vmc-name>
```

Step 3 Query the deletion result by following the instructions in **7.3.2.1.2 Creating a PVC Change Resource**.

----End

8 Common Operations

8.1 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
```

If the following information is displayed, the installation is successful.

```
version.BuildInfo{Version:"v3.9.0", GitCommit:"7ceeda6c585217a19a1131663d8cd1f7d641b2a7",
GitTreeState:"clean", GoVersion:"go1.17.5"}
```

----End

8.2 Collecting Information

8.2.1 Obtaining the CSI Version

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
```

The following is an example of the command output.

```
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
```

The version information is displayed as follows:

```
4.5.0
```

----End

8.2.2 Viewing 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
```

The following is an example of the command output, where **IP** indicates the node IP address and **NODE** indicates the node name.


```
NAME                                  READY   STATUS    RESTARTS   AGE     IP           NODE          NOMINATED
NODE   READINESS GATES
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 node where the huawei-csi-controller service resides in the Kubernetes cluster through the management IP address.

Step 3 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 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-<container-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
```

The following is an example of the command output, where **IP** indicates the node IP address and **NODE** indicates the node name.

```
NAME                  READY   STATUS    RESTARTS   AGE     IP           NODE          NOMINATED NODE
READINESS GATES
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 node where the huawei-csi-node service resides in the Kubernetes cluster through the management IP address.

Step 3 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 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-<container-id>.log
```

----End

8.2.3 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 following command. The displayed version is **v4.5.0**.

```
oceanctl version
```

The following is an example of the command output.

```
Oceanctl Version: v4.5.0
```

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

```
$ oceanctl collect logs --help
Collect logs of one or more nodes in specified namespace in Kubernetes

Usage:
  oceanctl collect logs [flags]

Examples:
  # Collect logs of all nodes in specified namespace
  oceanctl collect logs -n <namespace>

  # Collect logs of specified node in specified namespace
  oceanctl collect logs -n <namespace> -N <node>

  # Collect logs of all nodes in specified namespace
  oceanctl collect logs -n <namespace> -a

  # Collect logs of all nodes in specified namespace with a maximum of 50 nodes collected at the same time
  oceanctl collect logs -n <namespace> -a --threads-max=50

  # Collect logs of specified node in specified namespace
  oceanctl collect logs -n <namespace> -N <node> -a

Flags:
  -a, --all              Collect all nodes messages
  -h, --help             help for logs
  -n, --namespace string   namespace of resources
  -N, --nodename string    Specify the node for which information is to be collected.
      --threads-max int    set maximum number[1~1000] of threads for nodes to be collected. (default 50)

Global Flags:
      --log-dir string   Specify the directory for printing log files. (default "/var/log/huawei")
```

Step 4 Run the following command to check whether a Pod is started properly. In the command, *huawei-csi* indicates the namespace for installing CSI.

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

The following is an example of the command output.

```
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 --threads-max=***<max_node_processing_num>* command to collect CSI logs of all nodes where CSI containers reside in the cluster. In the command, **threads-max** indicates the maximum number of nodes for which logs can be collected at the same time. The default value is **50**. You can set the value based on the host performance and load.

```
oceanctl collect logs -n huawei-csi -a --threads-max=10
```


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
```

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

8.3 Downloading a Container Image

Downloading a Container Image Using containerd

Step 1 Run the following command to download an image to a local path. In the command, *image:tag* indicates the image to be pulled and its tag.

```
ctr image pull <image>:<tag>
```

Step 2 Run the following command to export the image to a file. In the command, *image:tag* indicates the image to be exported, and *file* indicates the name of the exported image file.

```
ctr image export <file>.tar <image>:<tag>
```

----End

Downloading a Container Image Using Docker

Step 1 Run the following command to download an image to a local path. In the command, *image:tag* indicates the image to be pulled.

```
docker pull <image>:<tag>
```

Step 2 Run the following command to export the image to a file. In the command, *image:tag* indicates the image to be exported, and *file* indicates the name of the exported image file.


```
docker save <image>:<tag> -o <file>.tar
```

----End

Downloading a Container Image Using Podman

Step 1 Run the following command to download an image to a local path. In the command, *image:tag* indicates the image to be pulled.

```
podman pull <image>:<tag>
```

