HITACHI

Hitachi Vantara

Hitachi Storage Plug-in for Containers

Version 3.17.4

Quick Reference Guide

This Quick Reference Guide provides an implementation overview and describes the usage requirements, installation, and configuration of Storage Plug-in for Containers.

MK-92ADPTR142-28
February 2026

Contents

Preface

Hitachi Storage Plug-in for Containers lets you create containers and run stateful applications inside those containers by using the Hitachi storage volumes as dynamically provisioned persistent volumes. This Quick Reference Guide provides an implementation overview and describes the usage requirements, installation, and configuration of Storage Plug-in for Containers.

Product version

This document applies to Hitachi Storage Plug-in for Containers version 3.17.4.

Release notes

Read the release notes before installing and using this product. They may contain requirements or restrictions that are not fully described in this document or updates or corrections to this document. Release notes are available on the Hitachi Vantara documentation website: https://docs.hitachivantara.com.

Conventions for capacity values

Logical capacity units (for example, logical device capacity, cache memory capacity) are calculated based on the values that are outlined in the following table.

Logical capacity unit	Value
1 KiB	1,024 (2^{10}) bytes
1 MiB	1,024 KiB or $1{,}024^2$ bytes
1 GiB	1,024 MiB or $1{,}024^3$ bytes
1 TiB	1,024 GiB or $1{,}024^4$ bytes
1 PiB	1,024 TiB or $1{,}024^5$ bytes
1 EiB	1,024 PiB or $1{,}024^6$ bytes

Storage model abbreviations

This document uses the following abbreviations for storage models.

Abbreviation	Full name
VSP One Block	Hitachi Virtual Storage Platform One Block Collective name for the following storage models: ▪ Hitachi Virtual Storage Platform One Block 24 ▪ Hitachi Virtual Storage Platform One Block 26 ▪ Hitachi Virtual Storage Platform One Block 28 ▪ Hitachi Virtual Storage Platform One Block 85
VSP One Block 20 series	Hitachi Virtual Storage Platform One Block 20 series Collective name for the following storage models: ▪ Hitachi Virtual Storage Platform One Block 24 ▪ Hitachi Virtual Storage Platform One Block 26 ▪ Hitachi Virtual Storage Platform One Block 28
VSP One Block High End	Hitachi Virtual Storage Platform One Block High End
VSP One SDS Block	Hitachi Virtual Storage Platform One SDS Block
VSP family	Hitachi Virtual Storage Platform family Collective name for the following storage models: ▪ VSP E series ▪ VSP F350, F370, F700, F900 ▪ VSP G350, G370, G700, G900 ▪ VSP 5000 series
VSP E series	Hitachi Virtual Storage Platform E series Collective name for the following storage models: ▪ Hitachi Virtual Storage Platform E590 ▪ Hitachi Virtual Storage Platform E790 ▪ Hitachi Virtual Storage Platform E990 ▪ Hitachi Virtual Storage Platform E1090 ▪ Hitachi Virtual Storage Platform E590H

Abbreviation	Full name
	▪ Hitachi Virtual Storage Platform E790H ▪ Hitachi Virtual Storage Platform E1090H
VSP F350	Hitachi Virtual Storage Platform F350
VSP F370	Hitachi Virtual Storage Platform F370
VSP F700	Hitachi Virtual Storage Platform F700
VSP F900	Hitachi Virtual Storage Platform F900
VSP G350	Hitachi Virtual Storage Platform G350
VSP G370	Hitachi Virtual Storage Platform G370
VSP G700	Hitachi Virtual Storage Platform G700
VSP G900	Hitachi Virtual Storage Platform G900
VSP 5000 series	Hitachi Virtual Storage Platform 5000 series Collective name for the following storage models: ▪ Hitachi Virtual Storage Platform 5100 ▪ Hitachi Virtual Storage Platform 5200 ▪ Hitachi Virtual Storage Platform 5500 ▪ Hitachi Virtual Storage Platform 5600 ▪ Hitachi Virtual Storage Platform 5100H ▪ Hitachi Virtual Storage Platform 5200H ▪ Hitachi Virtual Storage Platform 5500H ▪ Hitachi Virtual Storage Platform 5600H

Accessing product documentation

Product user documentation is available on: https://docs.hitachivantara.com. Check this site for the most current documentation, including important updates that may have been made after the release of the product.

Getting help

The Hitachi Vantara Support Website is the destination for technical support of products and solutions sold by Hitachi Vantara. To contact technical support, log on to the Hitachi Vantara Support Website. For additional contact methods, go to https://support.hitachivantara.com/en/contact-support.html.

Hitachi Vantara Community is a global online community for Hitachi Vantara customers, partners, independent software vendors, employees, and prospects. It is the destination to get answers, discover insights, and make connections. **Join the conversation today!** Go to community.hitachivantara.com, register, and complete your profile.

Comments

Please send comments to doc.feedback@hitachivantara.com. Include the document title and number, including the revision level (for example, -07), and refer to specific sections and paragraphs whenever possible. All comments become the property of Hitachi Vantara LLC.

Thank you!

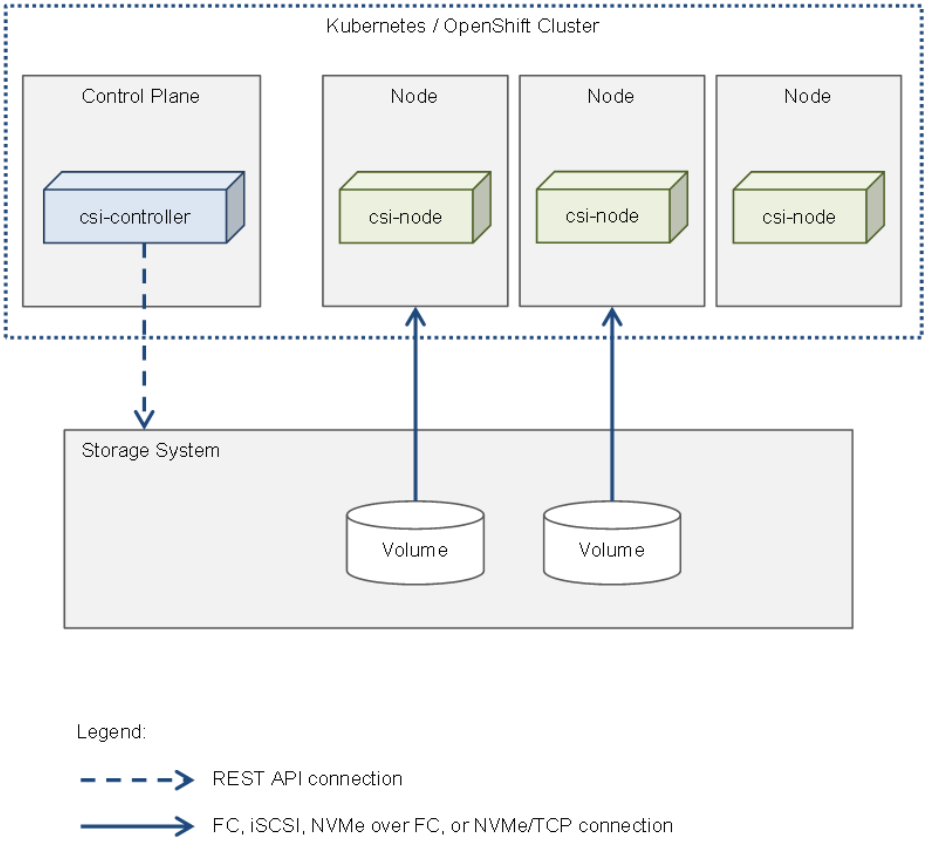
Chapter 1: Overview

Storage Plug-in for Containers is software for creating and managing persistent volumes for Hitachi storage systems in a Kubernetes environment. By using Storage Plug-in for Containers, you can use Hitachi storage volumes from stateful applications running on Kubernetes. For persistent volumes, in addition to creation and deletion, operations such as snapshots and clones are supported.

About Hitachi Storage Plug-in for Containers

Storage Plug-in for Containers integrates Kubernetes or OpenShift with Hitachi storage using Container Storage Interface (CSI).

The following diagram illustrates a container environment where Storage Plug-in for Containers is deployed.



The following table lists and describes the components of Storage Plug-in for Containers.

Component	Purpose
csi-controller	Implements the CSI controller service, which mainly uses the REST API for storage operations. This is deployed as Deployment and starts on a control plane. If the csi-controller cannot start on a control plane, it might start on a node.
csi-node	Implements the CSI node service, which primarily manages volumes on each node. This is deployed as DaemonSet, and all nodes must have this component.
Hitachi storage systems	Provides storage volumes for the containers.

Prerequisites

Before you install Storage Plug-in for Containers, verify that your system meets the minimum requirements.

Container orchestrators to be supported

Container orchestrator	Supported versions	Remarks
Red Hat OpenShift Container Platform	4.18, 4.19, and 4.20	—
Kubernetes	1.30, 1.31, 1.32, and 1.33	—
Rancher Kubernetes Engine 2 (RKE2)	1.30, 1.31, 1.32, and 1.33	If you use RKE2, any mention of Kubernetes in this guide must be referred to as RKE2.
Amazon Elastic Kubernetes Service (EKS)	1.30, 1.31, 1.32, and 1.33	If you use EKS, any mention of Kubernetes in this guide must be referred to as EKS.

For details on supported versions, see the Release Notes.

Server requirements

Component	Requirement
CPU	x86_64
Operating system	Refer to the release notes for details.
Interface	Fibre Channel, iSCSI, NVMe over FC, and NVMe/TCP for bare metal servers iSCSI and NVMe/TCP for virtual machines

Storage requirements

The following are the storage requirements for the supported storage systems:

Component	Requirement
Model and SVOS version	Refer to the release notes for details.
Interface	Fibre Channel, iSCSI, and NVMe over FC **Note:** NVMe/TCP is not supported.
Host group	Storage Plug-in for Containerscreates a host using predefined naming rules. If you want to create a host group manually, you must follow the same naming rules. **Note:** Do not use the host group created by the container plug-in for any purpose other than the Storage Plug-in for Containers.
User account	The built-in Storage administrator (View & Modify) user group. If you are using a customized user group, make sure it has the same roles as the built-in Storage Administrator (View & Modify) user group.
License	The following licenses are required: ▪ Dynamic Provisioning (DP) ▪ Hitachi Thin Image (HTI) ▪ Hitachi Thin Image Advanced (HTIA) for VSP One Block storage systems.

Component	Requirement
SVP	Single and dual SVP configurations are supported. **Note:** SVP is not required for VSP One Block storage systems.

Storage requirements for VSP One SDS Block.

Component	Requirement
Version	Refer to the release notes for details.
Interface	Fibre Channel, iSCSI, and NVMe/TCP NVMe/TCP is supported only for VSP One SDS Block 01.16.00.4x.
User account	▪ If multitenancy functionality is not used: The user must be assigned the Storage role. ▪ If multitenancy functionality is used: See Multitenancy functionality settings (on page 23) and set a user.

Network requirements

Storage Plug-in for Containers uses port 80 or 443 for REST API connection. Use this information when configuring the firewall.

Note: Storage Plug-in for Containers does not support IPv6. Use IPv4.

Pre-installation tasks

Before you install Storage Plug-in for Containers, review and apply the server and storage pre-installation requirements.

Setting up the environment

Storage Plug-in for Containers enables dynamic operation of storage systems when containers are used. Set up the environment to use Storage Plug-in for Containers.

- Check and apply the requirements for the server (where you plan to install Storage Plug-in for Containers, Kubernetes, and OpenShift), Hitachi storage systems, Kubernetes, OpenShift, Rancher Kubernetes Engine 2, and Amazon EKS.
- Set up the Kubernetes or the OpenShift environment.
- Configure the supported storage systems using the appropriate protocol to enable communication with Kubernetes or Open Shift.

Server pre-installation

The following table outlines the pre-installation tasks for each server component.

If you are using VSP One SDS Block, for information on the server settings, see the VSP One SDS Block manual.

Component	Tasks
Hypervisor	If you want to use virtual machines, set up the hypervisor. **Note:** Storage Plug-in for Containers is tested with VMware vSphere 7.0/8.0.
Fibre Channel	Verify that HBA is installed on nodes that implement a Fibre Channel connection with the storage system.
iSCSI	Verify that iSCSI initiator software is installed on nodes that implement an iSCSI connection with the storage system. If the software is not installed, refer to: https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/9/html/managing_storage_devices/configuring-an-iscsi-initiator_managing-storage-devices#creating-an-iscsi-initiator_configuring-an-iscsi-initiator **Note:** Storage Plug-in for Containers does not support IQNs that include uppercase alphabetic characters.

Component	Tasks
NVMe over FC	NVMe over FC connections are supported by the following OSs. Check the following before specifying settings: ▪ Ubuntu 20.04, 22.04, 24.04 ▪ Red Hat Enterprise Linux 9 ▪ Red Hat Enterprise Linux CoreOS (OpenShift) Specify the following settings for a node that connects with the storage system through an NVMe over FC connection. The method for specifying settings differs depending on the OS. ▪ Ubuntu Using the following command, install the `nvme-cli` tool: `apt-get install nvme-cli` ▪ Red Hat Enterprise Linux • Broadcom (Emulex) Using the following website as reference, install the `nvme-cli` tool: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/9/html/managing_storage_devices/configuring-nvme-over-fabrics-using-nvme-fc_managing-storage-devices • QLogic Using the following website as reference, install the `nvme-cli` tool and then reload the QLogic module (qla2xxx): https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/9/html/managing_storage_devices/configuring-nvme-over-fabrics-using-nvme-fc_managing-storage-devices ▪ Red Hat Enterprise Linux CoreOS You do not need to install the `nvme-cli` tool.

Component	Tasks
	Note: ▪ Do not change a host NQN while the host is running. However, if a change while the host is running is necessary, perform the drain operation on the relevant node, and then change the host NQN. After changing the host NQN, restart the host. ▪ When there are multiple nodes, make sure that no duplicate host NQN exists.
NVMe/TCP	NVMe/TCP connections are supported by the following OSs. Check the following before specifying settings: ▪ Red Hat Enterprise Linux 9 ▪ Red Hat Enterprise Linux CoreOS (OpenShift) Specify the following settings for a node that connects with the storage system through an NVMe/TCP connection. The method for specifying settings differs depending on the OS. ▪ Red Hat Enterprise Linux Using the following website as reference, install the `nvme-cli` tool. After installation, if the `nvme-tcp` module is not loaded, load it. https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/9/html/managing_storage_devices/configuring-nvme-over-fabrics-using-nvme-tcp_managing-storage-devices ▪ Red Hat Enterprise Linux CoreOS You do not need to install the `nvme-cli` tool.

Component	Tasks
	Note: ▪ Do not change a host NQN while the host is running. However, if a change while the host is running is necessary, perform the drain operation on the relevant node, and then change the host NQN. After changing the host NQN, restart the host. ▪ When there are multiple nodes, make sure that no duplicate host NQN exists.
Multipath function	For Fibre Channel and iSCSI, use Device Mapper Multipath. For NVMe over FC and NVMe/TCP, use Native NVMe Multipath. For details on multipath function settings, see Device Mapper Multipath settings (on page 17) or Native NVMe Multipath settings (on page 19).

Device Mapper Multipath settings

Enable Device Mapper Multipath and make sure that the `user_friendly_names` option is set to `yes`.

For example:

```
defaults {
        user_friendly_names yes
        find_multipaths yes
}
blacklist {
}
```

If you are using VSP One SDS Block, for information on settings specific to VSP One SDS Block, see the VSP One SDS Block manual, and check the sections describing the operating environment settings and the ALUA settings.

Note: The setting values might differ depending on the environment. Also see the documentation for your OS.

- Red Hat Enterprise Linux 7: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/dm_multipath/mpio_setup
- Red Hat Enterprise Linux 8: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/configuring_device_mapper_multipath/configuring-dm-multipath_configuring-device-mapper-multipath
- Red Hat Enterprise Linux 9: https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/9/html/configuring_device_mapper_multipath/configuring-dm-multipath_configuring-device-mapper-multipath
- Ubuntu: https://ubuntu.com/server/docs/device-mapper-multipathing-introduction

For OpenShift, you will need to use the MachineConfig YAML file. For details, see the official documentation: https://docs.openshift.com/container-platform/latest/machine_configuration/index.html

For OpenShift, the following is an example of the procedure:

Procedure

1. Obtain `multipath-machineconfig-sample.yaml` from the provided sample files.

   Note: You can obtain the sample files from https://github.com/hitachi-vantara/csi-operator-hitachi. The sample files are stored in the `csi-operator-hitachi/hspc/<Storage-Plug-in-for-Containers-version>/sample` directory.

