User Guide
V2.2.13

Kubernetes CSI

Issue	01
Date	2021-08-23

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://www.huawei.com

Email: support@huawei.com

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 an imminently hazardous situation which, if not avoided, will result in death or serious injury.
WARNING	Indicates a potentially hazardous situation which, if not avoided, could result in death or serious injury.
CAUTION	Indicates a potentially hazardous situation which, if not avoided, may 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	Calls attention to important information, best practices and tips. NOTE is used to address information not related to personal injury, equipment damage, and environment deterioration.

Contents

1 Overview

This document describes how to deploy and use the Kubernetes CSI plug-in so that Huawei enterprise and cloud storage devices provide persistent volume storage capabilities for Kubernetes.

2 Environment Support

- Kubernetes has been deployed and is running properly.
- A Huawei storage device is running properly.
- Tools for scanning disks and mounting files must be installed on the host machine in advance. If containers and services cannot run properly due to lack of system tools, view logs by referring to **8.1 Viewing Log Information** and install the tools on the host.

Table 2-1 Version mappings between Huawei Enterprise storage and Kubernetes

Kubernetes	Huawei Enterprise storage
1.13/1.14/1.15/1.16/1.17/1.18/1.19/1.20/1.21	OceanStor Dorado V6 6.0.RC1/6.0.0/6.0.1/6.1.0/6.1.2 OceanStor Dorado V3 V300R002 OceanStor F V5/V5 V500R007/V500R007 Kunpeng OceanStor F V3/V3 V300R006

Table 2-2 Version mappings between Huawei Distributed storage and Kubernetes

Kubernetes	Huawei Distributed storage
1.13/1.14/1.15/1.16/1.17/1.18/1.19/1.20/1.21	FusionStorage V100R006C30 FusionStorage Block 8.0.0/8.0.1 OceanStor Pacific 8.1.0

Table 2-3 Mappings among Huawei CSI, features, and Kubernetes versions (√: supported, x: not supported)

Feature	1.13	1.14	1.15	1.16	1.17	1.18	1.19	1.20	1.21
Create PVC	√	√	√	√	√	√	√	√	√
Delete PVC	√	√	√	√	√	√	√	√	√
Create Pod	√	√	√	√	√	√	√	√	√
Delete Pod	√	√	√	√	√	√	√	√	√
Offline Resize	x	x	x	√	√	√	√	√	√
Online Resize	x	x	x	√	√	√	√	√	√
Create Snapshot	x	x	x	x	√	√	√	√	√
Delete Snapshot	x	x	x	x	√	√	√	√	√
Restore	x	x	x	x	√	√	√	√	√
Clone	x	x	x	x	√	√	√	√	√

3 Restrictions

This section describes the restrictions on interconnecting CSI with storage.

Table 3-1 Restrictions

Scenario	Restriction	Restricted Storage	Remarks
PVC access mode	**ReadWriteOnce**: SAN/NAS **ReadWriteMany**: NAS **ReadWriteOnly**: SAN/NAS If SAN needs to use the **ReadWriteMany** mode, the Pod service must ensure data consistency.	SAN: OceanStor V3/V5, OceanStor Dorado V3/V6, FusionStorage 8.0 NAS: OceanStor V3/V5, OceanStor Dorado V6, OceanStor Pacific series	
PersistentVolume-Claim	A maximum of 100 items can be created or deleted in a batch.	SAN: OceanStor V3/V5, OceanStor Dorado V3/V6, FusionStorage 8.0 NAS: OceanStor V3/V5, OceanStor Dorado V6, OceanStor Pacific series	The maximum number of concurrent RESTful requests is 100.

Scenario	Restriction	Restricted Storage	Remarks
VolumeSnapshot	N/A	SAN: OceanStor V3/V5, OceanStor Dorado V3/V6, FusionStorage 8.0 NAS: OceanStor V3/V5, Dorado V6	
VolumeSnapshotRestore	Snapshot restoration, that is, creating a PVC from the snapshot instead of rolling back to the original PVC using the snapshot.	SAN: OceanStor V3/V5, OceanStor Dorado V3/V6, FusionStorage 8.0 NAS: OceanStor V3/V5	
VolumeExpansion	• Storage resources can be expanded but cannot be reduced. • PVCs in RWO mode do not support capacity expansion.	SAN: OceanStor V3/V5, OceanStor Dorado V3/V6, FusionStorage 8.0 NAS: OceanStor V3/V5, OceanStor Dorado V6	
VolumeClone	The StorageClass of the source and target PVCs must be the same.	SAN: OceanStor V3/V5, OceanStor Dorado V3/V6, FusionStorage 8.0 NAS: OceanStor V3/V5	

Scenario	Restriction	Restricted Storage	Remarks
HyperMetro	• HyperMetro can be used only when the storage system is normal. If the storage system is faulty, HyperMetro of only provisioned services is normal but new services cannot be provisioned. • HyperMetro and remote replication cannot be configured in the same StorageClass.	SAN: OceanStor V3/V5, OceanStor Dorado V3/V6 NAS: OceanStor V3/V5	
Remote replication	HyperMetro and remote replication cannot be configured in the same StorageClass.	SAN: OceanStor V3/V5, OceanStor Dorado V3/V6 NAS: OceanStor V3/V5, OceanStor Dorado V6	

Scenario	Restriction	Restricted Storage	Remarks
Residual drive letter	Due to a node fault, containerized applications are migrated to other nodes. After the node recovers, residual drive letters exist on the node. Manually clear the residual drive letters. For details, see **8.4 After a Worker Node in the Cluster Breaks Down and Recovers, Pod Failover Is Complete but the Source Host Where the Pod Resides Has Residual Drive Letters**.	SAN: OceanStor V3/V5, OceanStor Dorado V3/V6, FusionStorage 8.0	Condition: iSCSI/FC + Multipath

4 Deployment

4.1 Obtaining the Software Package

You can obtain Huawei Kubernetes CSI through Huawei Kubernetes CSI warehouse.

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

Step 2 Download the package ***eSDK_EnterPrise_Storage_Plugin_.*.****.zip**,*.*.** indicates the release version number.

Step 3 Decompress the package.

Step 4 In the decompressed directory, the package and docs all included.

----End

4.2 Components in the Software Package

Decompress **eSDK_EnterPrise_Storage_Plugin_.*.****eSDK_Cloud_Storage_Plugin_*.***.zip** to obtain the software packages and sample files required for installing and using CSI. The software package structure is shown in the following table.

Table 4-1 Component description

Component	Description
bin/huawei-csi	Implements the CSI API.
bin/secretGenerate	Encrypts plaintext passwords and produces secret objects.
bin/secretUpdate	Encrypts plaintext passwords and updates secret objects.
yamls	yaml sample files used in subsequent deployment

4.3 Creating a Huawei CSI Image

Huawei CSI runs as a container. Currently, Huawei CSI provides only a binary package (**bin/huawei-csi**) which cannot be used directly. Therefore, you need to create a CSI image based on the binary file to start the Huawei CSI service.

Step 1 Prepare a Linux host. Ensure that docker has been installed on the host and the host can access external networks.

Step 2 Log in to the Linux host.

Step 3 Create a directory on the host and go to the directory.

```
# mkdir image; cd image
```

Step 4 Copy the huawei-csi component to the current directory.

Step 5 Create file **Dockerfile**.

```
# vi Dockerfile
```

Step 6 Enter the following content in **Dockerfile** and save the file.

```
FROM ***:***

ADD ["huawei-csi", "/"]
RUN ["chmod", "+x", "/huawei-csi"]

ENTRYPOINT ["/huawei-csi"]
```

NOTICE

*****:***** indicates the basic image and its tag. Ensure that the basic image contains the **glibc** running environment, for example, **busybox:stable-glibc**.

Step 7 Create an image.

```
# docker build -f Dockerfile -t huawei-csi:*.*.* .
```

NOTE

.*. indicates the plug-in version number corresponding to the software package name. If the same image already exists in the environment, use **docker image rm** *<image-id>*.

Step 8 Export the image.

```
# docker save huawei-csi:*.*.* -o huawei-csi.tar
```


Step 9 Copy the **huawei-csi.tar** image file to all worker nodes in the Kubernetes cluster.

```
# scp huawei-csi.tar <user>@<ip>:/<path>
```

Step 10 Import the image.

```
# docker load -i huawei-csi.tar
```

----End

4.4 Interconnecting with Enterprise Storage

This section describes how to interconnect the huawei-csi plug-in with Huawei enterprise storage.