Step 2 Run the following command to export the image to a file. In the command, *image:tag* indicates the image to be exported, and *file* indicates the name of the exported image file.

```
podman save <image>:<tag> -o <file>.tar
```

----End

8.4 Updating the huawei-csi-controller or huawei-csi-node Service

Perform this operation when you need to update the huawei-csi-controller or huawei-csi-node service, for example, changing the number of copies for 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 Go to the **/helm/esdk** directory and run the following command to obtain the original service configuration file. **helm-huawei-csi** indicates the Helm chart name specified during the installation of the earlier version, and **huawei-csi** indicates the Helm chart namespace specified during the installation of the earlier version. For details about the component package path, see **Table 4-1**.

```
helm get values helm-huawei-csi -n huawei-csi -a > ./update-values.yaml
```

Step 3 Run the **vi update-values.yaml** command to open the file obtained in **Step 2** and modify the configuration items by referring to **4.2.1.3 Parameters in the values.yaml File of Helm**. After the modification, press **Esc** and enter **:wq!** to save the modification.

Step 4 Run the following command to update Huawei CSI services.

```
helm upgrade helm-huawei-csi ./ -n huawei-csi  -f ./update-values.yaml
```

----End

8.5 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 **/helm/esdk** directory and run the following command to obtain the original service configuration file. **helm-huawei-csi** indicates the Helm chart name specified during the installation of the earlier version, and **huawei-csi** indicates the Helm chart namespace specified during the installation of the earlier version. For details about the component package path, see **Table 4-1**.

```
helm get values helm-huawei-csi -n huawei-csi -a > ./update-values.yaml
```

Step 3 Run the **vi update-values.yaml** command to open the file obtained in **Step 2** and modify the configuration items. After the modification, 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
  # Timeout interval for running command on the host. support 1~600
  execCommandTimeout: 30
  # check the number of paths for multipath aggregation
  # Allowed values:
  #   true: the number of paths aggregated by DM-multipath is equal to the number of online paths
  #   false: the number of paths aggregated by DM-multipath is not checked.
  # Default value: false
  allPathOnline: false
  # 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 4 Run the following command to update the log configuration.

```
helm upgrade helm-huawei-csi ./ -n huawei-csi  -f ./update-values.yaml
```

----End

8.6 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 or 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 insert 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 insert 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

8.7 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

9 Troubleshooting

9.1 Huawei CSI Service Issues

9.1.1 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 4-1**.

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 following 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 -f values.yaml
```

The following is an example of the command output.

```
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

9.1.2 Huawei CSI Services Fail to Be Started and Error Message "/etc/localtime is not a file" Is Displayed

Symptom

During the installation and deployment of CSI, a Pod fails to run and is in the **ContainerCreating** state. Alarm **/etc/localtime is not a file** is generated for the Pod.

Root Cause Analysis

When the container mounts the **/etc/localtime** file on the host, the type is incorrectly identified. As a result, the container fails to mount the **/etc/localtime** file on the host and the Pod cannot run.

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 check the running status of the Pod of the CSI services.

```
kubectl get pod -n huawei-csi
```

The following is an example of the command output. *huawei-csi* indicates the namespace where the CSI services are deployed.

```
NAME                                  READY   STATUS              RESTARTS   AGE
huawei-csi-controller-6dfcc4b79f-9vjtq   9/9     ContainerCreating   0          24m
huawei-csi-controller-6dfcc4b79f-csphc   9/9     ContainerCreating   0          24m
huawei-csi-node-g6f4k                    3/3     ContainerCreating   0          20m
huawei-csi-node-tqs87                    3/3     ContainerCreating   0          20m
```


Step 3 Run the following command to check the **Events** parameter of the container.

```
kubectl describe pod huawei-csi-controller-6dfcc4b79f-9vjtq -n huawei-csi
```

The following is an example of the command output. In the command, *huawei-csi-controller-6dfcc4b79f-9vjtq* indicates the name of the Pod in the **ContainerCreating** state found in **Step 2**, and *huawei-csi* indicates the namespace to which the Pod belongs.

```
...
Events:
  Type     Reason       Age              From               Message
  ----     ------       ----             ----               -------
  Normal   Scheduled    96s              default-scheduler  Successfully assigned huawei-csi/huawei-csi-controller-6dfcc4b79f-9vjtq to node1
  Warning  FailedMount  33s (x8 over 96s)  kubelet          MountVolume.SetUp failed for volume "host-time" : hostPath type check failed: /etc/localtime is not a file
```

Step 4 Run the **cd /helm/esdk/templates** command to go to the CSI installation package path. For the path, see **Table 4-1**.