2. If necessary, change the multipath settings in `multipath-machineconfig-sample.yaml`.

   The following default multipath settings are specified in `multipath-sample.conf`. The character string obtained by encoding this file in base64 is specified in `multipath-machineconfig-sample.yaml`.

   ```
   defaults {
   user_friendly_names yes
   find_multipaths yes
   }
   blacklist {
   }
   ```

   a. Obtain `multipath-sample.conf` from the provided sample files.
   b. Edit `multipath-sample.conf` to change the multipath settings.
   c. Run the following command to obtain `multipath-sample.conf` encoded in base64:

   ```
   # cat multipath-sample.conf | base64 -w0
   ```

d. Change the `spec.config.storage.files.contents.source` setting in `multipath-machineconfig-sample.yaml`.

The character string specified for `spec.config.storage.files.contents.source` corresponds to the multipath settings encoded in base64. Replace this character string with the base64-encoded character string obtained from `multipath-sample.conf`.

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  name: multipath-machineconfig-sample
  labels:
    machineconfiguration.openshift.io/role: worker
spec:
  config:
    ignition:
      version: 3.2.0
    storage:
      files:
      - contents:
          source: data:text/plain;charset=utf-8;base64,
ZGVmYXVsdHMgewp1c2VyX2ZyaWVuZGx5X25hbWVzIHllcwpmaW5kX211bHRpcGF0aHMgeWVzCn0K
YmxhY2tsaXN0IHsKfQo=
          verification: {}
        filesystem: root
        mode: 400
        path: /etc/multipath.conf
```

3. Run the following command:

```
# oc apply -f multipath-machineconfig-sample.yaml
```

Note: MachineConfig applies to compute nodes only. After MachineConfig is created, all compute nodes are automatically restarted one by one, and `/etc/multipath.conf` is created on all compute nodes.

4. On each compute node, open `/etc/multipath.conf` and verify that the settings have been applied.

Note: It might take time for the settings to be applied.

Native NVMe Multipath settings

The method for specifying Native NVMe Multipath settings differs depending on the OS.

Ubuntu

By default, Native NVMe Multipath is enabled. If Native NVMe Multipath is disabled, enable it.

Red Hat Enterprise Linux

Using the following website as reference, enable Native NVMe Multipath:

https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/9/html/managing_storage_devices/enabling-multipathing-on-nvme-devices_managing-storage-devices

Red Hat Enterprise Linux CoreOS

By default, Native NVMe Multipath is enabled. If Native NVMe Multipath is disabled, enable it.

If you are using VSP One SDS Block, for information on settings specific to VSP One SDS Block, see the VSP One SDS Block manual, and check the sections describing the operating environment settings.

Storage pre-installation for VSP family and VSP One Block

The following table outlines the pre-installation tasks to be completed for each storage component.

Component	Task
Program products	▪ Enable Dynamic Provisioning (DP) license ▪ Enable Hitachi Thin Image (HTI) license ▪ Enable Hitachi Thin Image Advanced (HTIA) license for VSP One Block storage systems.
Pool	Create an DP pool. Dynamic Tiering is not supported. For VSP One Block storage systems, a DP pool configured with flash media (SSD) is required. For VSP One Block storage systems, the following pools are not supported: ▪ Pools different from the primary volume ▪ Pools containing external volume pool volumes ▪ Pools set with Data Direct Map attribute
Fibre Channel connection	Use a Fibre Channel switch for communication between the storage and servers. Set the following parameters for storage ports using Element Manager for Block 20: ▪ Connection Type: P-to-P ▪ Fabric: ON ▪ Security: Enabled

Component	Task
	Storage Plug-in for Containers automatically performs the following actions: ▪ Creates host groups for each host if there is no host group. If you want to use existing host groups, rename them according to the naming rule (see Host group and iSCSI target naming rules (on page 22)). ▪ Adds the WWNs for all of the HBA ports in each host to the host group created for each host. **Note:** Storage Plug-in for Containers will overwrite host mode options even if existing host groups have other host mode options.
iSCSI connection	Enable port security by Storage Navigator. Storage Plug-in for Containers automatically performs the following actions: ▪ Creates iSCSI targets for each host if there is no iSCSI target. If you want to use existing iSCSI targets, rename them according to the naming rule (see Host group and iSCSI target naming rules (on page 22)). ▪ Adds the IQN to the iSCSI target corresponding to each host that will join the Kubernetes cluster. ▪ Logs in to the iSCSI target on each host. If you want to use CHAP, do the following: ▪ Create an iSCSI target (see Host group and iSCSI target naming rules (on page 22)). ▪ Set CHAP for the port and iSCSI target. ▪ Log in to the iSCSI target with CHAP authentication. Run login from each host. **Note:** Storage Plug-in for Containers will overwrite host mode options even if existing iSCSI targets have other host mode options.
NVMe over FC connection	Create an NVM subsystem to manage the path between the host and storage system. The NVM subsystem must be dedicated to Storage Plug-in for Containers. Do not use the NVM subsystem used for Storage Plug-in for Containers for any purpose other than Storage Plug-in for Containers.

Component	Task
	1. Create the NVM subsystem for the storage system. Enable the namespace security, and specify Linux as the host mode. **2.** Set the operation mode of the Fibre Channel port to NVMe mode. **3.** Disable LUN security on the Fibre Channel port. **4.** Set an NVM subsystem port. For details about each of the above steps, see the *Provisioning Guide for Open Systems* or *Provisioning Guide*.

Host group and iSCSI target naming rules

Storage Plug-in for Containers automatically searches host groups and iSCSI targets based on the name.

If you want to use an already existing host group or iSCSI target, refer to either the naming rule of host groups or iSCSI targets depending on your storage connection:

Naming rule of host groups

Storage Plug-in for Containers searches host groups by the naming rule. If Storage Plug-in for Containers cannot find any host group in the port, it automatically creates the host group. If you already have host groups, you need to delete them or rename them according to the following naming rule:

"spc-<*wwn1*>-<*wwn2*>-<*wwn3*>"

Naming rule details:

- <*wwn1*>, <*wwn2*>, <*wwn3*> are the world wide name of each host.
- <*wwn1*>, <*wwn2*>, <*wwn3*> are sorted by name.
- If the host has more than three WWNs, Storage Plug-in for Containers sorts <*wwn1*>, <*wwn2*> ... <*wwnN*> and uses lower three names.
- If the host has only one or two WWNs, the names are "spc-<*wwn1*>" or "spc-<*wwn1*>-<*wwn2*>".

Naming rule of iSCSI targets

Storage Plug-in for Containers searches iSCSI targets by the naming rule. If Storage Plug-in for Containers cannot find any iSCSI target, it automatically creates the iSCSI target, "spc-<*hashed-IQN*>". If you already have iSCSI targets, you need to delete them or rename them according to the following naming rule: "spc-<*any-string*>"

Storage pre-installation for VSP One SDS Block

Component	Task
Fibre Channel connection	Use the Fibre Channel switch for communication between storage and servers.
Server resource	If you have already created a Server resource in VSP One SDS Block, verify the following: Storage Plug-in for Containers automatically performs the following actions: ▪ Finds an existing Server resource with the host WWN, IQN, or NQN configured. ▪ If an existing Server resource is not found, it creates a new Server resource and configures it with a host WWN, IQN, or NQN. ▪ Verifies and configures the Server resource to connect to the all compute ports. If you want to use CHAP, do the following: ▪ Set CHAP for the compute port. ▪ Log in to the iSCSI target with CHAP authentication. Run login from each host. **Note:** Do not add a WWN, IQN, or NQN of multiple hosts to the same Server resource. Only one Server resource can be associated with a single host.

Multitenancy functionality settings

If you use the multitenancy functionality, set the items in the following table.

Note: An environment in which one Kubernetes/OpenShift cluster and one VSP One SDS Block are used has the following restrictions related to Virtual Private Storage (VPS):

- Multiple VPSs cannot be used.
- VPS and resources that do not belong to the VPS cannot be used at the same time.

Component	Task
VPS	Create VPS. For the maximum number of compute nodes set for VPS, set a number greater than the number of nodes that make up the Kubernetes/OpenShift cluster. For the maximum number of sessions, set at least 20.
User group	Create a user group that belongs to the created VPS. For the scope, set only the created VPS. For the role, set VpsStorage.
User	Create users and assign them to the user group you created in the preceding task. Do not assign the users to a user group other than the user group you created in the preceding task.

Chapter 2: Installation

This chapter describes how to install Storage Plug-in for Containers. The installation method depends on whether your environment is OpenShift or Kubernetes.

Installation on OpenShift

Storage Plug-in for Containers is easily deployed to OpenShift using the Operator, which can be installed from OperatorHub. To install Storage Plug-in for Containers, follow the steps below.

Note:

- If there is a previous version of Storage Plug-in for Containers, remove it before performing the installation procedure.
- If you want to install Storage Plug-in for Containers in an OpenShift Container Platform environment that does not have access to the internet, mirror the certified-operators catalog in advance. For details on the procedure, see https://docs.openshift.com/container-platform/latest/disconnected/mirroring/installing-mirroring-installation-images.html#olm-mirror-catalog_installing-mirroring-installation-images.

  For example, for OpenShift Container Platform version 4.10, the index image of the certified-operators catalog is registry.redhat.io/redhat/certified-operator-index:v4.10. For details, see https://docs.openshift.com/container-platform/latest/operators/understanding/olm-rh-catalogs.html.

Procedure

1. Access OperatorHub from the OpenShift web console.
2. Search Hitachi Storage Plug-in for Containers and install the Operator.

   Note: Select the following settings in Operator Subscription:

   - Installation mode: Select **A specific namespace on the cluster** and specify any namespace.
   - Update approval: Select **Manual** and approve the Install Plan (see https://docs.openshift.com/).
3. Confirm the status of the Operator is **Succeeded**.
4. Confirm the status of the Operator Pod is **Running**.
5. Click **Create Instance** on the Operator Details.
6. Click **Create**. If you want to make an advanced configuration, refer to Configuration of Storage Plug-in for Containers instance (on page 27).

7. Confirm the status READY is **true** using the following command:

```
# oc get hspc -n <Storage-Plug-in-for-Containers-namespace>
NAME    READY    AGE
hspc    true     30s
```

Installation on Kubernetes

For Kubernetes, you can install Storage Plug-in for Containers using Operator. To install Storage Plug-in for Containers, perform the following procedure.

Note: If there is a previous version of Storage Plug-in for Containers, remove it before performing the installation procedure.

Procedure

1. Create a clone of https://github.com/hitachi-vantara/csi-operator-hitachi, and then move to the target Storage Plug-in for Containers version.

```
# git clone https://github.com/hitachi-vantara/csi-operator-hitachi
# cd csi-operator-hitachi/hspc/<Storage-Plug-in-for-Containers-version>/operator
```

2. Create the namespace for the Operator, confirm that the namespace was created succesfully for the Operator:

```
# kubectl create -f hspc-operator-namespace.yaml
# kubectl get namespaces | grep hspc-operator-system
hspc-operator-system   Active   53s
```

Note: Verify that the namespace is used in all plugin configurations and operations.

3. Create the Operator and confirm the Operator is running:

```
# kubectl create -f hspc-operator.yaml

# kubectl get deployment -n hspc-operator-system
NAME                              READY  UP-TO-DATE  AVAILABLE   AGE
hspc-operator-controller-manager  1/1    1           1           14s
```

4. If you want to change the Storage Plug-in for Containers settings, edit `hspc_v1_hspc.yaml`.

   On the namespace specified in `hspc_v1_hspc.yaml`, Storage Plug-in for Containers is created. Change the settings as necessary.

   If you want to make an advanced configuration, refer to Configuration of Storage Plug-in for Containers instance (on page 27).

5. Create a Storage Plug-in for Containers instance, and confirm that `READY` is `true`.

```
# kubectl create -f hspc_v1_hspc.yaml

# kubectl get hspc -n <Storage-Plug-in-for-Containers-namespace>
NAME READY AGE
hspc true 30s
```

For <*Storage-Plug-in-for-Containers-namespace*>, specify the namespace specified in `hspc_v1_hspc.yaml`.

Configuration of Storage Plug-in for Containers instance

You can configure Storage Plug-in for Containers by editing the `CustomResource` YAML file, which includes the following parameters:

Parameter	Description
`spec.imagePullSecrets`	Specify this parameter if a Secret is required to pull an image.
`spec.controller.containers.name`	Name of the Storage Plug-in for Containers that you want to configure in hspc-`csi-controller` pods. For example, `hspc-csi-driver`, `csi-provisioner`, and so on are the key to the container name inside the hspc-`csi-controller`. The **`kubectl describe deployment hspc-csi-controller -n <Storage-Plug-in-for-Containers-namespace>`** command is used to get the container names.
`spec.controller.containers.image`	The image name of `hspc-csi-controller`
`spec.controller.containers.imagePullPolicy`	The image pull policy of `hspc-csi-controller`. The default value is `IfNotPresent`.
`spec.controller.containers.env`	List of environment variables to set in `hspc-csi-controller` container. Refer to Environment variables (on page 30).

Parameter	Description
`spec.controller.containers.args`	Arguments to the entry point for `hspc-csi-controller`. This replaces all parameters at `spec.template.spec.containers.args` in a deployment of the container `hspc-csi-controller`.
`spec.controller.tolerations`	Specify the toleration of the Pod that runs `hspc-csi-controller`. The same format as Kubernetes is to be used. For details, see https://kubernetes.io/docs/concepts/scheduling-eviction/taint-and-toleration/.
`spec.controller.affinity.nodeAffinity`	Specify the node affinity of the Pod that runs `hspc-csi-controller`. The same format as Kubernetes is to be used. For details, see https://kubernetes.io/docs/tasks/configure-pod-container/assign-pods-nodes-using-node-affinity/.
`spec.node.containers.name`	Name of the container that you want to configure in `hspc-csi-node` pods. For example, `hspc-csi-driver`, `liveness-probe`, and so on are the key to the container name inside `hspc-csi-node`. The **`kubectl describe daemonset hspc-csi-node -n <Storage-Plug-in-for-Containers-namespace>`** command is used to get the container names.
`spec.node.containers.image`	The image name of `hspc-csi-node`
`spec.node.containers.imagePullPolicy`	The image pull policy of `hspc-csi-node`. The default value is `IfNotPresent`.
`spec.node.containers.env`	List of environment variables to set in `hspc-csi-node` container.
`spec.node.containers.args`	Arguments to the entry point for `hspc-csi-node`. This replaces all parameters at `spec.template.spec.containers.args` in a deployment of the container `hspc-csi-node`.

Parameter	Description
`spec.node.tolerations`	Specify the toleration of the Pod that runs `hspc-csi-node`. The same format as Kubernetes is to be used. For details, see https://kubernetes.io/docs/concepts/scheduling-eviction/taint-and-toleration/.
`spec.node.affinity.nodeAffinity`	Specify the node affinity of the Pod that runs `hspc-csi-node`. The same format as Kubernetes is to be used. For details, see https://kubernetes.io/docs/tasks/configure-pod-container/assign-pods-nodes-using-node-affinity/.

Multipath configuration using a DaemonSet

You can deploy and manage multipath settings on Kubernetes nodes through a DaemonSet, ensuring reliable and high-performance connections between the node and the storage system.

Before you begin

- Install the current version of Hitachi Storage Plug-in for Containers in your cluster.
- Verify there is access to all the cluster nodes.

Procedure

1. Create a clone of https://github.com/hitachi-vantara/csi-operator-hitachi, and then navigate to the target Storage Plug-in for Containers version.

   ```
   # git clone https://github.com/hitachi-vantara/csi-operator-hitachi
   # cd csi-operator-hitachi/hspc/<version>/sample
   ```

2. Apply the `rosa-daemonset.yaml` file in the namespace where you installed Storage Plug-in for Containers. Enter:

   ```
   # kubectl create -f rosa-daemonset.yaml
   ```

3. Verify that the DaemonSet has been deployed successfully. Enter:

   ```
   #kubectl get daemonset hspc-iscsi-init -n <namespace>
   ```

   Ensure that the DaemonSet has created one Pod on each node in the cluster.

4. Using SSH, log on to the node.

5. Verify the `multipathd` service is active and running. Enter:

```
#systemctl status multipathd
```

6. Verify that the multipath configuration file exists and contains the correct settings. Enter:

```
#cat /etc/multipath.conf
defaults {
    user_friendly_names yes
    find_multipaths yes
        }
blacklist {
    device {
        vendor "<vendor name>"
        product "<product name>"
    }}
```

Note: Repeat steps 3 and 4 on all the nodes in the cluster.

Environment variables

The following is the environment variable of `hspc-csi-driver` on `hspc-csi-controller`.

Environment variable name	Description
`SPC_VERIFY_CERTIFICATE`	If `true`, the TLS certificate of the storage is checked in the HTTPS connection (Default: `false`).
`TZ`	Timezone for logging. For example, `Asia/Tokyo` (Default: `UTC`).

The following is an example of the procedure for enabling certificate verification of the `hspc-csi-driver`.

1. Check the current settings using the following command:

```
# kubectl get deployment -n <Storage-Plug-in-for-Containers-namespace> hspc-csi-controller -o yaml
<…>
    - name: hspc-csi-driver
      env:
      - name: CSI_ENDPOINT
        value: unix:///csi/csi-controller.sock
      - name: KUBE_NODE_NAME
        valueFrom:
```

```
          fieldRef:
            apiVersion: v1
            fieldPath: spec.nodeName
<…>
```

2. Add `env: SPC_VERIFY_CERTIFICATE` to Storage Plug-in for Containers manifests.