4.4.1 Interconnecting with Enterprise Storage SAN over iSCSI

NOTICE

Before deployment, ensure that:

- An iSCSI client has been installed on all worker nodes.
- All worker nodes can properly connect to the back-end storage management IP address.
- All worker nodes can properly connect to the back-end storage service IP address.
- If multipath networking is used, multipath software has been installed on all worker nodes.

Perform the following steps only on any master node of Kubernetes.

Step 1 Create file **huawei-csi-configmap.yaml** and add the following content to it. For details, see sample file **yamls/huawei-csi-configmap-oceanstor-iscsi.yaml** in the software package.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: huawei-csi-configmap
  namespace: kube-system
data:
  csi.json: |
    {
        "backends": [
            {
                "storage": "oceanstor-san",
                "name": "***",
                "urls": ["https://*.*.*.*:8088", "https://*.*.*.*:8088"],
                "pools": ["***", "***"],
                "parameters": {"protocol": "iscsi", "portals": ["*.*.*.*", "*.*.*.*"]}
            }
        ]
    }
```


Table 4-2 Configuration items

Configuration Item	Format	Description	Remarks
name	String	Name of a storage backend. This parameter is mandatory.	The value can contain uppercase letters, lowercase letters, digits, and hyphens (-).
storage	String	Type of the storage backend. This parameter is mandatory.	The value is fixed to **oceanstor-san**.
pools	List	Name of the used storage pool. This parameter is mandatory.	One or more storage pools are supported.
urls	List	Management URL of the storage backend. This parameter is mandatory.	One or more management URLs are supported. Currently, only IPv4 addresses are supported.
parameters	Dictionary	Variable parameters in the iSCSI scenario. This parameter is mandatory.	In the iSCSI scenario, set **protocol** to **iscsi** and **portals** to the iSCSI service IP address of the storage backend.

Step 2 Create file **huawei-csi-configmap.yaml**.

```
# kubectl create -f huawei-csi-configmap.yaml
```

----End

4.4.2 Interconnecting with Enterprise Storage SAN over FC

NOTICE

Before deployment, ensure that:

- All worker nodes can properly connect to the back-end storage management IP address.
- All worker nodes can properly connect to the storage backend over FC links.
- If multipath networking is used, multipath software has been installed on all worker nodes.
- The systool tool has been installed.

Perform the following steps only on any master node.

Step 1 Create file **huawei-csi-configmap.yaml** and add the following content to it. For details, see sample file **yamls/huawei-csi-configmap-oceanstor-fc.yaml** in the software package.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: huawei-csi-configmap
  namespace: kube-system
data:
  csi.json: |
    {
        "backends": [
            {
                "storage": "oceanstor-san",
                "name": "***",
                "urls": ["https://*.*.*.*:8088", "https://*.*.*.*:8088"],
                "pools": ["***", "***"],
                "parameters": {"protocol": "fc"}
            }
        ]
    }
```

Table 4-3 Configuration items

Configuration Item	Format	Description	Remarks
name	String	Name of a storage backend. This parameter is mandatory.	The value can contain uppercase letters, lowercase letters, digits, and hyphens (-).
storage	String	Type of the storage backend. This parameter is mandatory.	The value is fixed to **oceanstor-san**.
pools	List	Name of the used storage pool. This parameter is mandatory.	One or more storage pools are supported.
urls	List	Management URL of the storage backend. This parameter is mandatory.	One or more management URLs are supported. Currently, only IPv4 addresses are supported.
parameters	Dictionary	Variable parameters in the FC scenario. This parameter is mandatory.	In the FC scenario, set **protocol** to **fc**.

Step 2 Create file **huawei-csi-configmap.yaml**.


```
# kubectl create -f huawei-csi-configmap.yaml
```

----End

4.4.3 Interconnecting with Enterprise Storage NAS over NFS

NOTICE

Before deployment, ensure that:

- An NFS client has been installed on all worker nodes.
- All worker nodes can properly connect to the back-end storage management IP address.
- All worker nodes can properly connect to the IP address of the back-end storage NFS logical port.

Perform the following steps only on any master node.

Step 1 Create file **huawei-csi-configmap.yaml** and add the following content to it. For details, see sample file **yamls/huawei-csi-configmap-oceanstor-nfs.yaml** in the software package.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: huawei-csi-configmap
  namespace: kube-system
data:
  csi.json: |
    {
        "backends": [
            {
                "storage": "oceanstor-nas",
                "name": "***",
                "urls": ["https://*.*.*.*:8088", "https://*.*.*.*:8088"],
                "pools": ["***", "***"],
                "parameters": {"protocol": "nfs", "portals": ["*.*.*.*"]}
            }
        ]
    }
```

Table 4-4 Configuration items

Configuration Item	Format	Description	Remarks
name	String	Name of a storage backend. This parameter is mandatory.	The value can contain uppercase letters, lowercase letters, digits, and hyphens (-).
storage	String	Type of the storage backend. This parameter is mandatory.	The value is fixed to **oceanstor-nas**.

Configuration Item	Format	Description	Remarks
pools	List	Name of the used storage pool. This parameter is mandatory.	One or more storage pools are supported.
urls	List	Management URL of the storage backend. This parameter is mandatory.	One or more management URLs are supported. Currently, only IPv4 addresses are supported.
parameters	Dictionary	Variable parameters in the NAS scenario. This parameter is mandatory.	**portals**: logical port IP address or DNS zone of the specified storage device. Only one logical port IP address or DNS zone can be configured.

Step 2 Create file **huawei-csi-configmap.yaml**.

```
# kubectl create -f huawei-csi-configmap.yaml
```

----End

4.4.4 Interconnecting with Enterprise Storage NVMe over RoCE

NOTICE

Before deployment, ensure that:

- All worker nodes support NVMe over RoCE.
- The NVMe client has been installed on all worker nodes.
- All worker nodes can properly connect to the back-end storage management IP address.
- All worker nodes can properly connect to the back-end storage service IP address.
- If multipath networking is used, multipath software has been installed on all worker nodes.
- Currently, only Dorado V6 supports NVMe over RoCE.
- The host OS must meet storage compatibility requirements.

Perform the following steps only on any master node.

Step 1 Create file **huawei-csi-configmap.yaml** and add the following content to it. For details, see sample file **yamls/huawei-csi-configmap-oceanstor-roce.yaml** in the software package.


```
kind: ConfigMap
apiVersion: v1
metadata:
  name: huawei-csi-configmap
  namespace: kube-system
data:
  csi.json: |
    {
        "backends": [
            {
                "storage": "oceanstor-san",
                "name": "***",
                "urls": ["https://*.*.*.*:8088", "https://*.*.*.*:8088"],
                "pools": ["***", "***"],
                "parameters": {"protocol": "roce", "portals": ["*.*.*.*", "*.*.*.*"]}
            }
        ]
    }
```

Table 4-5 Configuration items

Configuration Item	Format	Description	Remarks
name	String	Name of a storage backend. This parameter is mandatory.	The value can contain uppercase letters, lowercase letters, digits, and hyphens (-).
storage	String	Type of the storage backend. This parameter is mandatory.	The value is fixed to **oceanstor-san**.
pools	List	Name of the used storage pool. This parameter is mandatory.	One or more storage pools are supported.
urls	List	Management URL of the storage backend. This parameter is mandatory.	One or more management URLs are supported. Currently, only IPv4 addresses are supported.
parameters	Dictionary	Variable parameters in the RoCE scenario. This parameter is mandatory.	In the RoCE scenario, set **protocol** to roce and **portals** to the NVMe over RoCE service IP address of the storage backend.

Step 2 Create file **huawei-csi-configmap.yaml**.

```
# kubectl create -f huawei-csi-configmap.yaml
```

----End

4.5 Interconnecting with Distributed Storage

This section describes how to interconnect the huawei-csi plug-in with Huawei cloud storage.

4.5.1 Interconnecting with Distributed Storage SAN over SCSI

NOTICE

Before deployment, ensure that:

- All worker nodes can properly connect to the back-end storage management IP address.
- The FusionStorage VBS client has been installed on all worker nodes.
- All worker nodes have been added to the FusionStorage Block client.

Perform the following steps only on any master node.

Step 1 Create file **huawei-csi-configmap.yaml** and add the following content to it. For details, see sample file **yamls/huawei-csi-configmap-fusionstorage-scsi.yaml** in the software package.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: huawei-csi-configmap
  namespace: kube-system
data:
  csi.json: |
    {
        "backends": [
            {
                "storage": "fusionstorage-san",
                "name": "***",
                "urls": ["https://*.*.*.*:28443"],
                "pools": ["***", "***"],
                "parameters": {"protocol": "scsi", "portals": [{"***": "*.*.*.*", "***": "*.*.*.*"}]}
            }
        ]
    }
```

Table 4-6 Configuration items

Configuration Item	Format	Description	Remarks
name	String	Name of a storage backend. This parameter is mandatory.	The value can contain uppercase letters, lowercase letters, digits, and hyphens (-).
storage	String	Type of the storage backend. This parameter is mandatory.	The value is fixed to **fusionstorage-san** for interconnection with FusionStorage Block.