Step 5 Take the **huawei-csi-controller.yaml** file as an example. Run the following command to view the file content.

```
vi huawei-csi-controller.yaml
```

Find the **host-time** configuration item under **volumes**, and delete the **type: File** line. Perform the same operations on the **huawei-csi-node.yaml** deployment file that involves the configuration item in the **templates** directory.

```
...
...
volumes:
  - hostPath:
      path: /var/log/
      type: Directory
    name: log
  - hostPath:
      path: /etc/localtime
      type: File
    name: host-time
...
...
```

Step 6 Uninstall and reinstall the service by referring to **4.3.1 Uninstalling Huawei CSI Using Helm**.

Step 7 Run the following command to check whether the Pod running status of Huawei CSI services is **Running**.

```
kubectl get pod -n huawei-csi
```

The following is an example of the command output.

```
NAME                                  READY   STATUS    RESTARTS   AGE
huawei-csi-controller-6dfcc4b79f-9vjts   9/9     Running   0          24m
huawei-csi-controller-6dfcc4b79f-csphb   9/9     Running   0          24m
huawei-csi-node-g6f41                 3/3     Running   0          20m
huawei-csi-node-tqs85                 3/3     Running   0          20m
```

----End

9.1.3 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
```

The following is an example of the command output.

```
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:4.5.0
...
```

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 insert 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:4.5.0**.

  ```
  containers:
    ...
    - name: huawei-csi-driver
      image: huawei-csi:4.5.0
  ```

Step 3 Run the following command to modify the configuration file of the huawei-csi-controller service: Press **I** or **Insert** to enter the insert 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:4.5.0**.

  ```
  containers:
    ...
    - name: huawei-csi-driver
      image: huawei-csi:4.5.0
  ```


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
```

The following is an example of the command output. If the Pod status is **Running**, the services are started successfully.

```
huawei-csi   huawei-csi-controller-58799449cf-zvhmv   9/9     Running     0          2m29s
huawei-csi   huawei-csi-node-7fxh6                    3/3     Running     0          12m
```

----End

9.2 Storage Backend Issues

9.2.1 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@ubuntu-master:/opt/huawei-csi/backend# oceanctl delete backend onas
Error: secret "onas" deleted
configmap "onas" deleted
Error from server (InternalError): Internal error occurred: failed calling webhook "storage-backend-controller.xuanwu.huawei.i
ler.huawei-csi.svc:443/storagebackendclaim?timeout=10s": no service port 443 found for service "huawei-csi-controller"
```

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
```

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
kubectl scale deployment huawei-csi-controller -n huawei-csi --replicas=1
```

Step 3 Run the following command to check whether CSI Controller is successfully started.

```
kubectl get pod -n huawei-csi
```

The following is an example of the command output. If the Pod status is **Running**, Controller is successfully started.


```
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

9.2.2 A Backend Fails to Be Created Using the oceanctl Tool and Error Message "context deadline exceeded" Is Displayed

Symptom

A user fails to create a storage backend using the oceanctl tool, and "failed to call webhook: xxx :context deadline exceeded; error: exist status 1" is displayed on the console.

Root Cause Analysis

When a storage backend is created, the webhook service provided by CSI is invoked to verify the connectivity with the storage management network and the storage account and password. The possible causes are as follows:

- Huawei CSI fails to verify the connectivity of the storage management network.
- The communication between kube-apiserver and CSI webhook is abnormal.

Huawei CSI Fails to Verify the Connectivity of the Storage Management Network

Perform the following steps to check whether Huawei CSI fails to verify the connectivity of the storage management network.

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 obtain CSI service information. *huawei-csi* indicates the namespace where the CSI services are deployed.

```
kubectl get pod -n huawei-csi -owide
```

The following is an example of the command output.

```
NAME                       READY   STATUS    RESTARTS   AGE   IP         NODE      NOMINATED NODE
READINESS GATES
huawei-csi-controller-xxx   9/9     Running   0          19h   host-ip1   host-1    <none>           <none>
huawei-csi-node-mnqbz       3/3     Running   0          19h   host-ip1   host-1    <none>           <none>
```

Step 3 Log in to the node where huawei-csi-controller resides, for example, **host-1** in **Step 2**.