```
apiVersion: csi.hitachi.com/v1
kind: HSPC
metadata:
  name: hspc
  namespace: <Storage-Plug-in-for-Containers-namespace>
spec:
  controller:
    containers:
      - name: hspc-csi-driver
        env:
        - name: SPC_VERIFY_CERTIFICATE
          value: "true"
```

3. Uninstall and reinstall Storage Plug-in for Containers. For more information on how to uninstall and reinstall Storage Plug-in for Containers, see Installation (on page 25) and Uninstallation (on page 85).
4. Check the changes.

```
# kubectl get deployment -n <Storage-Plug-in-for-Containers-namespace> hspc-csi-
controller -o yaml
<…>
    - name: hspc-csi-driver
      env:
      - name: CSI_ENDPOINT
        value: unix:///csi/csi-controller.sock
      - name: KUBE_NODE_NAME
        valueFrom:
          fieldRef:
            apiVersion: v1
            fieldPath: spec.nodeName
      - name: SPC_VERIFY_CERTIFICATE
        value: "true"
<…>
```

Chapter 3: Usage

This chapter describes the settings and command examples for each component used in Storage Plug-in for Containers.

The procedures that follow are performed by using sample files.

Note: You can obtain the sample files from https://github.com/hitachi-vantara/csi-operator-hitachi. The sample files are stored in the `csi-operator-hitachi/hspc/<Storage-Plug-in-for-Containers-version>/sample` directory.

Secret settings

The Secret file contains the storage URL, user name, password, and host mode settings that are necessary for Storage Plug-in for Containers to work with your environment. The following sample provides information about the required parameters.

Parameter references for secret-sample.yaml

```
apiVersion: v1
kind: Secret
metadata:
  name: secret-sample              #(1)
type: Opaque
data:
  url: aHR0cDovLzE3Mi4xNi4xLjE=    #(2)
  user: VXNlcjAx                   #(3)
  password: UGFzc3dvcmQwMQ==       #(4)
  hostModeOptions: ODgsODENCg==          #(5)
```

Legend:

(1) Secret name

(2) base64-encoded storage URL

Use the IP address of the SVP for the VSP 5000 series storage systems. .

Use the IP address of the storage controller for the following: VSP E series, VSP F350, F370, F700, F900, and VSP G350, G370, G700, G900.

Use the service IP address for the following: VSP One B20 series and VSP One Block High End.

Example:

```
echo -n "http://172.16.1.1" | base64
```

(3) base64-encoded storage user name.

Example:

```
echo -n "User01" | base64
```

(4) base64-encoded storage password.

Example:

```
echo -n "Password01" | base64
```

(5) base64-encoded host mode options.

Example:

```
echo -n "88,81" | base64
```

Note: By default, the host mode options are set to 2, 22, 25, 68, and 91. If you need to add additional host mode options, you must specify them in the `hostModeOptions` parameter in the `secret.yaml` file. You must delete and recreate the pod to apply the user-defined host mode options.

StorageClass settings

The StorageClass file contains storage settings that are necessary for Storage Plug-in for Containers to work with your environment. The following sample provides information about the required parameters.

Note: After creating a StorageClass and PVC, re-creating StorageClass will not affect the existing PVCs.

StorageClass for VSP family and VSP One Block storage systems

Parameter references for sc-sample.yaml

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-sample                        #(1)
  annotations:
    kubernetes.io/description: Hitachi Storage Plug-in for Containers
provisioner: hspc.csi.hitachi.com
reclaimPolicy: Delete
volumeBindingMode: Immediate
allowVolumeExpansion: true
parameters:
  serialNumber: "54321"                  #(2)
  poolID: "1"                            #(3)
  portID : CL1-A,CL2-A                   #(4)
  connectionType: fc                     #(5)
  storageEfficiency: "CompressionDeduplication"      #(6)
  storageEfficiencyMode: "PostProcess"               #(7)
  csi.storage.k8s.io/fstype: ext4                    #(8)
  csi.storage.k8s.io/node-publish-secret-name: "secret-sample"        #(9)
  csi.storage.k8s.io/node-publish-secret-namespace: "default"         #(10)
  csi.storage.k8s.io/provisioner-secret-name: "secret-sample"         #(9)
  csi.storage.k8s.io/provisioner-secret-namespace: "default"          #(10)
  csi.storage.k8s.io/controller-publish-secret-name: "secret-sample"  #(9)
  csi.storage.k8s.io/controller-publish-secret-namespace: "default"   #(10)
  csi.storage.k8s.io/node-stage-secret-name: "secret-sample"          #(9)
  csi.storage.k8s.io/node-stage-secret-namespace: "default"           #(10)
  csi.storage.k8s.io/controller-expand-secret-name: "secret-sample"   #(9)
  csi.storage.k8s.io/controller-expand-secret-namespace: "default"    #(10)
```

Legend:

(1) StorageClass name

(2) Storage serial number

(3) HDP pool ID

(4) Port ID. Use a comma separator for multipath. If an NVMe over FC connection is used, this specification is unnecessary.

(5) Connection type between storage and nodes. `fc`, `iscsi`, and `nvme-fc` are supported. If `connectionType` is not specified, `fc` is set.

Note: If an NVMe over FC connection is used, add `nvmSubsystemID`, and then specify a value.

(6) Activation of adaptive data reduction. `"Compression"`, `"CompressionDeduplication"`, and `"Disabled"` are supported. The default is `"Disabled"`, and If `"Disabled"` is specified, adaptive data reduction is disabled. For a storage system where the compression accelerator module is installed, if you specify `"Compression"` or `"CompressionDeduplication"` for `storageEfficiency`, the compression function using the compression accelerator module is automatically activated.

Note: For VSP One B20 series, `"Disabled"` is not supported. The default is `"CompressionDeduplication"`.

(7) Execution mode of adaptive data reduction. You can specify this parameter when `storageEfficiency` is `"Compression"` or `"CompressionDeduplication"`, and `"Inline"` and `"PostProcess"` are supported for the parameter. If `storageEfficiencyMode` is not specified, adaptive data reduction runs in the default execution mode, which depends on the storage system model. For details on the parameter, see the description of adaptive data reduction in the *Provisioning Guide for Open Systems* or *Provisioning Guide*.

Caution:

- If the LDEV was created with Storage Plug-in for Containers, do not change the parameters related to adaptive data reduction.

(8) Filesystem type. `ext4` and `xfs` are supported. If `csi.storage.k8s.io/fstype` is not specified, `ext4` is set.

(9) Secret name

(10) Secret namespace

Storage Class for VSP One SDS Block

Parameter references for sc-sample-vsp-one-sds-block.yaml

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-sample-vsp-one-sds-block                                          #(1)
  annotations:
    kubernetes.io/description: Hitachi Storage Plug-in for Containers
provisioner: hspc.csi.hitachi.com
reclaimPolicy: Delete
volumeBindingMode: Immediate
allowVolumeExpansion: true
parameters:
  storageType: vsp-one-sds-block                                             #(2)
  connectionType: fc                                                 #(3)
  storageEfficiency: Disabled                                        #(4)
  csi.storage.k8s.io/fstype: ext4                                    #(5)
  csi.storage.k8s.io/node-publish-secret-name: "secret-sample"       #(6)
  csi.storage.k8s.io/node-publish-secret-namespace: "default"        #(7)
  csi.storage.k8s.io/provisioner-secret-name: "secret-sample"        #(6)
  csi.storage.k8s.io/provisioner-secret-namespace: "default"         #(7)
  csi.storage.k8s.io/controller-publish-secret-name: "secret-sample" #(6)
  csi.storage.k8s.io/controller-publish-secret-namespace: "default"  #(7)
  csi.storage.k8s.io/node-stage-secret-name: "secret-sample"         #(6)
  csi.storage.k8s.io/node-stage-secret-namespace: "default"          #(7)
  csi.storage.k8s.io/controller-expand-secret-name: "secret-sample"  #(6)
  csi.storage.k8s.io/controller-expand-secret-namespace: "default"   #(7)
```

Legend:

(1) StorageClass name

(2) Storage type. This parameter must be set to `vsp-one-sds-block` when using VSP One SDS Block.

(3) Connection type between storage and nodes. `fc`, `iscsi`, and `nvme-tcp` are supported. If `connectionType` is not specified, `fc` is set.

(4) The setting of the data reduction function for volumes. `Compression` and `Disabled` are supported. The default is `Disabled`, and if `Disabled` is specified, the data reduction function is disabled. If you are using the multitenancy functionality, you cannot set this parameter. For multitenancy functionality, the data reduction function settings for volumes comply with the VPS settings.

(5) Filesystem type. `ext4` and `xfs` are supported. If `csi.storage.k8s.io/fstype` is not specified, `ext4` is set.

(6) Secret name

(7) Secret namespace

PersistentVolumeClaim settings

In this section, you will configure PersistentVolumeClaim settings, which are required by Storage Plug-in for Containers to dynamically create a new volume for a storage system.

The PersistentVolumeClaim file contains volume information that is used by Storage Plug-in for Containers to create PersistentVolumes. The following sample provides information about the required parameters.

Note:

- If you want to use the existing volume of the storage system as PersistentVolumeClaim, see Static provisioning (on page 50).
- If you use PersistentVolumeClaim to be configured in this section and the static provisioning function at the same time, a static PV created by following the procedure described in Creating a PV (on page 51) must be properly associated with a PVC by performing the procedure described in Creating a PVC (on page 59). If you have not performed the procedure described in Creating a PVC (on page 59), perform the following procedure before configuring the PersistentVolumeClaim settings in this section.

  1. Check PVs for which association has not been completed.

     ```
     kubectl get pv
     ```

     PVs whose `STATUS` is `Available` have not been associated.

  2. For PVs for which association has not been completed, check whether `claimRef` is specified.

     ```
     kubectl get <PV-name> -o yaml
     ```

  3. If there are any PVs for which `claimRef` is not specified, perform either of the following procedures for each PV.
     - Re-create the static PV, specify `claimRef`, and then perform the procedure described in Creating a PVC (on page 59).
     - If you do not need the PV, delete it.

If you configure PersistentVolumeClaim settings as described in this section when there is a PV for which `claimRef` is not specified, no PV will be dynamically created by Storage Plug-in for Containers, and a PVC might be associated with the static PV whose association is not complete.

Parameter references for pvc-sample.yaml

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-sample                #(1)
spec:
  accessModes:
  - ReadWriteOnce                 #(2)
  resources:
    requests:
      storage: 1Gi                #(3)
  storageClassName: sc-sample     #(4)
```

Legend:

(1) PersistentVolumeClaim name

(2) Specify `ReadWriteOnce` or `ReadOnlyMany`, or `ReadWriteMany`. To use `ReadOnlyMany`, see ReadOnlyMany (on page 44). To use `ReadWriteOnce` or `ReadWriteMany`, see Raw block volume (on page 42).

(3) Volume size

(4) StorageClass name

Usage restrictions for a PersistentVolumeClaim

- If a failure occurs when creating a PersistentVolumeClaim, a PersistentVolumeClaim object will be created without the PersistentVolume. In this case, delete the PersistentVolumeClaim object using the `kubectl delete pvc <PVC-name>` command.
- If a failure occurs when deleting a PersistentVolumeClaim, a PersistentVolumeClaim object will be deleted but the PersistentVolume object will remain and any storage asset associated with the PersistentVolume object may also remain. In this case, see Viewing the volume properties of PersistentVolume (on page 89) and obtain the volume ID of the storage. Delete the PersistentVolume using the `kubectl delete pv <PV-name>` command. Also, delete the storage asset (LDEV). For details, see the user guide for the storage system in your environment.

Pod settings

The Pod file contains volume information. Storage Plug-in for Containers mount volumes based on this information.

Parameter references for pod-sample.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-sample            #(1)
```

```
spec:
  containers:
    - name: my-busybox
      image: busybox
      volumeMounts:
      - mountPath: "/data"    #(2)
        name: sample-volume
      command: ["sleep", "1000000"]
      imagePullPolicy: IfNotPresent
  volumes:
    - name: sample-volume
      persistentVolumeClaim:
        claimName: pvc-sample  #(3)
```

Legend:

(1) Pod name

(2) Path (path where the volume is mounted inside a container)

(3) PersistentVolumeClaim name

Command examples

Following are examples of creating and deleting a Secret, StorageClass, PersistentVolumeClaim, and Pod using commands in practice.

Note: If your environment is OpenShift, replace Kubernetes Command Line Interface (CLI) with OpenShift CLI. For more information about OpenShift CLI, refer to the OpenShift CLI reference.

Create a Secret, StorageClass, PersistentVolumeClaim, and Pod

```
# kubectl create -f secret-sample.yaml
secret/secret-sample created

# kubectl get secret
NAME            TYPE     DATA   AGE
secret-sample   Opaque   3      34s

# kubectl create -f sc-sample.yaml
storageclass.storage.k8s.io/sc-sample created

# kubectl get sc
NAME        PROVISIONER            AGE
sc-sample   hspc.csi.hitachi.com   21s

# kubectl create -f pvc-sample.yaml
persistentvolumeclaim/pvc-sample created

# kubectl get pvc
NAME         STATUS   VOLUME                                     CAPACITY   ACCESS
MODES   STORAGECLASS   AGE
pvc-sample   Bound    pvc-cf8c6089-0386-4c39-8037-e1520a986a7d   1Gi
RWO            sc-sample      28s

# kubectl create -f pod-sample.yaml
pod/pod-sample created

# kubectl get pod
NAME         READY   STATUS    RESTARTS   AGE
pod-sample   1/1     Running   0          20s
```

Caution: If the LDEV was created with Storage Plug-in for Containers, do not change the nickname.

Confirm a PersistentVolume information created by Storage Plug-in for Containers

```
# kubectl get pv
NAME                                       CAPACITY   ACCESS MODES   RECLAIM
POLICY   STATUS   CLAIM                STORAGECLASS   REASON   AGE
pvc-3796f902-ed64-4636-9d25-e73e28e556f2   1Gi        RWO
Delete           Bound    default/pvc-sample   sc-sample               19h

# kubectl describe pv pvc-3796f902-ed64-4636-9d25-e73e28e556f2
Name:            pvc-3796f902-ed64-4636-9d25-e73e28e556f2
Labels:          <none>
Annotations:     pv.kubernetes.io/provisioned-by: hspc.csi.hitachi.com
                 volume.kubernetes.io/provisioner-deletion-secret-name: secret-
sample
                 volume.kubernetes.io/provisioner-deletion-secret-namespace: default
Finalizers:      [kubernetes.io/pv-protection]
StorageClass:    sc-sample
Status:          Bound
Claim:           default/pvc-sample
Reclaim Policy:  Delete
Access Modes:    RWO
VolumeMode:      Filesystem
Capacity:        1Gi
Node Affinity:   <none>
Message:
Source:
    Type:              CSI (a Container Storage Interface (CSI) volume source)
    Driver:            hspc.csi.hitachi.com
    FSType:            ext4
    VolumeHandle:      01--scsi--900000070010--914--spc-ecdf100f22
    ReadOnly:          false
    VolumeAttributes:      connectionType=fc
                           hostModeOption=
                           ldevIDDec=914
                           ldevIDHex=03:92
                           nickname=spc-ecdf100f22
                           ports=CL5-C
                           size=1Gi
                           storage.kubernetes.io/
csiProvisionerIdentity=1685522390822-8081-hspc.csi.hitachi.com
Events:                <none>
```

Delete a Secret, StorageClass, PersistentVolumeClaim, and Pod

```
# kubectl get pod
NAME         READY   STATUS    RESTARTS   AGE
pod-sample   1/1     Running   0          30s

# kubectl delete pod pod-sample
pod "pod-sample" deleted

# kubectl get pvc
NAME         STATUS   VOLUME                                     CAPACITY   ACCESS
MODES   STORAGECLASS   AGE
pvc-sample   Bound    pvc-cf8c6089-0386-4c39-8037-e1520a986a7d   1Gi
RWO            sc-sample      46s

# kubectl delete pvc pvc-sample
persistentvolumeclaim "pvc-sample" deleted

# kubectl get sc
NAME        PROVISIONER        AGE
sc-sample   hspc.csi.hitachi.com   53s

# kubectl delete sc sc-sample
storageclass.storage.k8s.io "sc-sample" deleted

# kubectl get secret
NAME            TYPE     DATA   AGE
secret-sample   Opaque   3      74s

# kubectl delete secret secret-sample
secret "secret-sample" deleted
```

Raw block volume

Kubernetes supports raw block volumes in addition to filesystem volumes. This section describes how to apply a raw block volume.

Before you begin

This feature requires the StorageClass.

Parameter references for pvc-sample-block.yaml

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-sample-block    #(1)
spec:
  accessModes:
```

```
  - ReadWriteOnce    #(2)
  volumeMode: Block
  resources:
    requests:
      storage: 1Gi    #(3)
  storageClassName: sc-sample    #(4)
```

Legend:

(1) PersistentVolumeClaim name

(2) Specify `ReadWriteOnce` or `ReadWriteMany`.