Configuration Item	Format	Description	Remarks
pools	List	Name of the used storage pool. This parameter is mandatory.	One or more storage pools are supported.
urls	List	Management URL of the storage backend. This parameter is mandatory.	Management URL of FusionStorage. Only one URL can be configured.
parameters	Dictionary	Variable parameter. This parameter is mandatory.	Set **protocol** to **scsi** and **portals** to the IP pair list of the hosts and VBS nodes. The parameter format is **[{"hostname":"*.*.*.*"}]**, where **hostname** indicates the host name of a worker node and ***.*.*.*** indicates the management IP address of the FusionStorage Block client. If there are multiple worker nodes, configure them in dictionary format, separated by commas(,).

Step 2 Create file **huawei-csi-configmap.yaml**.

```
# kubectl create -f huawei-csi-configmap.yaml
```

----End

4.5.2 Interconnecting with Distributed Storage SAN over iSCSI

NOTICE

Before deployment, ensure that:

- The iSCSI client has been installed on all worker nodes.
- All worker nodes can properly connect to the back-end storage management IP address.
- All worker nodes can properly connect to the back-end storage service IP address.
- If multipath networking is used, multipath software has been installed on all worker nodes.
- Worker nodes name can only contain digits, letters, underscores (_), hyphens (-), periods (.), and colons (:) and starts with a digit, letter, or underscore, and the name length cannot exceed 31 characters.
- Only FusionStorage 8.0 and later versions support iSCSI mode.

Perform the following steps only on any master node.

Step 1 Create file **huawei-csi-configmap.yaml** and add the following content to it. For details, see sample file **yamls/huawei-csi-configmap-fusionstorage-iscsi.yaml** in the software package.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: huawei-csi-configmap
  namespace: kube-system
data:
  csi.json: |
    {
        "backends": [
            {
                "storage": "fusionstorage-san",
                "name": "***",
                "urls": ["https://*.*.*.*:28443"],
                "pools": ["***", "***"],
                "parameters": {"protocol": "iscsi", "portals": ["*.*.*.*", "*.*.*.*"]}
            }
        ]
    }
```

Table 4-7 Configuration items

Configuration Item	Format	Description	Remarks
name	String	Name of a storage backend. This parameter is mandatory.	The value can contain uppercase letters, lowercase letters, digits, and hyphens (-).

Configuration Item	Format	Description	Remarks
storage	String	Type of the storage backend. This parameter is mandatory.	The value is fixed to **fusionstorage-san** for interconnection with FusionStorage Block.
pools	List	Name of the used storage pool. This parameter is mandatory.	One or more storage pools are supported.
urls	List	Management URL of the storage backend. This parameter is mandatory.	Management URL of FusionStorage.
parameters	Dictionary	Variable parameters. This parameter is mandatory.	In the iSCSI scenario, set **protocol** to **iscsi** and **portals** to the iSCSI service IP addresses of the storage backends separated by commas (,).

Step 2 Create file **huawei-csi-configmap.yaml**.

```
# kubectl create -f huawei-csi-configmap.yaml
```

----End

4.5.3 Interconnecting with Distributed Storage NAS over NFS

NOTICE

Before deployment, ensure that:

- An NFS client has been installed on all worker nodes.
- All worker nodes can properly connect to the back-end storage management IP address.
- All worker nodes can properly connect to the IP address of the back-end storage NFS logical port.

Perform the following steps only on any master node.

Step 1 Create file **huawei-csi-configmap.yaml** and add the following content to it. For details, see sample file **yamls/huawei-csi-configmap-fusionstorage-nfs.yaml** in the software package.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: huawei-csi-configmap
  namespace: kube-system
data:
```



```
csi.json: |
  {
      "backends": [
          {
              "storage": "fusionstorage-nas",
              "name": "***",
              "urls": ["https://*.*.*.*:28443"],
              "pools": ["***", "***"],
              "parameters": {"protocol": "nfs", "portals": ["*.*.*.*"]}
          }
      ]
  }
```

Table 4-8 Configuration items

Configuration Item	Format	Description	Remarks
name	String	Name of a storage backend. This parameter is mandatory.	The value can contain uppercase letters, lowercase letters, digits, and hyphens (-).
storage	String	Type of the storage backend. This parameter is mandatory.	The value is fixed to **fusionstorage-nas**.
pools	List	Name of the used storage pool. This parameter is mandatory.	One or more storage pools are supported.
urls	List	Management URL of the storage backend. This parameter is mandatory.	Management URL of FusionStorage.
parameters	Dictionary	Variable parameters in the NAS scenario. This parameter is mandatory.	**portals**: logical port IP address of the specified storage device. Only one logical port IP address can be configured.

Step 2 Create file **huawei-csi-configmap.yaml**.

```
# kubectl create -f huawei-csi-configmap.yaml
```

----End

4.6 Starting the huawei-csi Service

NOTICE

An image may need to be downloaded during the procedure. Therefore, worker nodes in the Kubernetes cluster must be able to access external networks. In an intranet environment, obtain the image package in other ways and manually import it into all worker nodes.

Perform the following steps only on any master node.

Step 1 Use the encryption tool to enter the storage user name and password.

- Initial configuration

```
# chmod +x secretGenerate
# ./secretGenerate
*************************The 1 Backend Info**************************
Current backend name is: <backend-1-name>
Current backend url is: [<backend-1-url>]
*********************************************************************
Enter backend <backend-1-name>'s user:       #Enter the user name of storage 1.
Enter backend <backend-1-name>'s password:   #Enter the password of storage 1.
Please Enter the password again:             #Enter the password of storage 1 again.

*********************Create CSI Secret Successful**********************
```

- Non-initial configuration

```
# chmod +x secretGenerate
# ./secretGenerate
The secret info is exist. Do you force to update it? Y/N

*************************The 1 Backend Info**************************
Current backend name is: <backend-1-name>
Current backend url is: [<backend-1-url>]
*********************************************************************
Enter backend <backend-1-name>'s user:       #Enter the user name of storage 1.
Enter backend <backend-1-name>'s password:   #Enter the password of storage 1.
Please Enter the password again:             #Enter the password of storage 1 again.

*********************Create CSI Secret Successful**********************
```

- Configuring multiple backends (For details about how to configure multiple backends, see **7.1 Configuring Multiple Backends**.)

```
# chmod +x secretGenerate
# ./secretGenerate
*************************The 1 Backend Info**************************
Current backend name is: <backend-1-name>
Current backend url is: [<backend-1-url>]
*********************************************************************
Enter backend <backend-1-name>'s user:       #Enter the user name of storage 1.
Enter backend <backend-1-name>'s password:   #Enter the password of storage 1.
Please Enter the password again:             #Enter the password of storage 1 again.

*************************The 2 Backend Info**************************
Current backend name is: <backend-2-name>
Current backend url is: [<backend-2-url>]
*********************************************************************
Enter backend <backend-2-name>'s user:       #Enter the user name of storage 2.
Enter backend <backend-2-name>'s password:   #Enter the password of storage 2.
Please Enter the password again:             #Enter the password of storage 2 again.

*********************Create CSI Secret Successful**********************
```


- Run the following check command after the operation is complete:

```
# kubectl get secret -n kube-system | grep huawei-csi-secret
huawei-csi-secret                        Opaque                                1      8m
```

Step 2 Compile the **huawei-csi-rbac.yaml** file. For details, see sample file **yamls/huawei-csi-rbac.yaml** in the software package. For details of resize and snapshot, please see: **Appendix: Details of th...**

Create RBAC permissions.

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

Step 3 Compile the **huawei-csi-controller.yaml** file. For details, see sample file **yamls/huawei-csi-controller.yaml** in the software package. For details of resize and snapshot, please see: **Appendix: Details of the...**

Start the controller service.

```
# kubectl create -f huawei-csi-controller.yaml
```

NOTE

In the **huawei-csi:*.*.*** field in the yaml file, replace ***.*.*** with the version number of the created Huawei CSI image.

Step 4 Compile the **huawei-csi-node.yaml** file. For details, see sample file **yamls/huawei-csi-node.yaml** in the software package.