Step 4 Go to the **/var/log/huawei** directory.

```
# cd /var/log/huawei
```

Step 5 View the **storage-backend-controller** log. The following uses the storage connection timeout as an example.

```
tail -n 1000 storage-backend-controller
```

The following is a log example.


```
2024-01-01 06:30:44.280661 1 [INFO]: Try to login https://192.168.129.155:8088/deviceManager/rest
2024-01-01 06:31:44.281626 1 [ERROR]: Send request method: POST, Url: https://192.168.129.155:8088/
deviceManager/rest/xx/sessions, error: Post "https://192.168.129.155:8088/deviceManager/rest/xx/sessions":
context deadline exceeded (Client.Timeout exceeded while awaiting headers)
2024-01-01 06:31:44.281793 1 [WARNING]: Login https://192.168.129.155:8088/deviceManager/rest error
due to connection failure, gonna try another Url
2024-01-01 06:31:44.291668 1 [INFO]: Finished validateCreate huawei-csi/backend-test.
2024-01-01 06:31:44.291799 1 [ERROR]: Failed to validate StorageBackendClaim, error: unconnected
```

Step 6 If the log contains information about login timeout, login failure, or long request duration, check the connectivity between the host machine and the storage or the network status.

Step 7 If no request is recorded in the log, the communication between kube-apiserver and CSI webhook is abnormal.

----End

Abnormal Communication Between kube-apiserver and CSI Webhook

Contact the Kubernetes platform administrator to check the network between kube-apiserver and CSI webhook. For example, if kube-apiserver has an HTTPS proxy, the CSI webhook service may fail to be accessed.

NOTE

In the temporary workaround, the webhook resource will be deleted. This resource is used to check whether the entered account information is correct and whether the connection to the storage can be set up when a storage backend is created. Therefore, deleting this resource affects only the verification during backend creation and does not affect other functions. Pay attention to the following:

- Ensure that the host machine where the huawei-csi-controller service is located can properly communicate with the storage.
- Ensure that the entered account and password are correct.

Step 1 Run the following command to view CSI webhook information.

```
kubectl get validatingwebhookconfiguration storage-backend-controller.xuanwu.huawei.io
```

The following is an example of the command output.

```
NAME                                      WEBHOOKS   AGE
storage-backend-controller.xuanwu.huawei.io   1          4d22h
```

Step 2 Contact the Kubernetes platform administrator to check whether the communication between kube-apiserver and CSI webhook is abnormal.

Step 3 Perform the following temporary workaround: Run the following command to delete the webhook.

```
kubectl delete validatingwebhookconfiguration storage-backend-controller.xuanwu.huawei.io
```

Step 4 Create a storage backend. For details, see **5.1 Managing Storage Backends**.

Step 5 If the communication between kube-apiserver and CSI webhook is restored, you need to reconstruct the webhook. In this case, run the following command to restart CSI Controller and restore the number of CSI Controller copies by specifying **--replicas=***. In the following example, the number is restored to **1**. Change it based on actual requirements.

Change the number of copies to 0 first.

```
kubectl scale deployment huawei-csi-controller -n huawei-csi --replicas=0
```


Then restore the number of copies to the original number.

```
kubectl scale deployment huawei-csi-controller -n huawei-csi --replicas=1
```

----End

9.2.3 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 OceanStor Dorado 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 OceanStor Dorado 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 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 **5.1.3 Updating a Storage Backend**.

Step 5 Log in to the storage device as user admin and modify the login policy. Take OceanStor Dorado as an example. On DeviceManager, choose **Settings** > **User and Security** > **Security Policies** > **Login Policy**, click **Modify**, and enable **Account Lockout**.

----End

9.3 PVC Issues

9.3.1 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 **6.1.1.1.1 StorageClass Configuration Examples in Typical Dynamic Volume Provisioning Scenarios**.

  c. Create a PVC. For details, see **6.1.1.1.3 PVC Parameters for Dynamic Volume Provisioning**.

- 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 **6.1.1.1.1 StorageClass Configuration Examples in Typical Dynamic Volume Provisioning Scenarios**.

e. Create a PVC. For details, see **6.1.1.1.3 PVC Parameters for Dynamic Volume Provisioning**.