(3) Volume size

(4) StorageClass name

Parameter references for pod-sample-block.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-sample-block              #(1)
spec:
  containers:
    - name: my-busybox
      image: busybox
      volumeDevices:
      - devicePath: "/block"    #(2)
        name: sample-volume
      command: ["sleep", "1000000"]
      imagePullPolicy: IfNotPresent
  volumes:
    - name: sample-volume
      persistentVolumeClaim:
        claimName: pvc-sample-block    #(3)
```

Legend:

(1) Pod name

(2) Path (path where the volume is mounted in the container)

(3) PersistentVolumeClaim name

Command examples

- Create a PersistentVolumeClaim for a raw block volume:

```
# kubectl create -f pvc-sample-block.yaml
```

- Create a Pod for a raw block volume:

```
# kubectl create -f pod-sample-block.yaml
```

ReadWriteMany

You can mount a volume on one or many nodes in your Kubernetes cluster and perform read-write operations.

Note: OpenShift virtualization supports the ReadWriteMany access mode.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-sample-block    #(1)
spec:
  accessModes:
  - ReadWriteMany    #(2)
  volumeMode: Block
  resources:
    requests:
      storage: 7Gi    #(3)
  storageClassName: sc-sample    #(4)
```

Legend:

(1) PersistentVolumeClaim name

(2) Specify `ReadWriteMany`.

(3) Volume size

(4) StorageClass name

ReadOnlyMany

You can mount a volume on one or many nodes in your Kubernetes cluster and perform read-only operations.

To create a PersistentVolumeClaim with ReadOnlyMany, you must create the PersistentVolumeClaim from an existing PVC.

Note: Use the PersistentVolumeClaim manifest file used in the Cloning a PVC (on page 61) section and specify ReadOnlyMany, as shown in the following example.

Note: This feature is not supported in VSP One B20 series and VSP One SDS Block.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-rox-sample
spec:
  dataSource:
    name: pvc-sample
    kind: PersistentVolumeClaim
    apiGroup: ""
  accessModes:
  - ReadOnlyMany # Specify "ReadOnlyMany" here.
  resources:
    requests:
      storage: 1Gi
  storageClassName: sc-sample
```

Resource partitioning

By using this function, you can partition storage system resources for each Kubernetes cluster.

The following are examples of resource partitioning:

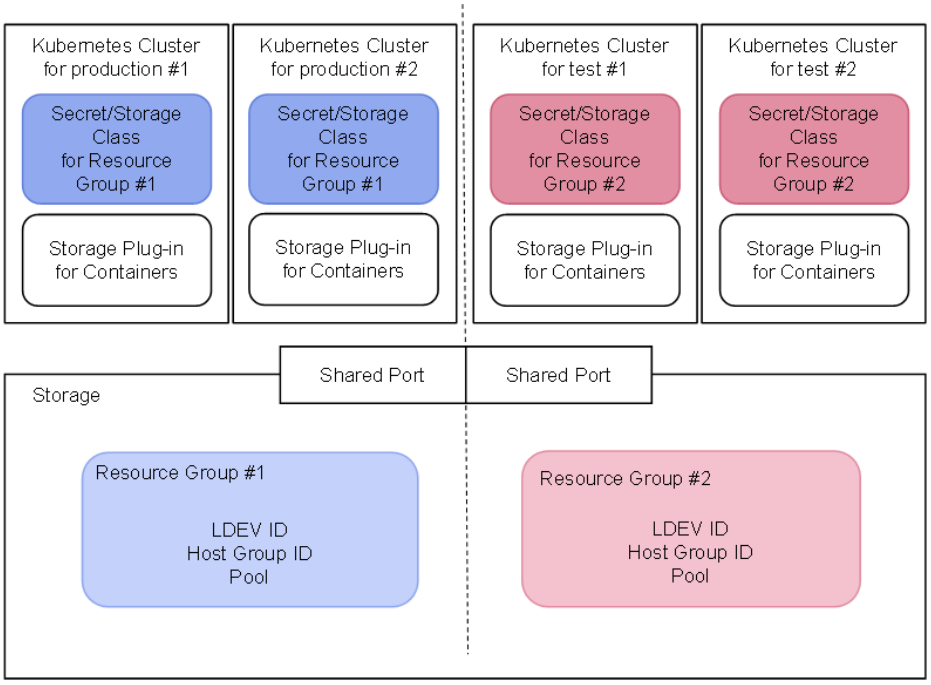
- You can restrict the range of LDEV IDs added to a resource group for a specific Kubernetes cluster.
- You can isolate the impacts between Kubernetes clusters.

Note: Resource partitioning is not supported for VSP One SDS Block.

Before you use the resource partitioning, the storage system settings, Secret and StorageClass settings, are required.

Supported configurations

The following are examples of configurations in which storage system resources can be partitioned.



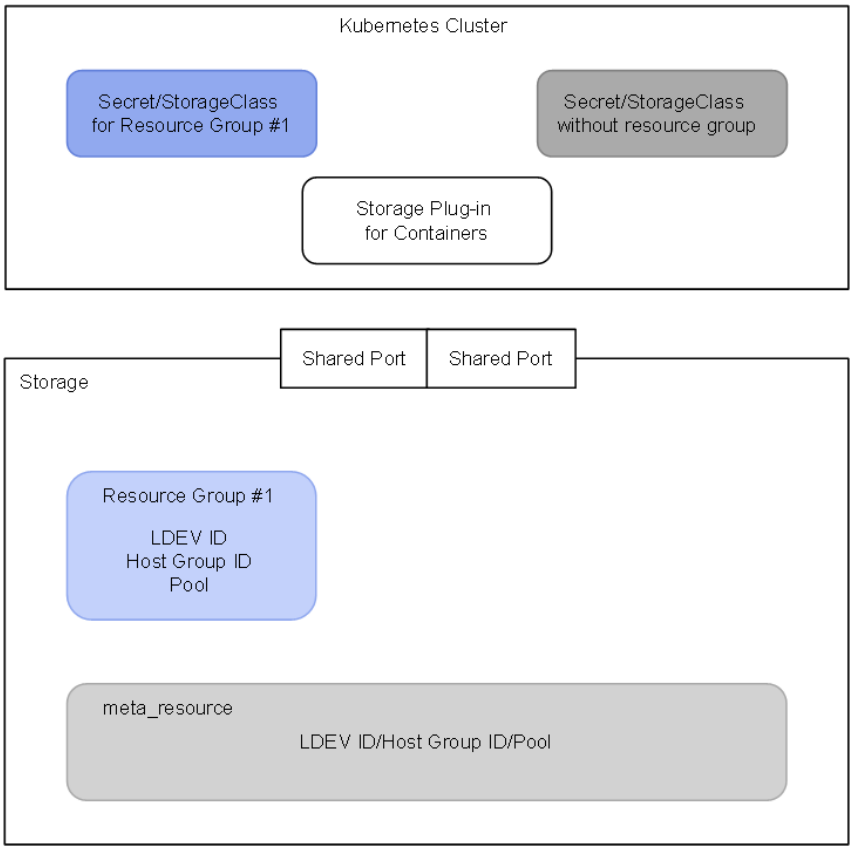
Unsupported configurations

The following are examples of configurations that are not supported.

Example 1

You cannot include both the following configurations in the same Kubernetes cluster.

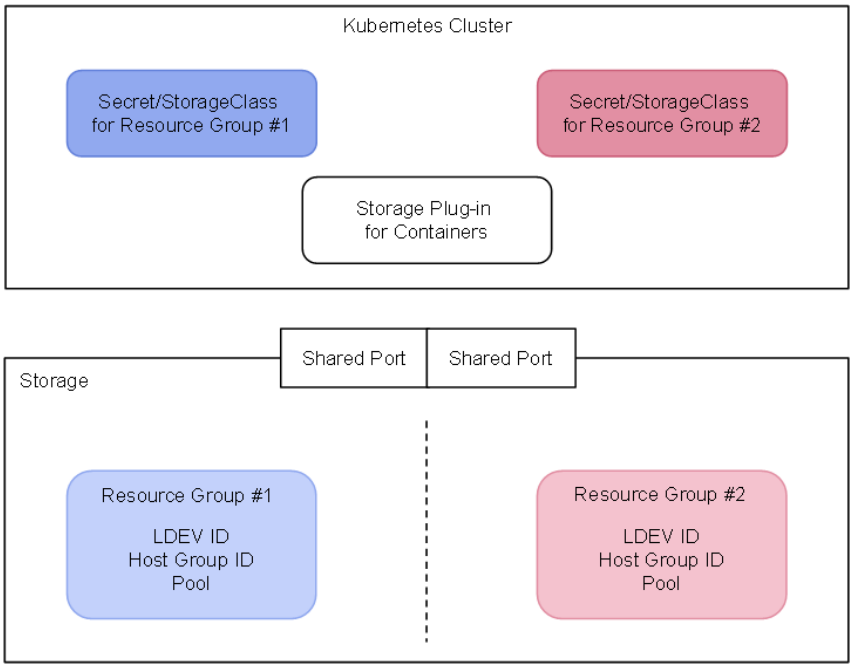
- StorageClass and Secret are configured for a resource group.
- StorageClass and Secret are temporarily configured for use with meta resource.



Example 2

If multiple resource groups are configured for a single storage system, each of those resource groups cannot correspond to a resource group in the same Kubernetes cluster.

Only one resource group (containing storageClass and Secret) per storage system can be configured for a Kubernetes cluster.



Storage system requirements and settings

Set your storage system to meet the following requirements:

Storage system resources	Descriptions
Resource group	You cannot use multiple resource groups for a single Kubernetes cluster.
Storage system user group and Storage system user	Storage system users must have access only to the resource group that you created. The storage system user must not have access to other resource groups.
Pool	Create a storage pool from the pool volumes that belong to the resource group.
LDEV ID	Allocate the necessary number of unused LDEV IDs to the resource group. If you enable the adaptive data reduction function, a deduplication system data volume is created. Register the LDEV ID required to allocate this volume to each resource group. For details about the number of LDEV IDs that need to be registered, see the user guide for the storage system in your environment.

Storage system resources	Descriptions
Host Group	For each port of a storage system defined in StorageClass, prepare the same number of host group IDs as the number of hosts. For example, if the number of hosts is 3 and the number of ports is 2, a total of 6 host group IDs are required. For each storage system port, allocate the prepared host group IDs to the resource groups.
NVM subsystem	If NVMe over FC is used, assign the NVM subsystem to the resource group.
Port	If NVMe over FC is used, assign the storage system port to the resource group.

Secret settings

Specify the resource group ID of the storage system.

Example of Secret settings:

```
apiVersion: v1
kind: Secret
metadata:
  name: secret-sample
type: Opaque
data:
  url: aHR0cDovLzE3Mi4xNi4xLjE=
  user: VXNlcjAx
  password: UGFzc3dvcmQwMQ==
stringData:
  resourceGroupID: "1"    # Specify resource group ID
```

Note: You cannot use VSM for the `resourceGroupID`.

StorageClass settings

If you use iSCSI as a storage system connection, specify the port IP address in number order. If you use Fibre Channel or NVMe over FC as a storage system connection, no additional setting is required for StorageClass.

Examples of StorageClass settings:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-sample
provisioner: hspc.csi.hitachi.com
reclaimPolicy: Delete
```

```
volumeBindingMode: Immediate
allowVolumeExpansion: true
parameters:
  serialNumber: "54321"
  poolID: "1"
  portID : CL1-A,CL2-A
  connectionType: iscsi
  portIP: "192.168.10.10, 192.168.10.11"    # Specify iSCSI Port IP Addresses.
<...>
```

Static provisioning

This function allows existing volumes in a storage system to be used as PVCs by a container orchestrator. By using this function, you can perform operations on existing volumes in the same way as PVCs dynamically provisioned by using Storage Plug-in for Containers.

Requirements for using static provisioning

The requirements for using static provisioning are as follows:

Make sure that the volume meets the following requirements:

Requirements for VSP family and VSP One Block

- The LDEV has the DP attribute.

  For VSP family, the LDEV does not have the DRS attribute.

  For VSP One Block, the LDEV has the DRS attribute.

  Tip: The DRS attribute indicates the data reduction shared volume.

- The format of the nickname is `spc-<10-digit-hexadecimal-number>`.

  <10-digit-hexadecimal-number> needs to be a unique value for each LDEV.

  If the nickname is the same as that of another LDEV, the functions supported by Storage Plug-in for Containers might not work properly.

  You can also use a command to generate a unique character string for a nickname. The following is an example command:

  ```
  # echo spc-$(cat /dev/urandom | tr -dc a-f0-9 | head -c 10)
  ```

- The LDEV is not mapped to a port.
- No pairs are formed.

For other requirements, see Prerequisites (on page 11).

If the LDEV is assigned to a specific resource group, also see the storage system requirements in Resource partitioning (on page 45).

Note: Static provisioning is not supported for stretched PVCs.

Requirements for VSP One SDS Block

Refer to the *Hitachi Virtual Storage Software Block Storage Administrator Guide* and check the following requirements.

- `naaId` can be obtained as volume information.

  For volumes created for VSP One SDS Block v1.11 or earlier versions, `naaId` cannot be obtained. Therefore, these volumes are not supported.

- `volumeType` is `Normal`.

- `name` and `nickname` have the same value, and their format is `spc-<10-digit-hexadecimal-number>`.

  You can also use a command to generate a unique character string for `name`. The following is an example command:

```
# echo spc-$(cat /dev/urandom | tr -dc a-f0-9 | head -c 10)
```

- The volume is not connected to a computer node.

For other requirements, see Prerequisites (on page 11).

If the volume is assigned to a specific VPS, also see Multitenancy functionality settings (on page 23).

Creating Secret and StorageClass

Create the Secret and StorageClass to be specified when creating a PV and PVC.

For details about the settings in the YAML files for Secret and StorageClass, see Secret settings (on page 32) and StorageClass settings (on page 33). If you are using VSP family or VSP One Block and the volume is assigned to a specific resource group, also see the descriptions of Secret settings and StorageClass settings in Resource partitioning (on page 45).

For the parameters of StorageClass, specify values based on the status of the target volume. If the values specified for the parameters do not match the actual volume status, the functions supported by Storage Plug-in for Containers might not work properly.

Creating a PV

Create a PV to be associated with a PVC.

Note:

- You can create only one PV for one volume.

  If you create more than one PV for one volume, the functions supported by Storage Plug-in for Containers might not work properly.
- If values specified for the parameters of the PV are incorrect, an unexpected error message might be displayed.