Start the node service.

```
# kubectl create -f huawei-csi-node.yaml
```

NOTE

- In the **huawei-csi:*.*.*** field in the yaml file, replace ***.*.*** with the version number of the created Huawei CSI image.
- In the **huawei-csi-driver** parameter **args** in the **.yaml** file, **--volume-use-multipath** indicates that multipathing is enabled by default. To modify the parameter, run the following commands:

```
args:
 - "--endpoint=/csi/csi.sock"
 - "--containerized"
 - "--driver-name=csi.huawei.com"
 - "--volume-use-multipath=false"
```

Step 5 After the preceding steps are complete, the containerized huawei-csi service is deployed. Run the following check command:

```
# kubectl get pod -A | grep huawei
kube-system   huawei-csi-controller-695b84b4d8-tg64l   3/3   Running   0   14s
kube-system   huawei-csi-node-g6f7z                    2/2   Running   0   14s
```

----End

- **Appendix: Details of the resize and snapshot**

NOTE

The csi-resizer has been supported since Kubernetes v1.16

- For the huawei-csi-rbac.yaml file, please refer to the yamls/huawei-csi-resize-rbac.yaml in the software package
- For the huawei-csi-controller.yaml file, please refer to the yamls/huawei-csi-resize-controller.yaml in the software package

The csi-snapshotter has been supported since Kubernetes v1.17

- For the huawei-csi-rbac.yaml file, please refer to the yamls/huawei-csi-resize-snapshot-rbac.yaml in the software package
- For the huawei-csi-controller.yaml file, please refer to the yamls/huawei-csi-resize-snapshot-controller.yaml in the software package

5 Upgrade Operations

NOTICE

- The CSI upgrade does not affect delivered resources such as PVCs, snapshots, and Pods.
- During the upgrade, CSI cannot be used to deliver new resources.

5.1 Uninstalling the Original CSI

Perform the following steps only on any master node of Kubernetes.

Step 1 Delete the node service. (**huawei-csi-node.yaml** is the configuration information in **Step 4**.)

```
# kubectl delete -f huawei-csi-node.yaml
```

Run the following command to check the containerized huawei-csi-node service. If no command output is displayed, the deletion is complete.

```
# kubectl get pod -A | grep huawei-csi
kube-system   huawei-csi-node-g6f7z                 2/2     Running     0          14s
```

Step 2 Delete the controller service. (**huawei-csi-controller.yaml** is the configuration information in **Step 3**.)

```
# kubectl delete -f huawei-csi-controller.yaml
```

Run the following command to check the containerized huawei-csi-controller service. If no command output is displayed, the deletion is complete.

```
# kubectl get pod -A | grep huawei-csi
kube-system   huawei-csi-controller-695b84b4d8-tg64l   3/3     Running     0          14s
```

Step 3 Delete RBAC permissions. (**huawei-csi-rbac.yaml** is the configuration information in **Step 2**.)

```
# kubectl delete -f huawei-csi-rbac.yaml
```

Step 4 Delete the **configmap** object.


```
# kubectl delete configmap huawei-csi-configmap -n <namespace-name>
```

Run the following command to check the **configmap** information of **huawei-csi**. If no command output is displayed, the deletion is complete.

```
# kubectl get configmap -A | grep huawei-csi-configmap
kube-system   huawei-csi-configmap              1      129m
```

Step 5 Delete the **secret** object.

```
# kubectl delete secret huawei-csi-secret -n <namespace-name>
```

Run the following command to check the **secret** information of **huawei-csi**. If no command output is displayed, the deletion is complete.

```
# kubectl get secret -A | grep huawei-csi-secret
kube-system       huawei-csi-secret              Opaque              1      20m
```

----End

5.2 Installing a New CSI

For details about how to install a new CSI, see **4 Deployment**.

NOTICE

- In upgrade scenarios, ensure that the values of **storage**, **name**, and **pools** are the same as those in the original configuration file.

```
    "backends": [
        {
            "storage": "oceanstor-san",
            "name": "***",
            "urls": ["https://*.*.*.*:8088", "https://*.*.*.*:8088"],
            "pools": ["***", "***"],
            "parameters": {"protocol": "iscsi", "portals": ["*.*.*.*", "*.*.*.*"]}
        }
    ]
```

- In upgrade scenarios, you need to change the values of **urls** and **parameters**. For details, see **4.4 Interconnecting with Enterprise Storage** or **4.5 Interconnecting with Distributed Storage**.

5.3 Updating CSI

NOTICE

- When CSI is updated, the resource services created by Huawei CSI can run properly.
- If CSI needs to continue managing the original resources after the update, ensure that **huawei-csi-configmap** (**4.4 Interconnecting with Enterprise Storage** or **4.5 Interconnecting with Distributed Storage**) remains unchanged.

Perform the following steps only on any master node of Kubernetes.

Updating the configmap Object

Step 1 View the **configmap** object.

```
# kubectl get configmap huawei-csi-configmap -n <namespace-name>
```

Step 2 Delete the **huawei-csi-node** and **huawei-csi-controller** objects.

```
# kubectl delete -f huawei-csi-controller.yaml && kubectl delete -f huawei-csi-node.yaml
```

Step 3 Delete the **configmap** object.

```
# kubectl delete configmap huawei-csi-configmap -n <namespace-name>
```

Step 4 Edit the **huawei-csi-configmap.yaml** file according to **4.4 Interconnecting with Enterprise Storage** or **4.5 Interconnecting with Distributed Storage**.

Step 5 Create the **configmap** object.

```
# kubectl create -f huawei-csi-configmap.yaml
```

Step 6 Createthe **huawei-csi-node** and **huawei-csi-controller** objects.

```
# kubectl create -f huawei-csi-controller.yaml && kubectl create -f huawei-csi-node.yaml
```

----End

Updating the secret Object

Step 1 Run the following command to add the execute permission on **secretUpdate** (**4.2 Components in the Software Package**).

```
# chmod +x secretUpdate
```

Step 2 Run the **secretUpdate** tool and enter information as prompted.

```
# ./secretUpdate
**************************All Secret Info*****************************
<current secret info>
**********************************************************************
**************************All Backend Info*****************************
The 1 backend name is: <backend-1-name>   backend url is: [<backend-1-url>]
Do you want to update it? Y/N               #Determine whether to update the object.
Enter backend <backend-1-name>'s user:      #Enter the user name of storage 1.
Enter backend <backend-1-name>'s password:  #Enter the password of storage 1.
Please Enter the password again:            #Enter the password of storage 1 again.

The 2 backend name is: <backend-2-name>   backend url is: [<backend-2-url>]
Do you want to update it? Y/N               #Determine whether to update the object.
Enter backend <backend-2-name>'s user:      #Enter the user name of storage 2.
Enter backend <backend-2-name>'s password:  #Enter the password of storage 2.
Please Enter the password again:            #Enter the password of storage 2 again.

********************Update CSI Secret Successful*********************
```

----End

6 Instructions for Use

This chapter describes how to use Huawei storage to provide PersistentVolume for Kubernetes.

6.1 Creating StorageClass

6.1.1 Creating LUN StorageClass

Perform the following steps only on any master node.

Step 1 Configure the StorageClass yaml file.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: "***"
provisioner: "csi.huawei.com"
parameters:
  volumeType: "lun"
  allocType: "thin"
  cloneFrom: "**"
  cloneSpeed: "**"
  fsType: "**"
```

Table 6-1 Parameter description

Parameter	Description	Remarks
name	User-defined name of a StorageClass object.	

Parameter	Description	Remarks
provisioner	provisioner identifier.	The value is fixed to **csi.huawei.com**.
volumeType	Type of the volume to be created.	The value is fixed to **lun**.
allocType	How the volume is allocated.	This parameter is optional. The value can be **thin** or **thick**, and the default value is **thin**.
cloneFrom	Original volume of the specified clone.	This parameter is optional. The format is *Storage backend name.Original volume name*.
cloneSpeed	Speed of the specified clone.	This parameter is optional. The value ranges from 1 to 4 and the default value is **3**. **4** indicates the highest speed.
fsType	Type of the specified file system.	This parameter is optional. The value can be **ext2**, **ext3**, or **ext4**, and the default value is **ext4**.

Step 2 Run the following command to create StorageClass based on the yaml file.

```
# kubectl create -f /path/to/yaml/file
```

----End

6.1.2 Creating File System StorageClass

Perform the following steps only on any master node.

Step 1 Configure the StorageClass yaml file.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: "***"
provisioner: "csi.huawei.com"
parameters:
  volumeType: "fs"
  allocType: "thin"
  authClient: "*"
  cloneFrom: "**"
  cloneSpeed: "**"
```

Table 6-2 Parameter description

Parameter	Description	Remarks
name	User-defined name of a StorageClass object.	
provisioner	provisioner identifier.	The value is fixed to **csi.huawei.com**.