- 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

9.3.2 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

9.3.3 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.

9.3.4 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**.

9.4 Pod Issues

9.4.1 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 following command to check whether a DM multipathing device with abnormal multipathing status exists.

   ```
   multipath -ll
   ```

   The following is an example of the command output. 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.

   ```
   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. Run the following command to check whether the residual DM multipathing device is readable.
   ```
   dd if=/dev/dm-12 of=/dev/null count=1 bs=1M iflag=direct
   ```
   The following is an example of the command output. If the returned result is **Input/output error** and the read data is **0 bytes (0 B) copied**, the device is unreadable. *dm-xx* indicates the device ID obtained in **step 1.1**.
   ```
   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
      ```
      The following is an example of the command output.
      ```
      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 following 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 following commands in sequence to query the multipathing and disk information. If the residual **dm-12** device and SCSI disks **sdi** and **sdj** are cleared, the clearing is complete.
   a. View multipathing information.
      ```
      multipath -ll
      ```
      The following is an example of the command output. The residual **dm-12** device is cleared.
      ```
      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
```

b. View device information.

```
ls -l /sys/block/
```

The following is an example of the command output. SCSI disks **sdi** and **sdj** are cleared.

```
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
```

c. View disk information.

```
ls -l /dev/disk/by-id/
```

The following is an example of the command output. SCSI disks **sdi** and **sdj** are cleared.

```
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

9.4.2 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 **8.2.2 Viewing 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
```

The following is an example of the command output.

```
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

9.4.3 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 **8.2.2 Viewing 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.

9.4.4 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 **8.2.2 Viewing 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 and error message "error: exit status 255" is displayed. The possible cause is that the value of **LimitNOFILE** of container runtime containerd is too large (over 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.
- 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 **8.2.2 Viewing 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

9.4.5 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 **8.2.2 Viewing Huawei CSI Logs**). The log shows that the mount command fails to be executed.

Root Cause Analysis

The possible cause is that the NFS 4.0/4.1/4.2 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 3/4.0/4.1/4.2 protocol on the storage side and retry the default mounting.
- Specify an available NFS protocol for mounting. For details, see **6.1.1.1.1 StorageClass Configuration Examples in Typical Dynamic Volume Provisioning Scenarios**.

9.4.6 A Pod Fails to Be Created and Message "publishInfo doesn't exist" Is Displayed in the Events Log

Symptom

When a Pod is being created, the Pod keeps in the **ContainerCreating** state. It is found that the following alarm event is printed for the Pod: **rpc error: code = Internal desc = publishInfo doesn't exist**

Root Cause Analysis

As required by CSI, when a workload needs to use a PV, the Container Orchestration system (CO system, communicating with the CSI plug-in using RPC requests) invokes the ControllerPublishVolume interface (provided by huawei-csi-controller) in the **CSI protocol** provided by the CSI plug-in to map the PV, and then invokes the NodeStageVolume interface (provided by huawei-csi-node) provided by the CSI plug-in to mount the PV. During a complete mounting operation, only the huawei-csi-node service receives the NodeStageVolume

request. Before that, the huawei-csi-controller service does not receive the ControllerPublishVolume request. As a result, the huawei-csi-controller service does not map the PV volume and does not send the mapping information to the huawei-csi-node service. Therefore, error message **publishInfo doesn't exist** is reported.

Solution

To solve this problem, Kubernetes needs to invoke the ControllerPublishVolume interface.

If this operation is triggered by all workloads created by earlier versions in the cluster, this problem will not occur.

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 obtain the information about the node where a workload is located.

```
kubectl get pod error-pod -n error-pod-in-namespace -owide
```

The following is an example of the command output.

```
NAME     READY   STATUS              RESTARTS   AGE   IP       NODE     NOMINATED NODE   READINESS GATES
pod-nfs  0/1     ContainerCreating   0          3s    <none>   node-1   <none>           <none>
```

Step 3 Fail over the workload to another node.

Step 4 If the failover cannot be completed in the cluster, you can delete the workload and create a new one on the original node.

Step 5 Check whether the workload is successfully started. If it fails to be started, contact Huawei technical support engineers.