Procedure

1. Create a YAML file.

   For the parameters of the PV, specify values based on the status of the target volume and the settings of the created StorageClass. If the values specified for the parameters do not match the actual volume status, the functions supported by Storage Plug-in for Containers might not work properly.

   Example of the YAML file:

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: static-pv
  annotations:
    pv.kubernetes.io/provisioned-by: hspc.csi.hitachi.com
  spec:
  persistentVolumeReclaimPolicy: Delete
  accessModes:
  - ReadWriteOnce
  capacity:
    storage: 1Gi
  volumeMode: Filesystem
  csi:
    fsType: ext4
    volumeAttributes:
      connectionType: fc
      ports: CL1-A, CL2-A
    volumeHandle: 01--scsi--900000070010--50000--spc-c3d46c5a71
    driver: hspc.csi.hitachi.com
    controllerExpandSecretRef:
      name: secret-sample
      namespace: default
    controllerPublishSecretRef:
      name: secret-sample
      namespace: default
  storageClassName: sc-sample
  claimRef:
    name: static-pvc
    namespace: default
```

   Parameters for VSP family and VSP One Block:

Parameter	Description	Required or optional
`metadata.name`	Specify the PV name.	Required
`metadata.annotations.pv.kubernetes.io/provisioned-by`	Specify `hspc.csi.hitachi.com`.	Required if `Delete` is specified for `persistentVolumeReclaimPolicy`
`spec.persistentVolumeReclaimPolicy`	Specify the reclaim policy. If you specify `Retain`, the PV and LDEV will not be deleted even when the PVC is deleted. Instead, the PV will be in the released state, so if you want to reuse the LDEV, you will need to re-create the PV. For details, see https://kubernetes.io/docs/concepts/storage/persistent-volumes/#retain. If you specify `Delete`, the PV and LDEV will be deleted when the PVC is deleted. The default value is `Retain`.	Optional
`spec.accessModes`	Specify the access mode. For details about supported access modes, see PersistentVolumeClaim settings (on page 37).	Required
`spec.capacity.storage`	Specify the LDEV size.	Required
`spec.volumeMode`	Specify `Filesystem` or `Block`. If the target LDEV was used as a raw block volume, be sure to specify `Block`. If the LDEV was used as a raw block volume and you specify `Filesystem`, it will be formatted as a file system and the existing data will be deleted. The default value is `Filesystem`.	Optional
`spec.csi.fsType`	Specify the file system type of the target LDEV. The default value is `ext4`.	Optional

Parameter	Description	Required or optional
	If you specify `Block` for `volumeMode`, this parameter is disabled. For details about supported file system types, see StorageClass settings (on page 33).	
`spec.csi.volumeAttributes.connectionType`	Specify the type of connection between the storage system and the node. For details about supported connection types, see StorageClass settings (on page 33).	Required
`spec.csi.volumeAttributes.ports`	Specify the storage port ID. For multipath configurations, use commas to delimit the storage port IDs.	Required if `fc` or `iscsi` is specified for `connectionType`
`spec.csi.volumeAttributes.portIPs`	Specify the storage port IP address. For multipath configurations, use commas to delimit the storage port IP addresses.	Required if `iscsi` is specified for `connectionType` and the LDEV is assigned to a specific resource group
`spec.csi.volumeAttributes.nvmSubsystemID`	Specify the NVM subsystem ID of the storage system.	Required if `nvme-fc` is specified for `connectionType`
`spec.csi.volumeHandle`	Specify the value in the following format: `01--<IO-protocol>--<storage-device-ID>--<LDEV-ID>--<LDEV-nickname>`	Required

Parameter	Description	Required or optional
	<IO-protocol> Specify the value as follows: ▪ If `fc` or `iscsi` is specified for `connectionType`: `scsi` ▪ If `nvme-fc` is specified for `connectionType`: `nvme` ***<storage-device-ID>*** Check this value by referring to the *REST API Reference Guide* for each storage model. The storage device ID is a 12-digit value and the format is as follows: *<6-digit-fixed-value-for-each-storage-model><6-digit-serial-number>* For example, the fixed value for VSP 5100 is `900000`. ***<LDEV-ID>*** Check these values by using the storage system management software. Specify a value by using a decimal number. ***<LDEV-nickname>*** Check these values by using the storage system management software.	
`spec.csi.driver`	Specify `hspc.csi.hitachi.com`.	Required
`spec.csi.controllerExpandSecretRef.name`	Specify the name of the Secret.	Required
`spec.csi.controllerExpandSecretRef.namespace`	Specify the namespace of the Secret.	Required

Parameter	Description	Required or optional
`spec.csi.controllerPublishSecretRef.name`	Specify the name of the Secret.	Required
`spec.csi.controllerPublishSecretRef.namespace`	Specify the namespace of the Secret.	Required
`spec.storageClassName`	Specify the StorageClass name.	Required
`spec.claimRef.name`	Specify the PVC name to be created in Creating a PVC (on page 59).	Required If you do not specify this parameter, the PV might be associated with an unintended PVC and the Storage Plug-in for Containers functions might not work properly.
`spec.claimRef.namespace`	Specify the namespace of the PVC to be created in Creating a PVC (on page 59).	Required

Example of the YAML file for VSP One SDS Block:

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: static-pv
  annotations:
    pv.kubernetes.io/provisioned-by: hspc.csi.hitachi.com
spec:
  persistentVolumeReclaimPolicy: Delete
  accessModes:
  - ReadWriteOnce
  capacity:
    storage: 1Gi
  volumeMode: Filesystem
  csi:
    fsType: ext4
    volumeAttributes:
      connectionType: fc
    volumeHandle: 60060e81166020006016602000000004f--spc-909b93359a--4026f840-e15e-
4410-a37f-3862072c10ba
    driver: hspc.csi.hitachi.com
```

```
  controllerExpandSecretRef:
    name: secret-sample
    namespace: default
  controllerPublishSecretRef:
    name: secret-sample
    namespace: default
storageClassName: sc-sample-vsp-one-sds-block
claimRef:
  name: static-pvc
  namespace: default
```

Parameters for VSP One SDS Block:

Parameter	Description	Required or optional
`metadata.name`	Specify the PV name.	Required
`metadata.annotations.pv.kubernetes.io/provisioned-by`	Specify hspc.csi.hitachi.com.	Required if `Delete` is specified for `persistentVolumeReclaimPolicy`
`spec.persistentVolumeReclaimPolicy`	Specify the reclaim policy. If you specify `Retain`, the PV and volume will not be deleted even when the PVC is deleted. Instead, the PV will be in the released state, so if you want to reuse the volume, you will need to re-create the PV. For details, see https://kubernetes.io/docs/concepts/storage/persistent-volumes/#retain. If you specify `Delete`, the PV and volume will be deleted when the PVC is deleted. The default value is `Retain`.	Optional
`spec.accessModes`	Specify the access mode. For details about supported access modes, see PersistentVolumeClaim settings (on page 37).	Required
`spec.capacity.storage`	Specify the volume size.	Required
`spec.volumeMode`	Specify `Filesystem` or `Block`.	Optional

Parameter	Description	Required or optional
	If the target volume was used as a raw block volume, be sure to specify `Block`. If the volume was used as a raw block volume and you specify `Filesystem`, it will be formatted as a file system and the existing data will be deleted. The default value is `Filesystem`.	
`spec.csi.fsType`	Specify the file system type of the target volume. The default value is `ext4`. If you specify `Block` for `volumeMode`, this parameter is disabled. For details about supported file system types, see StorageClass settings (on page 33).	Optional
`spec.csi.volumeAttributes.connectionType`	Specify the type of connection between the storage system and the node. For details about supported connection types, see StorageClass settings (on page 33).	Required
`spec.csi.volumeHandle`	Specify the value in the following format: `<volume-naaId>--<volume-name>--<volume-id>` ***<volume-naaId>*** `naaId` of the target volume ***<volume-name>*** `name` of the target volume ***<volume-id>*** `id` of the target volume	Required

Parameter	Description	Required or optional
	For details about how to check `naaId`, `name`, and `id`, see the *Hitachi Virtual Storage Software Block Storage Administrator Guide*.	
`spec.csi.driver`	Specify `hspc.csi.hitachi.com`.	Required
`spec.csi.controllerExpandSecretRef.name`	Specify the name of the Secret.	Required
`spec.csi.controllerExpandSecretRef.namespace`	Specify the namespace of the Secret.	Required
`spec.csi.controllerPublishSecretRef.name`	Specify the name of the Secret.	Required
`spec.csi.controllerPublishSecretRef.namespace`	Specify the namespace of the Secret.	Required
`spec.storageClassName`	Specify the StorageClass name.	Required
`spec.claimRef.name`	Specify the PVC name to be created in Creating a PVC (on page 59).	Required If you do not specify this parameter, the PV might be associated with an unintended PVC and the Storage Plug-in for Containers functions might not work properly.
`spec.claimRef.namespace`	Specify the namespace of the PVC to be created in Creating a PVC (on page 59).	Required

2. Deploy the YAML file.

```
# kubectl apply -f <YAML-file-name>
```

Creating a PVC

Create a PVC so that Storage Plug-in for Containers functions can be used.

Procedure

1. Create a YAML file.

   Example of the YAML file:

   The following shows examples for VSP family or VSP One Block.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: static-pvc
  namespace: default
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: sc-sample
  volumeMode: Filesystem
  volumeName: static-pv
```

   Parameters:

Parameter	Description	Required or optional
`metadata.name`	Specify the PVC name.	Required
`metadata.namespace`	Specify the namespace of the PVC.	Optional
`spec.accessModes`	Specify the same value as the value of `accessModes` for the PV.	Required
`spec.resources.requests.storage`	Specify the same value as the value of `capacity.storage` for the PV.	Required
`spec.storageClassName`	Specify the same value as the value of `storageClassName` for the PV.	Required
`spec.volumeName`	Specify the PV name.	Required

Parameter	Description	Required or optional
		If you do not specify this parameter, the PVC might be associated with an unintended PV and the Storage Plug-in for Containers functions might not work properly.
`spec.volumeMode`	If you specified `volumeMode` for the PV, specify the same value for this parameter as you did for the PV. The default value is `Filesystem`.	Required if you specified `Block` for `volumeMode` of the PV

2. Deploy the YAML file.

```
# kubectl apply -f <YAML-file-name>
```

3. Confirm that STATUS of the PVC is Bound.
   When STATUS of the PVC is Bound, you can use the functions supported by Storage Plug-in for Containers.

Cloning a PVC

This feature can create a duplicate as a clone of an existing volume. A clone can be consumed in the same way as any standard volume.

Note:

- If the volume is expanded, confirm for completion before executing this feature. Refer to Expanding storage volume for a PVC (on page 67) for details.
- Flush the data before cloning for data consistency. For example, temporarily remove the Pod.
- This feature is not supported on VSP One SDS Block.

Before you begin

This feature requires the following resources:

- StorageClass
- PersistentVolumeClaim

Parameter references for pvc-from-pvc-sample.yaml

This YAML file is a manifest file for creating a clone from an existing volume "pvc-sample".

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-pvc-sample      #(1)
spec:
  dataSource:
    name: pvc-sample             #(2)
    kind: PersistentVolumeClaim
    apiGroup: ""
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi               #(3)
  storageClassName: sc-sample    #(4)
```

Legend:

(1) PersistentVolumeClaim name of clone

(2) PersistentVolumeClaim name of source

(3) Specify the size of the source volume. Obtain the size by using the `kubectl get pv <PV-name> -o yaml` command, which is displayed in **size**.

Note: If the volume is expanded or it is a statically provisioned PersistentVolume, obtain the size by using the `kubectl get pv <PV-name>` command, which is displayed in **CAPACITY**.

(4) Specify the same StorageClass name as the one used for dataSource.

Command examples

- Create a PersistentVolumeClaim for a clone:

```
# kubectl create -f pvc-from-pvc-sample.yaml
```

- Create a Pod from a clone:

```
example yaml file
kind: Pod
apiVersion: v1
metadata:
  name: pod-clone-pvc
  namespace: <name of the namespace>
spec:
  containers:  # Must be a list
    - name: my-busybox  # Add "-" to indicate a list item
      image: busybox
      volumeMounts:
        - mountPath: "/data"  #(2)
          name: sample-volume
      command: ["sleep", "1000000"]
      imagePullPolicy: IfNotPresent
  volumes:  # Must be a list
    - name: sample-volume
      persistentVolumeClaim:
        claimName: pvc-from-pvc-sample
```

Creating a snapshot for a PVC

This feature allows you to create a snapshot that is a point-in-time image of a volume. Snapshots can be used to restore or duplicate a previous state of an existing volume.

Note:

- On VSP One B20 series storage systems, when a clone is created from a snapshot, the snapshot is converted into a PVC. The snapshot entry may still appear in Kubernetes; however, it becomes a stale entry and cannot be used to create additional clones. No corresponding snapshot exists on the storage system.
- If the volume has been expanded, verify that the operation has successfully completed before creating the volume snapshot. See Expanding storage volume for a PVC (on page 67) for more details.
- Flush the data before creating a snapshot for data consistency. For example, temporarily remove the Pod.
- This feature is not supported on VSP One SDS Block.

Before you begin

This feature requires the following resources:

- StorageClass
- PersistentVolumeClaim

If your environment is Kubernetes, install Snapshot CRDs and Snapshot Controller per cluster. See https://github.com/kubernetes-csi/external-snapshotter. For Snapshot CRDs, use v1.

Note: If Snapshot Alpha or Beta CRDs are present in your environment, remove them before installing Snapshot v1 CRDs.

Parameter references for volumesnapshotclass-sample.yaml

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: snapshotclass-sample    #(1)
driver: hspc.csi.hitachi.com
deletionPolicy: Delete
parameters:
  poolID: "1"               #(2)
  retentionPeriod: "1"              #(3)
  csi.storage.k8s.io/snapshotter-secret-name: "secret-sample" #(4)
  csi.storage.k8s.io/snapshotter-secret-namespace: "default"  #(5)
```

Legend:

(1) `VolumeSnapshotClass` name

(2) Same poolID as the `StorageClass`

(3) (Optional) Retention period: Specify a retention period between 1 and 12,288 hours to create immutable snapshots.

Once a snapshot data retention period is configured, the snapshot data remains protected from any updates until the specified period expires. After the retention period ends, the snapshot data retention setting for the pair is automatically disabled.

Note:

- If `retentionPeriod` is specified in the `VolumeSnapshotClass`, the snapshot cannot be deleted or cloned for the duration of the defined retention period.
- Immutable snapshots is currently supported only on VSP One B20 series storage systems.

(4) Same Secret name as the `StorageClass`

(5) Same Secret namespace as the `StorageClass`

Parameter references for volumesnapshot-sample.yaml

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: snapshot-sample                                  #(1)
spec:
  volumeSnapshotClassName: snapshotclass-sample          #(2)
  source:
    persistentVolumeClaimName: pvc-sample                #(3)
```

Legend:

(1) VolumeSnapshot name

(2) VolumeSnapshotClass name

(3) PersistentVolumeClaim name from which the snapshot is obtained

Parameter references for pvc-from-snapshot-sample.yaml

Note: Creating a persistent volume from a snapshot is not supported on VSP One Block High End storage systems.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snapshot-sample    #(1)
spec:
  dataSource:
    name: snapshot-sample           #(2)
    kind: VolumeSnapshot
    apiGroup: snapshot.storage.k8s.io
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi                 #(3)
  storageClassName: sc-sample      #(4)
```

Legend:

(1) PersistentVolumeClaim name

(2) VolumeSnapshot name

(3) Specify the size of the source volume. Obtain the size by using the `kubectl get pv <PV-name> -o yaml` command, which is displayed in **size**.

Note: If the volume is expanded or it is a statically provisioned PersistentVolume, obtain the size by using the `kubectl get pv <PV-name>` command. The size is displayed in **CAPACITY**.

(4) Specify the same StorageClass name as the one used for dataSource.

Command examples

- Create a VolumeSnapshotClass:

```
# kubectl create -f volumesnapshotclass-sample.yaml
```

- Create a VolumeSnapshot:

```
# kubectl create -f volumesnapshot-sample.yaml
```

- Verify that `readyToUse` is true. If it is true, the creation of VolumeSnapshot is complete.