Parameter	Description	Remarks
volumeType	Type of the volume to be created.	The value is fixed to **fs**.
authClient	Client that can access the FS volume.	This parameter is mandatory. You can enter the client host name, client IP address, or client IP address segment, or use asterisks (*) to represent all client IP addresses. You can specify multiple clients which are separated by semicolons (;).
allocType	How the volume is allocated.	This parameter is optional. The value can be **thin** or **thick**, and the default value is **thin**.
cloneFrom	Original volume of the specified clone.	This parameter is optional. The format is *Storage backend name.Original volume name*.
cloneSpeed	Speed of the specified clone.	This parameter is optional. The value ranges from 1 to 4 and the default value is **3**. **4** indicates the highest speed.

Step 2 Run the following command to create StorageClass based on the yaml file.

```
# kubectl create -f /path/to/yaml/file
```

----End

6.2 Creating PersistentVolumeClaim

Perform the following steps only on any master node.

Step 1 Configure the PersistentVolumeClaim yaml file.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: "***"
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: "***"
  resources:
    requests:
      storage: ***Gi
```


Table 6-3 Parameter description

Parameter	Description	Remarks
name	User-defined name of a PersistentVolumeClaim object.	
storageClassName	Name of the StorageClass object.	Set this parameter to the name of the StorageClass object created in **6.1 Creating StorageClass**.
storage	Size of the volume to be created.	The value format is ***Gi. The unit is GB.
accessModes	Access mode of the volume.	LUN volumes support ReadWriteOnce, ReadOnlyMany, and ReadWriteMany. If the ReadWriteMany mode is used and multiple Pods access the volume at the same time, the Pod service must ensure data consistency. FS volumes support ReadWriteOnce, ReadOnlyMany, and ReadWriteMany.

Step 2 Run the following command to create PersistentVolumeClaim based on the yaml file.

```
# kubectl create -f /path/to/yaml/file
```

----End

6.3 Creating Pod

Perform the following steps only on any master node.

Step 1 Configure the Pod yaml file.

```
kind: Pod
apiVersion: v1
metadata:
  name: "***"
spec:
  containers:
    - name: "***"
      image: "***"
      volumeMounts:
        - name: mypv
          mountPath: "***"
  volumes:
    - name: mypv
      persistentVolumeClaim:
        claimName: "***"
```


Table 6-4 Parameter description

Parameter	Description	Remarks
metadata:name	User-defined name of a Pod object.	
spec:containers:name	User-defined container name.	
spec:containers:image	Container image.	
spec:containers:image:volumeMounts:mountPath	Mount path of the PersistentVolumeClaim object in the container.	
spec:volumes:persistentVolumeClaim:claimName	Name of the PersistentVolumeClaim object.	Set this parameter to the name of the PersistentVolumeClaim object created in **6.2 Creating PersistentVolumeClaim6.1 Creating StorageClass**.

Step 2 Run the following command to create Pod based on the yaml file.

```
# kubectl create -f /path/to/yaml/file
```

----End

6.4 Creating Snapshots

The CSI supports snapshot v1beta1 since Kubernetes 1.17. For details, please see: **https://kubernetes-csi.github.io/docs/external-snapshotter.html**

6.4.1 Installing the Snapshot-Dependent Component Service

Perform the following steps only on any master node.

Prerequisites

Before the installation, check whether the snapshot-related resource service has been installed. Run **kubectl api-resources | grep snapshot | awk '{print $1}'** on the master server. If the following information is displayed, you do not need to install the service again.

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

If the result does not contain the preceding service, perform the following operations to install the service.

Step 1 Compile **volumesnapshotclasses.yaml** file if missing volumesnapshotclasses . For details, see sample file **yamls/snapshot.storage.k8s.io_volumesnapshotclasses.yaml** in the software package.

Create file **volumesnapshotclasses.yaml**.

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

Step 2 Compile **volumesnapshotcontents.yaml** file if missing volumesnapshotcontents . For details, see sample file **yamls/snapshot.storage.k8s.io_volumesnapshotcontents.yaml** in the software package.

Create file **volumesnapshotcontents.yaml**.

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

Step 3 Compile **volumesnapshots.yaml** file if missing volumesnapshots . For details, see sample file **yamls/snapshot.storage.k8s.io_volumesnapshots.yaml** in the software package.

Create file **volumesnapshots.yaml**.

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

Step 4 Configure according to **4.6 Starting the huawei-csi Service**.

Step 5 For details about snapshot, see **Appendix: Details of the resize and snapshot**. If **huawei-csi-resize-snapshot-controller.yaml** is used for deployment, the containerized huawei-csi service is displayed as follows:

```
# kubectl get pod -A | grep huawei
kube-system     huawei-csi-controller-fd5f97768-qlldc     6/6     Running     0     16s
kube-system     huawei-csi-node-25txd                     2/2     Running     0     15s
```

----End

6.4.2 Creating VolumeSnapshotClass

Step 1 Configure the VolumeSnapshotClass yaml file. For details, see sample file **yamls/snapshotclass.yaml** in the software package.

```
apiVersion: snapshot.storage.k8s.io/v1beta1
kind: VolumeSnapshotClass
metadata:
  name: ***
driver: csi.huawei.com
deletionPolicy: Delete
```

Table 6-5 Parameter description

Parameter	Description	Remarks
name	User-defined name of a VolumeSnapshotClass object.	
driver	driver identifier.	The value is fixed to **csi.huawei.com**.
deletionPolicy	The policy of volumeSnapshotContent while VolumeSnapshot is deleted.	This parameter is mandatory. The value can be **Delete** or **Retain**.

Step 2 Run the following command to create VolumeSnapshotClass based on the yaml file.

```
# kubectl create -f /path/to/yaml/file
```

----End

6.4.3 Creating VolumeSnapshot

Step 1 Configure the VolumeSnapshot yaml file. For details, see sample file **yamls/snapshot.yaml** in the software package.

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

Table 6-6 Parameter description

Parameter	Description	Remarks
name	User-defined name of a VolumeSnapshot object.	
volumeSnapshotClassName	Name of the VolumeSnapshotClass object.	Set this parameter to the name of the VolumeSnapshotClass object created in **6.4.2 Creating VolumeSnapshotClass**.
persistentVolumeClaimName	The name of source PersistentVolumeClaim.	Set this parameter to the name of the PersistentVolumeClaim object created in **6.2 Creating PersistentVolumeClaim**.

Step 2 Run the following command to create VolumeSnapshot based on the yaml file.

```
# kubectl create -f /path/to/yaml/file
```

----End

6.5 Expanding Capacity

6.5.1 Installing the Component Service on Which Capacity Expansion Depends

For details about capacity expansion, see **Appendix: Details of the resize and snapshot**. If **huawei-csi-resize-snapshot-controller.yaml** is used for deployment, the containerized huawei-csi service is displayed as follows:

```
# kubectl get pod -A | grep huawei
kube-system     huawei-csi-controller-fd5f97768-qlldc     6/6     Running     0     16s
kube-system     huawei-csi-node-25txd                     2/2     Running     0     15s
```


6.5.2 Expanding PVC

Step 1 Configure the **StorageClass** yaml file by referring to **6.1 Creating StorageClass**. Add the **allowVolumeExpansion** configuration item. The following is an example:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: "***"
provisioner: "csi.huawei.com"
parameters:
  ...
allowVolumeExpansion: true
```

Step 2 Create PersistentVolumeClaim by referring to **6.2 Creating PersistentVolumeClaim**.

Step 3 Online expansion and offline expansion are supported. The operation command is as follows:

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

mypvc is the name of the PVC that needs to be expanded, and **120Gi** is the capacity after expansion.

----End

6.6 Create PVC From Source

When creating a PersistentVolumeClaim, you can create a PVC through the storage function by specifying the source object. The currently supported sources are PVC and VolumeSnapshot. For details, see **https://kubernetes-csi.github.io/docs/volume-datasources.html**.