----End

Checking Cluster Workloads

When Kubernetes invokes the CSI plug-in to complete volume mapping, the VolumeAttachment resource is used to save the mapping information, indicating that a specified volume is attached to or detached from a specified node. This problem occurs because publishInfo does not exist. You can view the VolumeAttachment resource information to check whether this problem is also involved in other workloads in the cluster. The procedure is as follows:

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 obtain the VolumeAttachment information and retain resources whose **ATTACHER** field is **csi.huawei.com**. **csi.huawei.com** indicates the Huawei CSI driver name and can be configured in the **values.yaml** file. The corresponding configuration item is **csiDriver.driverName**. For details about the configuration item, see **Table 4-7**.

```
kubectl get volumeattachments.storage.k8s.io
```


The following is an example of the command output.

```
NAME          ATTACHER         PV       NODE     ATTACHED   AGE
csi-47abxx    csi.huawei.com   pvc-1xx  node-1   true       12h
```

Step 3 Run the following command to view the VolumeAttachment resource details. In the following information, **csi-47abxx** is the resource name obtained in **Step 2**.

```
kubectl get volumeattachments.storage.k8s.io csi-47abxx -o yaml
```

The following is an example of the command output.

```
kind: VolumeAttachment
metadata:
  annotations:
    csi.alpha.kubernetes.io/node-id: '{"HostName":"node-1"}'
   finalizers:
  - external-attacher/csi-huawei-com
  name: csi-47abxxx
  uid: 0c87fa8a-c3d6-4623-acb8-71d6206d030d
spec:
  attacher: csi.huawei.com
  nodeName: debian-node
  source:
    persistentVolumeName: pvc-1xx
status:
  attached: true
  attachmentMetadata:
     publishInfo: '{<PUBLISH-INFO>}'
```

Step 4 If **status.attachmentMetadata.publishInfo** exists in the resource obtained in **Step 3**, the problem described in this FAQ is not involved in the workloads created using **pvc-1xx** on the **node-1** node. **node-1** and **pvc-1xx** are the query results in **Step 2**. If **status.attachmentMetadata.publishInfo** does not exist, rectify the fault by referring to **Solution**.

Step 5 If multiple VolumeAttachment resources exist, repeat **Step 3** to **Step 4**.

----End

9.4.7 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 **8.2.2 Viewing 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
```

9.4.8 "I/O error" Is Displayed When a Volume Directory Is Mounted to a Pod

Symptom

When a Pod reads or writes a mounted volume, message "I/O error" is displayed.

Root Cause Analysis

When a protocol such as SCSI is used, if the Pod continuously writes data to the mount directory, the storage device will restart. As a result, the link between the device on the host and the storage device is interrupted, triggering an I/O error. When the storage device is restored, the mount directory is still read-only.

Solution

Remount the volume. That is, reconstruct the Pod to trigger re-mounting.

9.4.9 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 following command to check whether the service is stopped.

```
lsmod | grep iscsi | grep iscsi_tcp
```

The following is an example of the command output.

```
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
```

9.5 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.

9.5.1 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 following command to create the PSP permission.

```
kubectl create -f psp-use.yaml
```

----End

9.5.2 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

9.5.3 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** directory and run the **vi values.yaml** command to open the configuration file.

```
vi values.yaml
```

Step 2 Change the default value **9808** of **controller.livenessProbePort** to an idle port, for example, **9809**.

```
controller:
  livenessProbePort: 9809
```

Step 3 Update Huawei CSI using Helm. For details, see **4.4.1.1 Upgrading Huawei CSI**.

----End

9.5.4 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 the following command 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 following command to check whether the addition is successful.