```
# kubectl get volumesnapshot -o yaml
```

> **Note:** If `readyToUse` is false, confirm the cause and solution by following the steps:
>
> 1. Obtain the `boundVolumeSnapshotContentName` by using the command: **`kubectl get volumesnapshot -o yaml`**
> 2. Confirm the error message by using the command: **`kubectl describe volumesnapshotcontent <VolumeSnapshotContentName>`**

- Create a PersistentVolumeClaim from a snapshot:

```
# kubectl create -f pvc-from-snapshot-sample.yaml
```

- Create a Pod:

```
apiVersion: v1
kind: Pod
metadata:
  name: pod-sample            #(1)
spec:
  containers:
    - name: my-busybox
      image: busybox
      volumeMounts:
      - mountPath: "/data"    #(2)
        name: sample-volume
      command: ["sleep", "1000000"]
      imagePullPolicy: IfNotPresent
  volumes:
    - name: sample-volume
      persistentVolumeClaim:
        claimName: pvc-from-snapshot-sample  #(3)
```

Expanding storage volume for a PVC

This feature can expand the capacity of an existing volume. There is no need to delete and recreate the Pod for volume expansion.

Caution:

- Confirm completion of volume expansion with the `kubectl get pvc` command, which is displayed in **CAPACITY**. Do not shut down the OS or drain the node before volume expansion completes.
- In NVMe over FC and NVMe/TCP, volume expansion in the state in which the PVC is attached to a host is not supported. Expand a volume in the state in which the PVC is detached from the host.

Before you begin

This feature requires the following resources:

- StorageClass
- PersistentVolumeClaim

Note: Volume expansion has the following restrictions:

- On VSP One B20 series storage systems, you cannot expand a cloned PVC to a capacity that is greater than its parent volume.
- The minimum additional size for volume expansion is 1 GiB.
- The maximum additional size for volume expansion is 7 TiB or a value that does not exceed the warning threshold of pool capacity. If you add more than 7 TiB, run the command again.
- Volume capacity cannot be reduced.
- If `allowVolumeExpansion` of StorageClass is set to `false`, a PersistentVolume created with this setting cannot be expanded.
- The size obtained by the `kubectl get pv <PV-name> -o yaml` command is not updated after the volume is expanded. If the volume is expanded, obtain the size by using the `kubectl get pv <PV-name>` command, which is displayed in **CAPACITY**.

Command examples

- Expand the capacity of an existing volume pvc-sample to 5GiB:

```
# kubectl patch pvc pvc-sample --patch \
'{"spec":{"resources":{"requests":{"storage": "5Gi"}}}}'
```

- Confirm the completion of volume expansion by looking at **CAPACITY**:

```
# kubectl get pv <PV-name>
NAME            CAPACITY     ACCESS MODES      RECLAIM POLICY
STATUS CLAIM                 STORAGECLASS      REASON AGE
<PV-name>       5Gi          RWO               Delete
Bound default/pvc-sample     sc-sample               35s
```

Caution:

- Use the `kubectl` command to change the size of an LDEV created with Storage Plug-in for Containers.
- After you apply the patch for volume expansion, the volume capacity increases. However, the PVC capacity might not be updated if it is not attached to a Pod. You must expand the filesystem from within the node.

Deleting resources created by Storage Plug-in for Containers

You can delete the resources created by Storage Plug-in for Containers.

To delete a pod, enter:

```
kubectl delete pod <pod-name>
```

To delete a PVC, enter:

```
kubectl delete pvc <pvc-name>
```

Note: On VSP One B20 series storage systems, if the PVC is cloned, then you must first delete the cloned PVC before you delete the parent PVC.

To delete a snapshot, enter:

```
kubectl delete vs <snapshot-name>
```

Troubleshooting when using static provisioning

A VolumeAttachment remains after you delete the Pod

If there are mistakes in the format or values of parameters in the YAML file for the PV, the creation of the Pod will fail. If you delete this Pod, a VolumeAttachment might remain.

The following are examples of failed attempts to create a Pod:

- Incorrect format of `volumeHandle` (error code: `0x0000c002`)
- Insufficient permissions for LDEV (error code: `0x00001007`)
- `controllerPublishSecretRef` is not specified (error code: `0x0000c00f`)

To delete a VolumeAttachment, run the following command:

```
# kubectl patch volumeattachments <VolumeAttachment-name> --type merge -p
'{"metadata":{"finalizers":null}}'
```

The PV remains after you delete the PVC

If there are mistakes in the format or values of parameters in the YAML file for the PV, even if you specify `Delete` for `persistentVolumeReclaimPolicy`, the PV might remain after you delete the PVC.

The following shows examples of a PV remaining after a PVC is deleted:

- Incorrect format of `volumeHandle` (error code: `0x0000c002`)
- Insufficient permissions for LDEV (error code: `0x00001007`)

To delete the PV, perform the following procedure:

1. If you need to delete the volume, use the `volumeHandle` information to check the target volume.

   ```
   # kubectl get pv <PV-name> -o yaml
   ```

2. Delete the PV.

   ```
   # kubectl delete pv <PV-name>
   ```

3. If you need to delete the volume, use the management software for the storage system to delete it.

The volume remains after you delete the PVC

In the following cases, if you specify `Delete` for `persistentVolumeReclaimPolicy`, the PV will be deleted after the PVC is deleted, but the volume will not be deleted.

VSP family and VSP One Block:

For the LDEV specified by <*LDEV-ID*> of `volumeHandle`, an incorrect value is specified for <*LDEV-nickname*>.

VSP One SDS Block:

- For the volume specified by <*volume-id*> of `volumeHandle`, an incorrect value is specified for <*volume-naaId*> or <*volume-name*>.
- For the volume specified by <*volume-id*> of `volumeHandle`, user permissions specified in the Secret are insufficient.

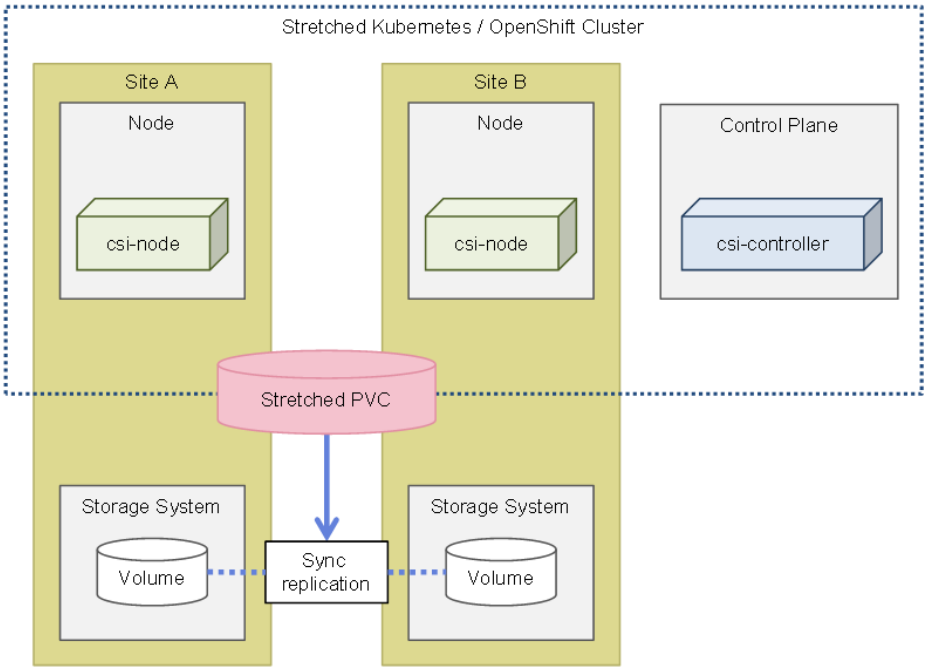
To delete the volume, use the management software for the storage system to delete it.

Stretched PVC

The Stretched PersistentVolumeClaim (PVC) feature automates the provisioning of synchronous replication between the storage system at each site in a single Kubernetes or OpenShift cluster that spans two sites. By using this feature, you can build a high-availability cluster that includes storage systems at two sites.

The provisioning of synchronous replication for conventional systems requires the storage system administrator, cluster administrator, and user to cooperate closely. By using the Stretched PVC feature, the cluster administrator and user can perform the provisioning of synchronous replication by themselves by using the command line tool for Kubernetes or OpenShift.

The following figure gives an overview of a cluster environment in which Stretched PVC is installed.



Requirements for using the Stretched PVC feature

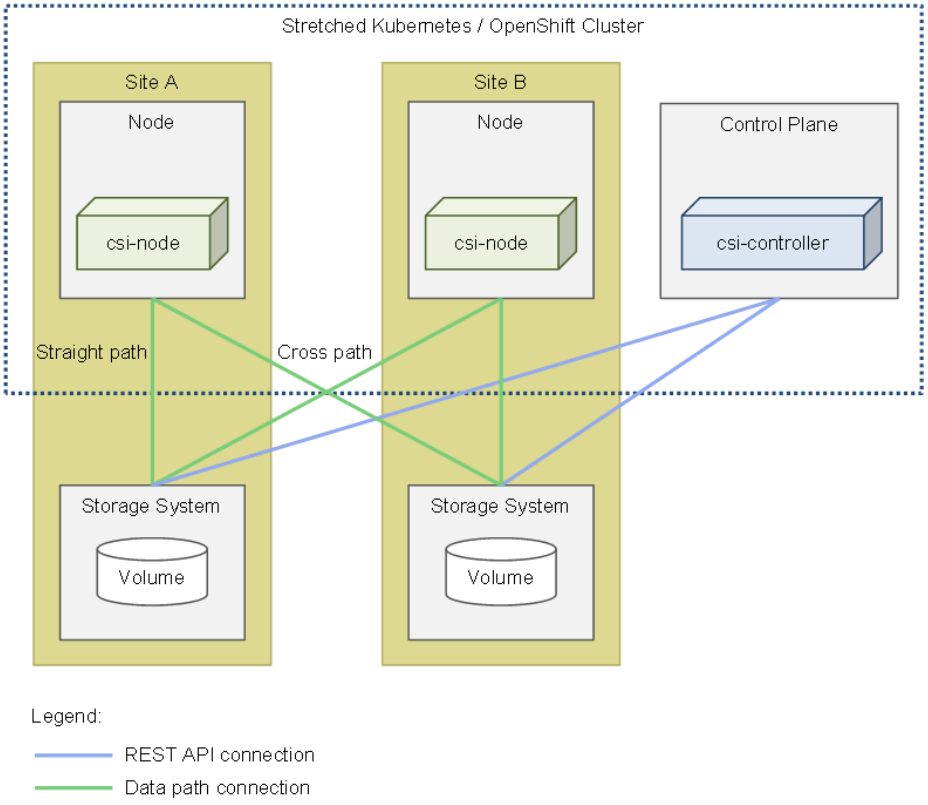
The Stretched PVC feature automates replication provisioning. Note that this feature includes neither the management and monitoring of the replication status, nor the management and monitoring of the status of the primary and secondary sites.

Requirements for all systems

The following table shows the system-wide requirements for using Stretched PVC.

Item	Description
Cluster configuration	Configure a single Kubernetes or OpenShift cluster (stretched cluster) that spans two sites. Then, allocate one storage system to each site.
REST API connection	Specify settings so that Storage Plug-in for Containers can communicate with the storage systems at the primary site and at the secondary site by using the REST API.
Data path connection	Specify settings so that each node can perform data communication with the storage systems at the primary site and at the secondary site. Only Fibre Channel is supported for communication between nodes and storage systems.

The following is an overview of connecting the server and storage systems in a stretched cluster.



Requirements and settings for storage systems

Check the following requirements and settings for both the primary site and secondary site. For details, see the *Global-Active Device User Guide*.

Item	Description
Storage system models	The following storage systems are supported: ▪ VSP One Block 20 series ▪ VSP E series ▪ VSP 5000 series ▪ VSP F350, F370, F700, F900 ▪ VSP G350, G370, G700, G900
License	Apply the required license by referring to the *Global-Active Device User Guide*.
Virtual storage machines	For each storage system, create one virtual storage machine. The virtual storage machine must meet the following requirements: ▪ At the primary site and the secondary site, specify the same serial number and model for the virtual storage machines. Note that a virtual storage machine is also required at the primary site. ▪ For the model of the virtual storage machines, specify any storage system model supported by the Stretched PVC feature. ▪ Use the virtual storage machines only for Storage Plug-in for Containers. Do not use the virtual storage machines for other purposes. Note the following when using the same combination of storage systems from multiple clusters: ▪ For each cluster, use the same storage system users and virtual storage machines. ▪ Make sure that the combination of the storage system set for the primary site and the storage system set for the secondary site for each cluster matches each other.
Users and user groups	Create a user group with access only to the virtual storage machine's resource group and add users to it.
Pool	Create a storage pool from the pool volumes that belong to the resource group.
LDEV ID	Assign the required number of unused LDEV IDs to the virtual storage machine. For the assigned LDEVs, change the virtual LDEV ID to unassigned (65534).

Item	Description
Host groups	For each port that you want to use, assign the required number of unused host group IDs for the virtual storage machine. The number of host group IDs must be at least the total number of hosts + 1.
Remote paths	Create remote paths. The same path group ID must be used for both the primary site and the secondary site.
Quorum disks	Create quorum disks. The same quorum disk ID must be used for both the primary site and the secondary site.

Using a regular PVC in a stretched cluster

To use a regular PVC instead of Stretched PVC in a stretched cluster, specify settings in advance by referring to Resource partitioning (on page 45).

Note: For host groups, use the ports that are not used by Stretched PVcs.

When connecting to a Pod, if you want to control the site to which the Pod will be deployed, use the Node Affinity or Node Selector feature of Kubernetes and OpenShift.

When using Stretched PVC together with other features

The following table shows whether Stretched PVC can be used together with other features:

Feature	Can be used with Stretched PVC?
Volume snapshot	Yes (For details about how to use it, see Creating a clone from a Stretched PVC (on page 79).)
Volume cloning	Yes (For details about how to use it, see Creating a clone from a Stretched PVC (on page 79).)
Volume expansion	Yes
Raw block volume	Yes
ReadWriteMany	Yes
ReadOnlyMany	Yes
Adaptive data reduction	Yes

Creating a Stretched PVC

You can create a Secret and StorageClass specifically for a Stretched PVC, and then use the created Secret and StorageClass to create a Stretched PVC. You can also use the created Stretched PVC in the same way as a regular PVC.

When Storage Plug-in for Containers receives the Secret and StorageClass specifically for the Stretched PVC, Storage Plug-in for Containers creates volumes at the specified primary site and secondary site and generates a global-active device pair. The volumes created at the primary site and secondary site and the global-active device pair must be deleted when the Stretched PVC is deleted. Use Storage Navigator to manage the status of the global-active device pair in the period between the creation and deletion of the Stretched PVC.

To create a Stretched PVC:

Procedure

1. Create a Secret specifically for the Stretched PVC. A sample (`secret-sample-stretched.yaml`) can be found in the yaml directory.

```
apiVersion: v1
kind: Secret
metadata:
  name: secret-sample-stretched
type: Opaque
stringData:
  primarySerial: "11111"                  #(1)
  primaryURL : http://172.16.0.1          #(2)
  primaryUser : primary-user              #(3)
  primaryPassword : primary-password      #(4)
  secondarySerial: "22222"                #(5)
  secondaryURL : http://172.16.0.2        #(6)
  secondaryUser : secondary-user          #(7)
  secondaryPassword : secondary-password  #(8)
```

   Legend:

   (1) Serial number of the storage system at the primary site

   (2) URL of the REST API of the storage system at the primary site

   (3) User of the storage system at the primary site

   (4) User password for the storage system at the primary site

   (5) Serial number of the storage system at the secondary site

   (6) URL of the REST API of the storage system at the secondary site

   (7) User of the storage system at the secondary site

   (8) User password for the storage system at the secondary site

2. Create a StorageClass specifically for the Stretched PVC. A sample (`sc-sample-stretched.yaml`) can be found in the yaml directory.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-sample-stretched
  annotations:
```

```
    kubernetes.io/description: Hitachi Storage Plug-in for Containers
provisioner: hspc.csi.hitachi.com
reclaimPolicy: Delete
volumeBindingMode: Immediate
allowVolumeExpansion: true
parameters:
  connectionType: fc
  replicationType: stretched                    #(1)
  copyGroupName: "spc<copy group name>"    #(2)
  copyPairName: "spc<copy pair name>"      #(3)
  consistencyGroupId: "1"                       #(4)
  quorumID: "30"                                #(5)
  primaryPoolID: "10"                           #(6)
  primaryPortID: CL1-A,CL2-A                    #(7)
  secondaryPoolID: "20"                         #(8)
  secondaryPortID: CL1-F                        #(9)
  csi.storage.k8s.io/node-publish-secret-name: "secret-sample-stretched"
  csi.storage.k8s.io/node-publish-secret-namespace: "default"
  csi.storage.k8s.io/provisioner-secret-name: "secret-sample-stretched"
  csi.storage.k8s.io/provisioner-secret-namespace: "default"
  csi.storage.k8s.io/controller-publish-secret-name: "secret-sample-stretched"
  csi.storage.k8s.io/controller-publish-secret-namespace: "default"
  csi.storage.k8s.io/node-stage-secret-name: "secret-sample-stretched"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
  csi.storage.k8s.io/controller-expand-secret-name: "secret-sample-stretched"
  csi.storage.k8s.io/controller-expand-secret-namespace: "default"
```

Legend:

(1) Parameter indicating that the replication type is Stretched PVC

(2) Optional parameter. Specify a unique, case-sensitive `copyGroupName` between 1–29 characters, starting with the `spc` prefix. Omitting the prefix will cause an error.