6.6.1 Clone

Step 1 Configure the PersistentVolumeClaim yaml file.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: ***
spec:
  storageClassName: ***
  dataSource:
    name: ***
    kind: PersistentVolumeClaim
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: ***Gi
```

Table 6-7 Parameter description

Parameter	Description	Remarks
metadata/ name	User-defined name of a PersistentVolumeClaim object.	

Parameter	Description	Remarks
spec/ storageClassName	Name of the StorageClass object.	Set this parameter to the name of the StorageClass object created in **6.1 Creating StorageClass**. It must be the same as the StorageClass of the objects in **dataSource**.
spec/ dataSource/ name	Name of the source PersistentVolumeClaim object	
spec/ resources/ requests/ storage	Size of the volume to be created.	No less than the source PVC capacity. The value format is ***Gi. The unit is GB.

Step 2 Run the following command to create PersistentVolumeClaim based on the yaml file.

```
# kubectl create -f /path/to/yaml/file
```

----End

6.6.2 Snapshot Restore

Step 1 Configure the PersistentVolumeClaim yaml file.

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: ***
spec:
  storageClassName: ***
  dataSource:
    name: ***
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: ***Gi
```

Table 6-8 Parameter description

Parameter	Description	Remarks
metadata/ name	User-defined name of a PersistentVolumeClaim object.	

Parameter	Description	Remarks
spec/ storageClassName	Name of the StorageClass object.	Set this parameter to the name of the StorageClass object created in **6.1 Creating StorageClass**. It must be the same as the StorageClass of the snapshot source PVC in **dataSource**.
spec/ dataSource/ name	Name of the source VolumeSnapshot object	
spec/ resources/ requests/ storage	Size of the volume to be created.	No less than the source VolumeSnapshot capacity. The value format is ***Gi**. The unit is GB.

Step 2 Run the following command to create PersistentVolumeClaim based on the yaml file.

```
# kubectl create -f /path/to/yaml/file
```

----End

7 Advanced Features

This chapter describes how to configure advanced features of Huawei storage.

7.1 Configuring Multiple Backends

Huawei CSI allows you to configure multiple backend which are separated by commas (,) (not required by the last backend). For details about each backend, see section **4.4 Interconnecting with Enterprise Storage** or **4.5 Interconnecting with Distributed Storage**.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: huawei-csi-configmap
  namespace: kube-system
data:
  csi.json: |
    {
        "backends": [
            {
                "storage": "***",
                "product": "***",
                "name": "backend1",
                ...
            },
            {
                "storage": "***",
                "product": "***",
                "name": "backend2",
                ...
            }
        ]
    }
```


7.2 Creating a Volume in the Specified Storage Backend

In the scenario where multiple storage backends are configured, you can specify a storage backend where you want to create a volume.

Configure the StorageClass yaml file. Add the **backend** configuration item under the **parameters** configuration item. The following is an example:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: "***"
provisioner: "csi.huawei.com"
parameters:
  ...
  backend: "***"
```

The value of **backend** is the name of a storage backend configured in the **huawei-csi-configmap.yaml** file.

The volume created using the StorageClass object will be created in the specified storage backend.

7.3 Creating a Volume in the Specified Storage Pool

In the scenario where multiple storage pools are configured, you can specify a storage pool where you want to create a volume.

Configure the StorageClass yaml file. Add the **pool** configuration item under the **parameters** configuration item. The following is an example:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: "***"
provisioner: "csi.huawei.com"
parameters:
  ...
  pool: "***"
```

The value of **pool** is the name of a storage pool.

NOTE

The volume created using the StorageClass object will be created in the specified storage pool. The existing PVC does not change the storage pool information.

7.4 Configuring ALUA

At the scenario of integrating with SAN/Block storage, and multipath involved, ALUA configuration is supported.

7.4.1 Configuring ALUA for Enterprise Storage

Modify the **huawei-csi-configmap.yaml** file. Add the ALUA parameter in the **parameters** configuration item.

```
{
  "backends": [
  {
    "storage": "oceanstor-san",
    ...
    "parameters": {..., "ALUA": {"<HostName>": {"MULTIPATHTYPE": "*", "FAILOVERMODE": "*",
"SPECIALMODETYPE": "*", "PATHTYPE": "*"}, "<HostName>": {...}}}
  }
  ]
}
```

Table 7-1 ALUA parameter description

Parameter	Description	Remarks
<HostName>	Regular expression of the host name. The worker node with the hostname matching it will use the corresponding ALUA configuration.	If configured as *, it indicates a default configuration. If one worker node cannot match any other <HostName> regular expression, it'll use the default one.
MULTIPATHTYPE	Multipathing type. 0: default 1: uses third-party multipathing software	
FAILOVERMODE	Initiator switchover mode. 0: early-version ALUA 1: common ALUA 2: ALUA not used 3: special ALUA	This parameter needs to be delivered only when uses third-party multipathing software is enabled, early-version ALUA is not supported in V5 all series.
SPECIALMODETYPE	Initiator special mode type. 0: mode 0 1: mode 1 2: mode 2 3: mode 3	This parameter needs to be delivered only when initiator switchover mode is special ALUA.
PATHTYPE	Initiator path type. 0: optimal path 1: non-optimal path	This parameter needs to be delivered only when uses third-party multipathing software is enabled.

NOTE

1. For different OS, the ALUA configuration may have discrepancy. Visit **https://support.huawei.com/enterprise/zh/index.html**, input "Host Connectivity Guide" in the search bar and start search. At the search results, choose the guide document according with your OS, and configure ALUA according to the recommendation from that guide.
2. This configuration way is suitable for Enterprise V3/V5 storage and Dorado V3.
3. A node that with a Pod provisioned does not proactively change ALUA information. The host ALUA configuration can be changed only after a Pod is provisioned again on the node.

7.4.2 Configuring ALUA for Dorado V6

Modify the **huawei-csi-configmap.yaml** file. Add the ALUA parameter in the **parameters** configuration item.

```
{
  "backends": [
  {
    "storage": "oceanstor-san",
    ...
    "parameters": {..., "ALUA": {"<HostName>": {"accessMode": "*", "hyperMetroPathOptimized": "*"}, "<HostName>": {...}}}
  }
  ]
}
```

Table 7-2 ALUA parameter description

Parameter	Description	Remarks
<HostName>	Regular expression of the host name. The worker node with the hostname matching it will use the corresponding ALUA configuration.	If configured as *, it indicates a default configuration. If one worker node cannot match any other <HostName> regular expression, it'll use the default one.
accessMode	Host access mode. 0: "balanced". 1: "asymmetric".	
hyperMetroPathOptimized	Whether the host path to the local HyperMetro array is preferred. 1: Yes 0: No	

NOTE

1. For different OS, the ALUA configuration may have discrepancy. Visit **https://support.huawei.com/enterprise/zh/index.html**, input "Host Connectivity Guide" in the search bar and start search. At the search results, choose the guide document for according with your OS, and configure ALUA according to the recommendation from that guide.
2. This configuration way is suitable for Dorado V6.
3. A node that with a Pod provisioned does not proactively change ALUA information. The host ALUA configuration can be changed only after a Pod is provisioned again on the node.

7.4.3 Configuring ALUA for Distributed Storage

Modify the **huawei-csi-configmap.yaml** file. Add the ALUA parameter in the **parameters** configuration item.

```
{
  "backends": [
  {
    "storage": "fusionstorage-san",
    ...
    "parameters": {..., "ALUA": {"<HostName>": {"switchoverMode": "*", "pathType": "*"},
"<HostName>": {...}}}
  }
  ]
}
```

Table 7-3 ALUA parameter description

Parameter	Description	Remarks
<HostName>	Regular expression of the host name. The worker node with the hostname matching it will use the corresponding ALUA configuration.	If configured as *, it indicates a default configuration. If one worker node cannot match any other <HostName> regular expression, it'll use the default one.
switchoverMode	Switchover mode. "Disable_alua": disables ALUA. "Enable_alua": enables ALUA.	
pathType	Path type. "optimal_path": preferred path. "non_optimal_path": non-preferred path.	

NOTE

1. Only available for iSCSI scenario of Distributed Storage.
2. A node that with a Pod provisioned does not proactively change ALUA information. The host ALUA configuration can be changed only after a Pod is provisioned again on the node.

7.5 Advanced Features of Enterprise Storage

7.5.1 Configuring QoS

NOTE

1. The QoS feature is not a standard feature of Kubernetes and is customized by storage vendors.
2. The QoS policy can be specified only when a PVC is created.
3. A created PVC cannot be modified on Kubernetes.

This section describes how to create a LUN or file system volume that supports QoS.

Step 1 Configure the **StorageClass** yaml file. Add the **qos** configuration item under the **parameters** configuration item. The following is an example:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: "***"
provisioner: "csi.huawei.com"
parameters:
  ...
  qos: '{"IOTYPE": 2, "MINIOPS": 1000}'
```

The value of the **qos** configuration item is JSON character strings in dictionary format. A character string is enclosed by single quotation marks and the dictionary key by double quotation marks.