```
kubectl get psp pks-privileged -o yaml
```

----End

10 Appendix

10.1 Example ALUA Configuration Policy of OceanStor 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 **7.1.1.1 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 **7.1.1.1 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 **7.1.1.1 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.

10.2 Example ALUA Configuration Policy of OceanStor Dorado

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 **7.1.1.1 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 **7.1.1.1 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 **7.1.1.1 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.

10.3 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 **7.1.1.2 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

7.1.1.2 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 **7.1.1.2 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.

10.4 Communication Matrix

Source Device	Host where CSI controller is located	Host where CSI controller is located	Host where CSI node is located	Kubernetes master node
Source IP Address	IP address of the source device	IP address of the source device	IP address of the source device	IP address of the source device
Source Port	1024 to 65536	1024 to 65536	1024 to 65536	1024 to 65536
Destination Device	Storage device	Host where CSI controller is located	Host where CSI node is located	Host where CSI controller is located
Destination IP Address	Management IP address of the storage device	IP address of the destination device	IP address of the destination device	IP address of the destination device
Destination Port (for Listening)	8088	9808	9800	4433
Protocol	TCP	TCP	TCP	TCP
Port Description	Used to create, manage, and delete volumes	Used by Kubernetes to check the health status of CSI controller	Used by Kubernetes to check the health status of CSI node	Used to invoke webhook verification

Listening Port Configurable	No	No	No	Yes
Authentication Mode	User name and password	Certificate	Certificate	Certificate
Encryption Mode	TLS 1.3/TLS 1.2	TLS 1.3/TLS 1.2	TLS 1.3/TLS 1.2	TLS 1.3/TLS 1.2
Plane	OM	O&M plane	O&M plane	O&M plane
Special Scenario	None	None	None	None
Remarks	Enable some source ports.	-	-	For details about how to change the webhook port, see the CSI user guide.

10.5 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 10-1**.
- For SAN resources, configure the minimum permissions by referring to **Table 10-2**.

NOTE

For details about how to configure permissions for user-defined roles, see **OceanStor Dorado 6000, Dorado 18000 Series Product Documentation**.

Table 10-1 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.

Permission Object	Parent Object	Read/Write Permission	Function
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.
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.
port	network	Read-only	Queries logical port information.

Table 10-2 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.

Permission Object	Parent Object	Read/Write Permission	Function
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.
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.

10.6 Huawei CSI Resource Management

This section lists the resource requests and limits used by each container of the Huawei CSI plug-in. For details about the unit, see **Resource units in Kubernetes**.

Table 10-3 Container resource requests and limits

Pod Name	Container Name	CPU Request	CPU Limit	Memory Request	Memory Limit
huawei-csi-controller	huawei-csi-driver	50m	500m	128Mi	1Gi
	storage-backend-sidecar	50m	300m	128Mi	512Mi
	storage-backend-controller	50m	300m	128Mi	512Mi
	huawei-csi-extender	50m	300m	128Mi	512Mi
	csi-attacher	50m	300m	128Mi	512Mi
	csi-provisioner	50m	300m	128Mi	512Mi
	csi-resize	50m	300m	128Mi	512Mi
	csi-snapshotter	50m	300m	128Mi	512Mi
	snapshot-controller	50m	300m	128Mi	512Mi
	liveness-probe	10m	100m	128Mi	128Mi
huawei-csi-node	huawei-csi-driver	50m	500m	128Mi	1Gi
	csi-node-driver-registrar	50m	300m	128Mi	128Mi
	liveness-probe	10m	100m	128Mi	128Mi

Modifying Resource Requests and Limits

If you need to modify the resource requests and limits of a container, perform the following steps (in the following example, Helm is used to install Huawei CSI):

Step 1 If Helm is used for installation, go to the **/helm/esdk/templates** directory. For manual deployment, the file to be modified is in the **/manual/esdk/deploy** directory. For details about the component package path, see **Table 4-1**.

Step 2 Modify the deployment template file.

- If the Pod name is **huawei-csi-controller**, modify the **huawei-csi-controller.yaml** file.
- If the Pod name is **huawei-csi-node**, modify the **huawei-csi-node.yaml** file.

NOTE

For details about Pod names, see **Table 10-3**.

For example, to modify the resource request of the **huawei-csi-driver** container in the Pod named **huawei-csi-node**, run the following command to edit the configuration file and find the container whose

spec.template.spec.containes.name is **huawei-csi-driver**. Modify resource requests and limits as required.

```
vi huawei-csi-node.yaml
```

Edit the following content.

```
containers
 - name: huawei-csi-driver
   ...
   resources:
     limits:
       cpu: 500m
       memory: 1Gi
     requests:
       cpu: 50m
       memory: 128Mi
```

Step 3 If Huawei CSI is not installed, the modification of resource requests and limits takes effect after Huawei CSI is installed by referring to **4.2.1.1 Installing Huawei CSI on Kubernetes, OpenShift, and Tanzu**.

Step 4 If Huawei CSI has been installed, the modification of resource requests and limits takes effect after Huawei CSI is updated by referring to **Upgrading Huawei CSI**.

----End