(3) Optional parameter. Specify a unique, case-sensitive `copyPairName` between 1–31 characters, starting with the `spc` prefix. Omitting the prefix will cause an error.

Note:

- The `copyPairName` and `consistencyGroupId` are applied only when a custom `copyGroupName` is provided. Otherwise, the system uses the default values for these parameters.
- The `copyPairName` name is unique to the copy group.

(4) Optional parameter. Specify the consistency group ID by using a decimal (base 10) number.

(5) ID of the Quorum disk

(6) Pool ID of the storage system at the primary site

(7) Port ID of the storage system at the primary site

(8) Pool ID of the storage system at the secondary site

(9) Port ID of the storage system at the secondary site

3. Create a PVC by specifying the created StorageClass.
4. Display the created PVC to make sure that STATUS is Bound.

Note: Creating a Stretched PVC takes more time than creating a regular PVC.

Checking information on Stretched PVCs

A PV is created after a Stretched PVC is created. To view storage system resources created by the Storage Plug-in for Containers, run `kubectl describe pv <PV-name>` and check `spec.csi.volumeAttributes`. You can also use this information for troubleshooting.

The `spec.csi.volumeAttributes` include the following parameters:

Parameter	Description
nickname	Nickname of the created LDEV (the same for both the primary site and the secondary site)
primarySerial	Serial number of the storage system at the primary site
primaryVolumeID	ID of the LDEV created at the primary site
secondarySerial	Serial number of the storage system at the secondary site
secondaryVolumeID	ID of the LDEV created at the secondary site

Creating a Stretched PVC with Adaptive Data Reduction (ADR)

Storage Plug-in for Containers supports Adaptive Data Reduction (ADR) for stretched PVCs that are provisioned using global-active device volumes.

Procedure

1. Create a StorageClass for the Stretched PVC with ADR. You can find a sample (`sc-sample-stretched-adr.yaml`) file in the `yaml` directory.

   Add the `storageEfficiency` and `storageEfficiencyMode` parameters to the `yaml` file to enable ADR for a stretched PVC.

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-sample-stretched-adr
  annotations:
    kubernetes.io/description: Hitachi Storage Plug-in for Containers
provisioner: hspc.csi.hitachi.com
reclaimPolicy: Delete
volumeBindingMode: Immediate
```

```
allowVolumeExpansion: true
parameters:
  connectionType: fc
  quorumID: "4"
  primaryPoolID: "1"
  primaryPortID: CL1-A
  secondaryPoolID: "2"
  secondaryPortID: CL1-B
  storageEfficiency: "CompressionDeduplication"  #(1)
  storageEfficiencyMode: "PostProcess"           #(2)
  csi.storage.k8s.io/node-publish-secret-name: "secret-sample-stretched"
  csi.storage.k8s.io/node-publish-secret-namespace: "default"
  csi.storage.k8s.io/provisioner-secret-name: "secret-sample-stretched"
  csi.storage.k8s.io/provisioner-secret-namespace: "default"
  csi.storage.k8s.io/controller-publish-secret-name: "secret-sample-stretched"
  csi.storage.k8s.io/controller-publish-secret-namespace: "default"
  csi.storage.k8s.io/node-stage-secret-name: "secret-sample-stretched"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
  csi.storage.k8s.io/controller-expand-secret-name: "secret-sample-stretched"
  csi.storage.k8s.io/controller-expand-secret-namespace: "default"
```

Legend:

(1) The storage efficiency for ADR. The value can be `"CompressionDeduplication"` or `"Compression"`

(2) The storage efficiency mode for ADR. The value can be `"PostProcess"` or `"Inline"`

2. Create a PVC by specifying the created StorageClass.
3. Display the created PVC to make sure that the STATUS is Bound.

Note: Creating a Stretched PVC takes more time than creating a regular PVC.

Expanding storage volumes for a Stretched PVC

You can expand storage volumes on both a stretched PVC and a stretched PVC with ADR.

Procedure

1. Create a stretched PVC. See Creating a Stretched PVC (on page 73).

   or

   Create a stretched PVC with ADR. See Creating a Stretched PVC with Adaptive Data Reduction (ADR) (on page 76).

Note:

- Verify that the value for the `allowVolumeExpansion` parameter is set to `true`. If the parameter is set to `false`, you will not be able to expand the storage volume.
- Verify that the GAD pair is healthy and in `Pair` state during volume expansion.

2. Run the following command to expand an existing stretched PVC:

```
kubectl patch pvc <pvc name> --patch '{"spec":{"resources":{"requests":
{"storage": "16Gi"}}}}' -n <namespace>
```

Note: The storage size must be more than the PVC size for the expansion to be successful.

Creating snapshots for a Stretched PVC

You can create and manage snapshots of Global Active Device (GAD) volumes.

Procedure

1. Create a stretched PVC. See Creating a Stretched PVC (on page 73).

   or

   Create a stretched PVC with ADR. See Creating a Stretched PVC with Adaptive Data Reduction (ADR) (on page 76).

2. Create a `VolumeSnapshotClass` yaml file. The following is a sample `yaml` file:

```
apiVersion: storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: snapshotclass-sample-stretched
driver: hspc.csi.hitachi.com
deletionPolicy: Delete
parameters:
  poolID: "1"
  serialNumber: "715035"  # Required for some volume types
  csi.storage.k8s.io/snapshotter-secret-name: "secret-sample-stretched"
  csi.storage.k8s.io/snapshotter-secret-namespace: "default"
```

3. Create a `VolumeSnapshot` yaml file. The following is a sample `yaml` file:

```
apiVersion: storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: VolumeSnapshot-stretched
spec:
  volumeSnapshotClassName: snapshotclass-sample-stretched
```

```
  source:
    persistentVolumeClaimName: pvc-sample
```

Creating a clone from a Stretched PVC

You can replicate Stretched PVC data to a regular PVC by creating a volume snapshot from the Stretched PVC and running a clone from the volume snapshot.

Before you begin

Before creating the clone data for a Stretched PVC, configure the storage system for a regular PVC. For details, see the explanations related to using a regular PVC in a stretched cluster, in Requirements for using the Stretched PVC feature (on page 70).

Item	Description
Storage system	Specify either a primary site or a secondary site.
Storage system user	Assign both the resource group for the virtual storage machine used for a stretched PVC and the resource group used for a regular PVC. Do not assign other resource groups.
LDEV	Assign the necessary number of LDEV IDs to the resource group used for a regular PVC. The assigned IDs must use sequential numbers. If the numbers are not sequential, the operations might not work correctly.

Procedure

1. Create a stretched PVC. See Creating a Stretched PVC (on page 73).
2. Create a stretched PVC snapshot.See Creating snapshots for a Stretched PVC (on page 78).
3. Create a Secret for a regular PVC. See Secret settings (on page 32).
4. Create a StorageClass for a regular PVC. See StorageClass settings (on page 33).
5. Create a PVC from the stretched PVC snapshot. Enter:

   Parameter references for pvc-from-stretched-snapshot-sample.yaml

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-stretched-snapshot-sample
spec:
  dataSource:
   name: VolumeSnapshot-stretched
   kind: VolumeSnapshot
   apiGroup: snapshot.storage.k8s.io
  accessModes:
   - ReadWriteOnce
```

```
resources:
 requests:
   storage: 1Gi
storageClassName: sc-sample
```

Connecting a Stretched PVC to a Pod

Similar to a regular PVC, a Stretched PVC can be used from a Pod.

When you create a Pod, the following message is output to the log of Storage Plug-in for Containers. The log displays the serial number of the storage system that successfully connected to the Pod. Generally, both the storage system at the primary site and the one at the secondary site will be used. However, depending on the status of the storage systems, only one of these storage systems (either the one at the primary site or the one at the secondary site) might be used.

```
[INFO]storages used for publish : [11111 22222]
```

Deleting a Stretched PVC

You can delete a Stretched PVC in the same way as deleting a regular PVC. When a Stretched PVC is deleted, the global-active device pair and the volumes of the storage systems at both the primary and secondary sites are also deleted.

Note:

- Deleting a Stretched PVC takes more time than deleting a regular PVC.
- If the Stretched PVC cannot be deleted, follow the guidance provided in the error message to determine the next steps. If the problem persists, see Forcibly deleting a Stretched PVC (on page 80).

Forcibly deleting a Stretched PVC

To delete a Stretched PVC, the storage systems at the primary site and secondary site must be running normally. If a failure occurs on the storage system at either site or on the global-active device pair, deletion of the Stretched PVC will fail. To force-delete a Stretched PVC, perform the following steps:

Procedure

1. Check the information about the resources created at the primary and secondary sites. See Checking information on Stretched PVCs (on page 76).
2. Delete the PVC and the PV.
3. Based on the information you checked in step 1, use Storage Navigator to delete the global-active device pair, volumes, and the associated LUNs. In addition, change the virtual LDEV IDs of the deleted volumes to unassigned (65534).

Failover

If a failure occurs on a storage system or at a site, allocate the Pod to a different site at your discretion. A Stretched PVC can continue to operate by using the volumes of the storage system that is running normally.

Note: Use Element manager for VSP One Block storage systems to split or resync a GAD pair.

Troubleshooting

Creating a Stretched PVC

If processing to create a Stretched PVC does not finish (the status does not change to Bound), run `kubectl describe pvc <PVC-name>` to check the error details. In addition, if the following error message appears, this indicates that volumes have already been created on the storage system. If you delete the PVC before the status of the PVC changes to Bound, the volumes will remain on the storage system without being deleted. After the PVC is deleted, use Storage Navigator to delete the volumes and the associated LUNs. In addition, change the virtual LDEV IDs of the deleted volumes to unassigned (65534).

```
failed to provision volume with StorageClass  ... : volumes created : [{id:100
serial:11111} {id:200 serial:22222}]
```

Pair status

If a failure occurs in a storage system, the global-active device pairs configuring the Stretched PVC might have an abnormal status. If an error occurs when you are using the Stretched PVC or a Pod connected to the Stretched PVC becomes unavailable, refer to the Global-Active Device User Guide and perform recovery for the storage system failure and the pair status. If recovery from the failure is difficult, by referring to Creating a clone from a Stretched PVC (on page 79) and creating a volume clone of either the primary or secondary storage system, you can continue to use the clone as the regular PVC.

Host failure

If a host fails, the status of the Pod run by the relevant host becomes Terminating and the Pod has an abnormal state. If you want to use this Pod with a normal host where the failure did not occur, do the following.

1. Forcibly delete the Pod.

   ```
   kubectl delete pod <Pod-name> --grace-period=0 --force
   ```

2. Delete the VolumeAttachment related to the Stretched PVC that the Pod was using.

   ```
   kubectl delete volumeattachment <VolumeAttachment-name>
   ```

Chapter 4: Upgrade

This chapter describes how to upgrade Storage Plug-in for Containers. The upgrade method you use depends on whether your environment is OpenShift or Kubernetes.

Note:

- If alpha versions of VolumeSnapshotClass and VolumeSnapshot are present in your environment, remove them before upgrading Storage Plug-in for Containers.
- If you are using an old version of OpenShift or Kubernetes, the latest Storage Plug-in for Containers might not operate. If you want to use the latest Storage Plug-in for Containers, upgrade OpenShift or Kubernetes first.
- Before upgrading Storage Plug-in for Containers, back up the settings configured in Configuration of Storage Plug-in for Containers instance (on page 27). The settings that are backed up might also be used for the new version of Storage Plug-in for Containers. If you do not have the hspc_v1_hspc.yaml file that was used when configuring the settings, run `kubectl get hspc -A -o yaml` to obtain the configured settings.

Upgrade on OpenShift

For OpenShift, you can upgrade Storage Plug-in for Containers using OpenShift web console.

Procedure

1. Delete HSPC on the Storage Plug-in for Containers tab for Operator Details.
2. Uninstall the Operator of Storage Plug-in for Containers.
3. Install new Storage Plug-in for Containers. See Installation on OpenShift (on page 25).

Upgrade on Kubernetes

For Kubernetes, you can upgrade Storage Plug-in for Containers by following the steps below.

Procedure

1. Delete Storage Plug-in for Containers and the Operator by using the previous version package:

```
# kubectl delete -f hspc_v1_hspc.yaml
# kubectl delete -f hspc-operator.yaml
```

2. Install new Storage Plug-in for Containers. See the topic Installation on Kubernetes (on page 26).

Chapter 5: Re-creation

This chapter describes how to re-create Storage Plug-in for Containers. The re-creation method depends on whether your environment is OpenShift or Kubernetes. To re-create Storage Plug-in for Containers, delete Storage Plug-in for Containers, and then create it again.

Deleting Storage Plug-in for Containers

OpenShift

Access the OpenShift web console, and delete the Storage Plug-in for Containers instance.

Kubernetes

Run the following command to delete the Storage Plug-in for Containers instance:

```
# kubectl delete hspc -n <Storage-Plug-in-for-Containers-namespace> hspc
```

For <*Storage-Plug-in-for-Containers-namespace*>, specify the namespace specified in `hspc_v1_hspc.yaml`.

Re-creating Storage Plug-in for Containers

OpenShift

Access the OpenShift web console, and create a Storage Plug-in for Containers instance.

For details, see Installation on OpenShift (on page 25).

Kubernetes

Run the following command to create a Storage Plug-in for Containers instance:

```
# kubectl create -f hspc_v1_hspc.yaml
```

For details, see Installation on Kubernetes (on page 26).

Chapter 6: Uninstallation

This chapter describes how to uninstall Storage Plug-in for Containers. This step includes removing any PersistentVolumeClaims, PersistentVolumes, StorageClasses, Storage Plug-in for Containers, and other elements. The uninstallation method you use depends on whether your environment is OpenShift or Kubernetes.

Uninstallation on OpenShift

For OpenShift, you can uninstall Storage Plug-in for Containers using OpenShift web console.

Procedure

1. Delete all Pods which are using the volumes created by Storage Plug-in for Containers.
2. Delete the VolumeSnapshotClass, VolumeSnapshot, PersistentVolumeClaim, the StorageClass, and the Secret that were created in relation to Storage Plug-in for Containers.
3. Delete HSPC on the Hitachi Storage Plug-in for Containers tab of the Operator Details.
4. Uninstall the Operator of Storage Plug-in for Containers.

Uninstallation on Kubernetes

For Kubernetes, you can uninstall Storage Plug-in for Containers by following the steps below.

Procedure

1. Delete all Pods which are using the volumes created by Storage Plug-in for Containers.
2. Delete the VolumeSnapshotClass, VolumeSnapshot, PersistentVolumeClaim, the StorageClass, and the Secret that were created in relation to Storage Plug-in for Containers.
3. Delete Storage Plug-in for Containers and the resources for the Operator:

```
# kubectl delete -f hspc_v1_hspc.yaml
# kubectl delete -f hspc-operator.yaml
# kubectl delete -f hspc-operator-namespace.yaml
```

4. If you created a Secret to pull an image, delete the Secret.

Chapter 7: Troubleshooting

When troubleshooting, you must understand what information to collect when an error occurs, the cases where an error occurs, and what action to take in each case.

Collecting information for troubleshooting

If a failure occurs in Storage Plug-in for Containers, collect the following information. Provide the collected information to customer support when you make an inquiry.

Information needed when contacting support

You can provide the following information for Storage Plug-in for Containers and the storage system to customer support for advanced troubleshooting.