Step 2 Use the StorageClass to create a PVC.

----End

Table 7-4 qos parameters

Parameter	Description	Remarks
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. Valid values are as follows: • **0**: read I/O • **1**: write I/O • **2**: read and write I/Os

Parameter	Description	Remarks
MAXBANDWIDTH	Maximum bandwidth.	The value is a positive integer, expressed in MB/s.
MINBANDWIDTH	Minimum bandwidth.	The value is a positive integer, expressed in MB/s.
MAXIOPS	Maximum IOPS.	The value is a positive integer.
MINIOPS	Minimum IOPS.	The value is a positive integer.
LATENCY	Maximum latency.	The value is a positive integer, expressed in ms.

NOTE

1. **MAXBANDWIDTH** or **MAXIOPS** cannot coexist with **MINBANDWIDTH**, **MINIOPS**, or **LATENCY**.
2. For OceanStor Dorado, **IOTYPE** must be set to **2** (read and write I/Os) and **MINBANDWIDTH**, **MINIOPS**, and **LATENCY** are unavailable.
3. vStore users do not support QoS policies.
4. 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.

7.5.2 Configuring vStore

Modify the **huawei-csi-configmap.yaml** file and add the **vstoreName** field.

```
{
  "backends": [
  {
    ...
    "user": "***",
    "password": "***",
    "vstoreName": "***"
  }
  ]
}
```

NOTE

- **user**, **password**, and **vstoreName** are the vStore user name, vStore user password, and vStore name configured on the storage device in advance.
- After the **huawei-csi-configmap.yaml** file is configured, you need to restart **huawei-csi-controller** and **huawei-csi-node**. Otherwise, the configuration does not take effect.

7.5.3 Configuring SAN Replication

To use SAN replication, finish the replication relationship configuration between two Huawei storage in advance. Please refer to relevant Huawei storage instruction for specifics.

In the **backends** section in the **huawei-csi-configmap.yaml** file, add two backends that form the replication relationship and add the replicaBackend field for each backend.

```
{
  "backends":[
  {
    ...
    "name": "replica1",
    "replicaBackend": "replica2"
  },
  {
    ...
    "name": "replica2",
    "replicaBackend": "replica1"
  }
  ]
}
```

NOTE

- replicaBackend indicates the peer backend name of two backends forming the replication relationship. As above example shows, replica1's peer backend is replica2, and replica2's peer backend is replica1.

In **parameters** in the **StorageClass** yaml file, add field as below.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: "***"
provisioner: "csi.huawei.com"
parameters:
  ...
  volumetype: lun
  replication: "true"
  replicationSyncPeriod: 3600
  backend: "***"
```

Table 7-5 Parameter description

Parameter	Description	Remarks
replication	Indicates whether a replication volume is created.	"true" denotes creating a replication volume; Not configuring this parameter or "false" denotes not creating a replication volume.
replicationSyncPeriod	Indicates the interval of replication synchronization	This parameter is optional. Unit is second. Default value is 3600s.
backend	Indicates the primary backend name where replication volume will be created	Corresponding to the backend name configured in huawei-csi-configmap.yaml.

Volumes created using this StorageClass are volumes with the SAN replication capability.

7.5.4 Configuring SAN HyperMetro

NOTICE

To use SAN HyperMetro, finish the HyperMetro relationship configuration between two Huawei storage in advance. Please refer to relevant Huawei storage instruction for specifics.

In the **backends** section in the **huawei-csi-configmap.yaml** file, add two backends that form the HyperMetro relationship and add the **hyperMetroDomain** field for each backend.

```
{
  "backends":[
  {
    ...
    "name": "hyperMetro1",
    "hyperMetroDomain": "***",
    "metroBackend": "hyperMetro2"
  },
  {
    ...
    "name": "hyperMetro2",
    "hyperMetroDomain": "***",
    "metroBackend": "hyperMetro1"
  }
  ]
}
```

NOTE

- **hyperMetroDomain** indicates the HyperMetro domain name configured between Huawei storage systems.
- **metroBackend** indicates the peer backend name of two backends forming the HyperMetro relationship. As above example shows, hyperMetro1's peer backend is hyperMetro2, and hyperMetro2's peer backend is hyperMetro1.

In **parameters** in the **StorageClass** yaml file, add field as below.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: "***"
provisioner: "csi.huawei.com"
parameters:
  ...
  volumetype: lun
  hyperMetro: "true"
```

Table 7-6 Parameter description

Parameter	Description	Remarks
hyperMetro	Indicates whether a HyperMetro volume is created.	"true" denotes creating a HyperMetro volume; Not configuring this parameter or "false" denotes not creating a HyperMetro volume.

Volumes created using this StorageClass are volumes with the SAN HyperMetro capability.

7.5.5 Configuring NAS Replication

NOTICE

To use NAS replication, finish the replication relationship configuration between two Huawei storage in advance. Please refer to relevant Huawei storage instruction for specifics.

In the **backends** section in the **huawei-csi-configmap.yaml** file, add two backends that form the replication relationship and add the replicaBackend field for each backend.

```
{
  "backends":[
  {
    ...
    "name": "replica1",
    "replicaBackend": "replica2"
  },
  {
    ...
    "name": "replica2",
    "replicaBackend": "replica1"
  }
  ]
}
```

NOTE

- replicaBackend indicates the peer backend name of two backends forming the replication relationship. As above example shows, replica1's peer backend is replica2, and replica2's peer backend is replica1.
- vStores are supported. Chapter 5.3.2 describes how to configure vStores.

In **parameters** in the **StorageClass** yaml file, add field as below.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: "***"
provisioner: "csi.huawei.com"
parameters:
  ...
  volumetype: fs
  replication: "true"
  replicationSyncPeriod: 3600
  backend: "***"
```


Table 7-7 Parameter description

Parameter	Description	Remarks
replication	Indicates whether a replication volume is created.	"true" denotes creating a replication volume; Not configuring this parameter or "false" denotes not creating a replication volume.
replicationSyncPeriod	Indicates the interval of replication synchronization	This parameter is optional. Unit is second. Default value is 3600s.
backend	Indicates the primary backend name where replication volume will be created	This parameter is conditional mandatory. At the circumstance of using vStore, if vStore belongs to a replication vStore pair, the replication volume only can be created at the primary end of vStore pair, so it is necessary to specify the primary backend where to create the volume, corresponding to the backend name configured in huawei-csi-configmap.yaml.

Volumes created using this StorageClass are volumes with the NAS replication capability.

7.5.6 Configuring NAS HyperMetro

NOTICE

To use NAS HyperMetro, finish the HyperMetro relationship configuration between two Huawei storage in advance, and create HyperMetro vStore pair. Please refer to relevant Huawei storage instruction for specifics.

In the **backends** section in the **huawei-csi-configmap.yaml** file, add two backends that form the HyperMetro relationship and add the **metrovStorePairID** field for each backend.

```
{
  "backends":[
  {
    ...
    "name": "hyperMetro1",
    "vstoreName": "***",
    "metrovStorePairID": "***",
    "metroBackend": "hyperMetro2"
  },
  {
    ...
```



```
    "name": "hyperMetro2",
    "vstoreName": "***",
    "metrovStorePairID": "***",
    "metroBackend": "hyperMetro1"
  }
  ]
}
```

NOTE

- NAS HyperMetro only supports vStore.
- metrovStorePairID indicates the HyperMetro vStore pair ID which vStores belong.
- **metroBackend** indicates the peer backend name of two backends forming the HyperMetro relationship. As above example shows, hyperMetro1's peer backend is hyperMetro2, and hyperMetro2's peer backend is hyperMetro1.

In **parameters** in the **StorageClass** yaml file, add field as below.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: "***"
provisioner: "csi.huawei.com"
parameters:
  ...
  volumetype: fs
  hyperMetro: "true"
```

Table 7-8 Parameter description

Parameter	Description	Remarks
hyperMetro	Indicates whether a HyperMetro volume is created.	"true" denotes creating a HyperMetro volume; Not configuring this parameter or "false" denotes not creating a HyperMetro volume.

Volumes created using this StorageClass are volumes with the NAS HyperMetro capability.

7.6 Advanced Features of Distributed Storage

7.6.1 Configuring QoS

This section describes how to create a LUN volume that supports QoS.

Step 1 Configure the **StorageClass** yaml file. Add the **qos** configuration item under the **parameters** configuration item. The following is an example:

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: "***"
provisioner: "csi.huawei.com"
parameters:
  ...
  qos: '{"maxMBPS": 999, "maxIOPS": 999}'
```


The value of the **qos** configuration item is a JSON character string in dictionary format. A character string is enclosed by single quotation marks and the dictionary key by double quotation marks.

Step 2 Use the StorageClass to create a PVC.

----End

Table 7-9 qos parameters

Parameter	Description	Remarks
maxMBPS	Maximum bandwidth.	Mandatory. The value is a positive integer, expressed in MB/s.
maxIOPS	Maximum IOPS	Mandatory. The value is a positive integer.

8 FAQ

The huawei-csi log directory is **/var/log/huawei**, the csi-controller log file is **huawei-csi-controller**, and the csi-node log file is **huawei-csi-node**. Set the secret log file to **huawei-csi-install**. You can run the following command to query the nodes where the controller and node reside:

```
# kubectl get pod -A -o wide | grep huawei
kube-system   huawei-csi-controller-695b84b4d8-tg64l   3/3     Running     0          14s   <host1-ip>
<host1-name>   <none>           <none>
kube-system   huawei-csi-node-g6f7z                    2/2     Running     0          14s   <host2-ip>   <host2-
name>   <none>           <none>
```


8.1 Viewing Log Information

The run log directory of huawei-csi is **/var/log/huawei**, the log file of csi-controller is **huawei-csi-controller**, the log file of csi-node is **huawei-csi-node**, and the log file for configuring **secret** is **huawei-csi-install**. You can run the following command to query the nodes where the csi-controller and csi-node reside.

```
# kubectl get pod -A -o wide | grep huawei
kube-system   huawei-csi-controller-695b84b4d8-tg64l   3/3     Running     0          14s   <host1-ip>
<host1-name>   <none>           <none>
kube-system   huawei-csi-node-g6f7z                    2/2     Running     0          14s   <host2-ip>   <host2-
name>   <none>           <none>
```


8.2 Failed to Create Pod Because the iscsi_tcp Service Is Not Started Properly When the Kubernetes Platform Is Set Up for the First Time

When a user creates Pod, error **Cannot connect ISCSI portal *.*.*.*: libkmod: kmod_module_insert_module: could not find module by name='iscsi_tcp'** is reported in **/var/log/huawei-csi-node**. This is because the iscsi_tcp service may be stopped after the Kubernetes platform is set up and the iscsi service is installed. You can run the **lsmod | grep iscsi** command to check whether the iscsi_tcp service is started. If the service is not started, run the **modprobe iscsi_tcp** command to start it.

8.3 Failed to Start huawei-csi-node Daemonset, the error message is "/var/lib/iscsi is not a directory

The huawei-csi-node service cannot be started when the huawei-csi-node DaemonSet is started. This is because the container does not have the **/var/lib/iscsi** directory. You just need to **set type whose hostpath is /var/lib/iscsi in the huawei-csi-node.yaml file to "" or delete the type line.**

8.4 After a Worker Node in the Cluster Breaks Down and Recovers, Pod Failover Is Complete but the Source Host Where the Pod Resides Has Residual Drive Letters

Symptom

A Pod runs on worker node A, and an external block device is mounted to the Pod by using CSI. Worker node A is powered off unexpectedly. After detecting the node fault, Kubernetes 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 Configurations

Kubernetes version: 1.13 or later

Storage type: block storage

Root Cause

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:

Troubleshooting method:

Step 1 Run the **multipath -ll** command to check whether a dm device with abnormal multipathing status exists.

As shown in the following figure, if the path status is **failed faulty running**, the corresponding dm device is **dm-12**, and the associated SCSI disk is **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 2**.
- If no, no further action is required.

Step 2 Check whether the residual dm device is readable.

Run the **dd if=/dev/***dm-xx* **of=/dev/null count=1 bs=10M iflag=direct** command.

(*dm-xx* indicates the device number obtained in step 2.)

If the returned result is **Input/output error** and the read data is **0 bytes (0 B) copied**, the device is unreadable.

```
#dd if=/dev/dm-12 of=/dev/null count=1 bs=10M iflag=direct
dd: error reading '/dev/dm-12': Input/output error
0+0 records in
0+0 records out
0 bytes (0 B) copied, 0.0236862 s, 0.0 kB/s
```

- If yes, record the residual *dm-xx* device and associated disk ID (for details, see **Step 1**) and perform the clearing operation.
- If no, contact technical support.

----End

Clearing method:

Step 1 Run the **multipath -f /dev/***dm-xx* command to delete residual multipath aggregation device information according to the dm device obtained in the troubleshooting method.

```
# multipath -f /dev/dm-12
```

If an error is reported, contact technical support.

Step 2 Run the following command to clear the residual SCSI disk according to the drive letter of the residual disk obtained in the troubleshooting method.

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.

Step 3 Check that the dm device and SCSI disk information has been cleared.

Run the **multipath -ll**, **ls -l /sys/block/**, and **ls -l /dev/disk/by-id/** commands in sequence to query the path and disk information. If the residual **dm-12** device and scsi disks **sdi** and **sdj** are cleared, the clearing is complete.

```
# multipath -ll
mpathb (3618cf24100f8f457014a764c000001f6) dm-3 HUAWEI  ,XSG1
size=100G features='0' hwhandler='0' wp=rw
`-+- policy='service-time 0' prio=-1 status=active
  |- 39:0:0:1        sdd 8:48  active ready running
  `- 38:0:0:1        sde 8:64  active ready running
mpathn (3618cf24100f8f457315a764c000001f6) dm-5 HUAWEI  ,XSG1
size=100G features='0' hwhandler='0' wp=rw
`-+- policy='service-time 0' prio=-1 status=active
  |- 39:0:0:2        sdc 8:32  active ready running
  `- 38:0:0:2        sdb 8:16  active ready running
# ls -l /sys/block/
total 0
lrwxrwxrwx 1 root root 0 Aug 11 19:56 dm-0 -> ../devices/virtual/block/dm-0
lrwxrwxrwx 1 root root 0 Aug 11 19:56 dm-1 -> ../devices/virtual/block/dm-1
lrwxrwxrwx 1 root root 0 Aug 11 19:56 dm-2 -> ../devices/virtual/block/dm-2
lrwxrwxrwx 1 root root 0 Aug 11 19:56 dm-3 -> ../devices/virtual/block/dm-3
lrwxrwxrwx 1 root root 0 Aug 11 19:56 sdb -> ../devices/platform/host35/session2/target35:0:0/35:0:0:1/
block/sdb
lrwxrwxrwx 1 root root 0 Aug 11 19:56 sdc -> ../devices/platform/host34/
target34:65535:5692/34:65535:5692:0/block/sdc
lrwxrwxrwx 1 root root 0 Aug 11 19:56 sdd -> ../devices/platform/host39/session6/target39:0:0/39:0:0:1/
block/sdd
lrwxrwxrwx 1 root root 0 Aug 11 19:56 sde -> ../devices/platform/host38/session5/target38:0:0/38:0:0:1/
block/sde
lrwxrwxrwx 1 root root 0 Aug 11 19:56 sdh -> ../devices/platform/host39/session6/target39:0:0/39:0:0:3/
block/sdh
lrwxrwxrwx 1 root root 0 Aug 11 19:56 sdi -> ../devices/platform/host38/session5/target38:0:0/38:0:0:3/
block/sdi
ls -l /dev/disk/by-id/
total 0
lrwxrwxrwx 1 root root 10 Aug 11 19:57 dm-name-mpathb -> ../../dm-3
lrwxrwxrwx 1 root root 10 Aug 11 19:58 dm-name-mpathn -> ../../dm-5
lrwxrwxrwx 1 root root 10 Aug 11 19:57 dm-uuid-mpath-3618cf24100f8f457014a764c000001f6 -> ../../dm-3
lrwxrwxrwx 1 root root 10 Aug 11 19:58 dm-uuid-mpath-3618cf24100f8f457315a764c000001f6 -> ../../dm-5
lrwxrwxrwx 1 root root  9 Aug 11 19:57 scsi-3618cf24100f8f457014a764c000001f6 -> ../../sdd
lrwxrwxrwx 1 root root  9 Aug 11 19:57 scsi-3618cf24100f8f45712345678000103e8 -> ../../sdi
lrwxrwxrwx 1 root root  9 Aug  3 15:17 scsi-3648435a100588052786543211fffffff -> ../../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-0x648435a100588052786543211fffffff -> ../../sdb
lrwxrwxrwx 1 root root  9 Aug  2 14:49 wwn-0x68886030000020aff44cc0d060c987f1 -> ../../sdc
```

----End