Information	Procedure
Command execution logs	Retrieve the command that you ran and the result of running that command.
Result of running the `kubectl describe` command for the operation target resource	Run the following command for the resource you operated. `# kubectl describe <resource> -n <Storage-Plug-in-for-Containers-namespace> <resource-name>`
Cluster information	Run the following command: `# kubectl cluster-info dump -A > dump.txt`
Pod information	Retrieve the command that you ran in step 1 of Collecting logs for Storage Plug-in for Containers (on page 87) and the result of running that command.
Operator logs	See Collecting logs for Storage Plug-in for Containers (on page 87).
CSI controller logs	See Collecting logs for Storage Plug-in for Containers (on page 87).

Information	Procedure
CSI node logs	See Collecting logs for Storage Plug-in for Containers (on page 87).
PVC-related manifests	Get the YAML files for StorageClass, Secret, and PersistentVolumeClaim.
Snapshot-related manifests	Get the YAML files for VolumeSnapshotClass, Secret, and VolumeSnapshot.
Snapshot-related logs	Collect the snapshot controller logs that you installed in the Creating a snapshot for a PVC (on page 63) chapter.
Application manifests	Get the YAML files for applications that uses Storage Plug-in for Containers PVCs.
Storage logs	See Collecting storage system information for VSP family and VSP One Block (on page 88) or Collecting storage system information for VSP One SDS Block (on page 88).

Collecting logs for Storage Plug-in for Containers

You can retrieve logs for your running containers using the `kubectl logs` command. To collect Storage Plug-in for Containers logs, you need to collect logs from the Operator, CSI controller, and CSI node.

Note: If necessary, set up cluster-level logging to save logs:
https://kubernetes.io/docs/concepts/cluster-administration/logging/

1. Before retrieving logs, run the following command to check the Pod name.

```
# kubectl get pod -A -o wide
```

2. Run the following command to retrieve logs.
   - Operator

     ```
     # kubectl logs -n <Storage-Plug-in-for-Containers-namespace> hspc -operator-controller-manager-<id>
     ```

   - CSI controller

     ```
     # kubectl logs -n <Storage-Plug-in-for-Containers-namespace> hspc -csi-controller-<id> -c hspc-csi-driver
     ```

     Note: You must specify `-c hspc-csi-driver` to run the command.

   - CSI node

     ```
     # kubectl logs -n <Storage-Plug-in-for-Containers-namespace> hspc-csi-node-<id> -c hspc-csi-driver
     ```

     Note:
     - You must specify `-c hspc-csi-driver` to run the command.
     - You will see multiple CSI node Pods because this is deployed as a DaemonSet. Collect logs from all these Pods.

3. Perform the following procedure to retrieve the directories as old logs might be rotated and removed from the retrieved logs.
   a. Check the target node.

      From the result of step 1, find the line where the value of `NAME` includes `hspc-csi-controller`, and check the value of `NODE` in the same line.

   b. Retrieve the directories stored under `/var/log/pods` for the node you checked in step a.

Collecting storage system information for VSP family and VSP One Block

If you are using an SVP, collect the regular dump files.

If you are not using an SVP, collect system dumps using the maintenance utility. For details about how to collect the dump files of storage systems, see the *System Administrator Guide*.

Collecting storage system information for VSP One SDS Block

Collect the dump files. For the procedure on collecting dump files, contact customer support.

Viewing the volume properties of PersistentVolume

When a volume is dynamically created by Storage Plug-in for Containers, information about the created volume is set in the `spec.csi.volumeAttributes` of the PersistentVolume. You can view these properties using the `kubectl get pv <PV-name> -o yaml` command.

These properties are mainly used for internal purposes. The following tables describe some properties that can be helpful when troubleshooting.

Volume properties for VSP family and VSP One Block.

Property	Description
ldevIDDec	Decimal LDEV ID
ldevIDHex	Hexadecimal LDEV ID
size	Capacity of the volume **Note:** Capacity shown here is the original capacity used when creating the volume.

Volume properties for VSP One SDS Block.

Property	Description
volumeID	ID of the volume created in VSP One SDS Block
size	Capacity of the volume **Note:** Capacity shown here is the original capacity used when creating the volume.

Notes on forcibly deleting a Pod

If you forcibly delete a Pod from a specific node, the deleted Pod and the information on the PVC associated with the Pod might remain on the relevant node, and an unexpected error might occur.

To properly delete this information, you must restart the relevant node before using the node again.

Creating and deleting PersistentVolumeClaim simultaneously

When PersistentVolumeClaims are created or deleted simultaneously, the storage might get overloaded and cause errors `0x0000100b`, `0x0000100f`, `0x0000101a`, or `0x0000f007`. This problem can be reduced by specifying the `--worker-threads` argument to the `csi-provisioner` container. This argument limits the number of simultaneously running create and delete operations. The default value is 20.

The following example shows how to reduce the number of `--worker-threads` to 10. For the YAML configuration, refer to Configuration of Storage Plug-in for Containers instance (on page 27).

```
apiVersion: csi.hitachi.com/v1
kind: HSPC
metadata:
  name: hspc
  namespace: <Storage-Plug-in-for-Containers-namespace>
spec:
  controller:
    containers:
      - name: csi-provisioner
        args:
        - --csi-address=/csi/csi-controller.sock
        - --timeout=300s
        - --v=5
        - --worker-threads=10
        - --default-fstype=ext4
```

If the problem persists, contact technical support.

Host group settings

If you encounter error `0x00001023`, you must modify the host group in the storage. Storage Plug-in for Containers searches the host group named "spc-<*wwn1*>-<*wwn2*>-<*wwn3*>", based on the naming rules (see Host group and iSCSI target naming rules (on page 22)). The error was likely generated because the host group's name may not follow the "spc-<*wwn1*>-<*wwn2*>-<*wwn3*>" naming format. To resolve the issue, delete the host group shown in the error message and rename the host group that has host WWNs.

1. Delete Storage Plug-in for Containers.

   For details, see Deleting Storage Plug-in for Containers (on page 84).
2. Delete the host group that is specified in the error message.
3. Search host groups that have WWNs for each host, and delete them or rename them to "spc-<*wwn1*>-<*wwn2*>-<*wwn3*>".
4. Create Storage Plug-in for Containers.

   For details, see Re-creating Storage Plug-in for Containers (on page 84).

Timeout errors when creating or deleting Pods

Depending on the status of the storage system or host OS, processing of Storage Plug-in for Containers might slow down and a timeout error might occur.

If a timeout error continues to occur when you create or delete a Pod, see Re-creation (on page 84) and re-create a Storage Plug-in for Containers instance. If the timeout error is not resolved, restart the node to which the Pod where the timeout error occurs is assigned. Restarting a node might affect other environments. For this reason, sufficiently check the scope of the impact, see the Kubernetes and OpenShift documentation, and then carefully restart the node while checking the procedures one by one.

Unable to schedule or create a Kubernetes pod

You cannot schedule or create a Kubernetes pod when you use PersistentVolumes configured on multipath-enabled block devices.

This could be when multipath services or modules are not configured correctly, or they are not running on the node where the pod is scheduled. The storage device, typically a SAN disk is not discovered or available to the container runtime.

Perform the following steps to resolve the issue:

1. Verify the Multipath service status. Enter:

   ```
   “systemctl status multipathd”
   ```

   Check if the `multipathd` daemon is active and running. If the deamon is not in the running state, continue to step 2.

2. Enable and start the Multipath service. Enter:

   ```
   “systemctl enable --now multipathd”
   ```

   This command starts the `multipathd` service immediately and verifies it when you boot the node.

3. Load the Multipath kernel module required for device-mapper multipath. Enter:

   ```
   “modprobe dm-multipath”
   ```

4. Verify the Multipath devices that are detected. Enter:

   ```
   “multipath -ll”
   ```

   Note:

   - You must reboot the node after configuring the multipath settings.
   - Verify that the storage devices used by the PersistentVolume are listed.

Troubleshooting virtual LDEV ID error

During operations, the following error may appear for a specific LDEV:

```
Internal desc = [HSPC0x0000b007] a virtual LDEV ID is defined : remove the virtual
LDEV ID, and consider using resource partitioning feature if it is not already
configured : ldev [140] has a virtual ldev id [140]
```

This occurs when an invalid or undefined Virtual LDEV ID is associated with the physical LDEV.

Perform the following steps to resolve the issue using the `raidcom` utility:

Note: You can also use other tools to remove the Virtual LDEV ID.

1. Verify the LDEV status. Enter:

   ```
   raidcom get ldev -ldev_id <ldev id> -fx
   ```

   Verify the value for the `VIR_LDEV` parameter. If the value is `fffe`, it indicates an invalid virtual LDEV entry is mapped.

2. Remap the virtual LDEV. Enter:

   ```
   raidcom map resource -ldev_id <ldev id> -virtual_ldev_id <ldev id>
   ```

   This updates the virtual LDEV reference and clears the invalid mapping.

3. Reverify the LDEV status. Enter:

   ```
   raidcom get ldev -ldev_id <ldev id> -fx
   ```

   Note: Verify that you use the same LDEV ID.

   Verify the value for the `VIR_LDEV` parameter. If no virtual LDEV is listed, it indicates that the invalid virtual LDEV has been successfully removed.

Appendix A: Advanced configuration

The appendix provides supplemental information that supports the main content of this document.

Editing the existing LUN configuration

You can update the LUN configuration by modifying the `SPC_LUN_MAX` parameter. By default, 256 LUNs are supported. However, you can update the value for the `SPC_LUN_MAX` parameter to 2048.

WARNING: Do not set the value for `SPC_LUN_MAX` parameter greater than 2048.

Note: Changing the environment value of `SPC LUN MAX` also require appropriate kernel level configuration based on the HBA driver to discover and support the LUNs. Refer to the vendor-specific guide to verify the settings.

Perform the following steps to patch the Storage Plug-in for Containers CSI driver image:

Procedure

1. Navigate to the cluster where Storage Plug-in for Containers is installed.
2. Log on to the controller node.
3. (Optional) If the Storage Plug-in for Containers was deployed using an operator, scale down the operator deployment to 0 replicas. Enter:

   ```
   kubectl scale deployment hspc-operator-controller-manager --replicas=0 -n
   <Storage-Plug-in-for-Containers-namespace>
   ```

4. Edit the Storage Plug-in for Containers deployment YAML file. Enter:

   ```
   kubectl edit deployment  hspc-csi-controller -n <Storage-Plug-in-for-Containers-
   namespace>
   ```

5. Under the `env` section of the `hspc-csi-driver` container, update the `SPC_LUN_MAX` parameter value. Enter:

   ```
   - env:
   - name: SPC_LUN_MAX
   value: "2048"
   ```

6. Save the changes.

Result

The pods are redeployed with the updated configuration.

Migrating existing ports of a `StorageClass`

You can update the existing ports of a `StorageClass`.

Before you begin

Verify the following secret keys are available:

- portID
- For iSCSI: portIP

For a stretched PVC, verify the following details are available:

- primaryPortID
- secondaryPortID
- For iSCSI: primaryPortIP and secondaryPortIP

Perform either of the following procedures to migrate the existing ports of a `StorageClass`.

- Modifying the StorageClass and PersistentVolumes (on page 94)
- Modifying the Secrets (on page 95)

Note: The following are the limitations when configuring ports using secrets:

- PersistentVolumes (PVs) are immutable. Once created, they retain the original port values defined at provisioning. The ports mapped to the PV correspond to the ports defined in the secrets.
- Port migration is not supported for NVMe over FC and NVMe/TCP. This protocol does not use `PortID`, but depends on the subsystem details. Create a subsystem with the required ports or update an existing subsystem. See the *Provisioning Guide for Open Systems* or the *Provisioning Guide*.

Modifying the StorageClass and PersistentVolumes

1. Identify the pod that requires port migration.
2. Locate the StorageClass, PV, and PVC used by that pod.
3. Capture the port information from the `StorageClass`.
4. Delete the pod.
5. Edit the PV previously associated with the deleted pod and modify the `persistentVolumeReclaimPolicy` parameter.
6. Update the parameter value from `Delete` to `Retain`.
7. Back up the `PV.yaml` file.
8. Delete the PVC.
9. Delete the PV.
10. Delete the `StorageClass`.

> **Note:** Make a note of the `StorageClass` name and the configuration parameters.

11. Create a new `StorageClass.yaml` file with the required ports.

> **Note:** Verify that the `StorageClass` name and the configuration parameters are the same, except for the port changes.

12. Update the backed-up `PV.yaml` file with the new port information defined in the new `StorageClass.yaml` file.
13. Recreate the PV using the modified `PV.yaml` file.
14. Create a PVC for the static PV.

> **Note:** Verify that the PVC name and the configuration parameters are the same as before.

15. Create the pod.

Modifying the Secrets

Procedure

1. Create `StorageClass`. See StorageClass settings (on page 33).
2. Create a PVC. See Creating a PVC (on page 59).
3. Create a pod. See Pod settings (on page 38).
   Verify that the mapped ports are the same that are defined in the `StorageClass`.
4. Delete the pod.
   This unmounts the volume and unmaps the ports.
5. Update the `secret.yaml` file to configure or change the port. Enter:

> **Note:** Verify that all values under `data` are base64-encoded.

```
portID: Q0wyLUc=     # CL2-G
portIP: MTcyLjI1LjU5LjE5Mg==  # 172.25.59.192
```

You can also add the configuration in plain text under `stringData`. Kubernetes will automatically convert to base64. For example:

```
apiVersion: v1
stringData:
  portID: CL2-G
  portIP: 172.25.59.192
data:
  password: SGl0YWNoaTE=
  url: aHR0cHM6Ly8xNzIuMjUuNDcuNTU=
  user: dWNwYQ==
kind: Secret
```

6. Recreate the pod.

   The CSI driver now uses the updated ports from the secret instead of the immutable StorageClass.
7. Verify that the pod is running. Enter:

```
kubectl get pods
```

Modifying the bulk mapping mode

Storage Plug-in for Containers supports two mapping modes, controlled by the `IS_BULK_MAPPING_MODE_ENABLED` environment variable. When set to `true`, the Storage Plug-in for Containers uses the Bulk API to map volumes to all the required ports in a single operation. When set to `false`, the volumes are mapped one port at a time using individual API calls.

Note: This feature is not supported on VSP G350, G370, G700, G900 storage models.

Perform the following steps:

Procedure

1. Navigate to the cluster where Storage Plug-in for Containers is installed.
2. Log on to the controller node.
3. (Optional) If the Storage Plug-in for Containers was deployed using an operator, scale down the operator deployment to 0 replicas. Enter:

```
kubectl scale deployment hspc-operator-controller-manager --replicas=0 -n
<Storage-Plug-in-for-Containers-namespace>
```

4. Edit the CSI Controller deployment YAML file. Enter:

```
kubectl edit deployment hspc-csi-controller -n <Storage-Plug-in-for-Containers-
namespace>
```

5. Locate the `IS_BULK_MAPPING_MODE_ENABLED` variable.
6. Update the value of the variable as required:

   Set the value to `true` to enable bulk mapping mode. For example:

```
IS_BULK_MAPPING_MODE_ENABLED = true
```

   Set the value to `false` to disable bulk mapping mode. For example:

```
IS_BULK_MAPPING_MODE_ENABLED = false
```

   Note: By default, the value of the `IS_BULK_MAPPING_MODE_ENABLED` variable is set to `true`.

7. Save the changes and exit the edit mode.
   Kubernetes will automatically roll out the updated configuration.
8. Verify the roll out status. Enter:

```
kubectl rollout status deployment hspc-csi-controller -n <Storage-Plug-in-for-Containers-namespace>
```

9. Verify the updated settings. Enter:

```
kubectl get deploy hspc-csi-controller -n $NAMESPACE -o jsonpath='{.spec.template.spec.containers[0].env}'
```

10. (Optional) Restart the controller pods manually. Enter:

```
kubectl delete pod -n $NAMESPACE -l app=hspc-csi-controller
```

Modifying CPU and memory resources for the CSI Controller deployment

You can update the CPU and memory limits for the `hspc-csi-controller` deployment.

Perform the following steps:

Procedure

1. Navigate to the cluster where Storage Plug-in for Containers is installed.
2. Log on to the controller node.
3. (Optional) If the Storage Plug-in for Containers was deployed using an operator, scale down the operator deployment to 0 replicas. Enter:

```
kubectl scale deployment hspc-operator-controller-manager --replicas=0 -n <Storage-Plug-in-for-Containers-namespace>
```

4. Edit the CSI Controller deployment YAML file. Enter:

```
kubectl edit deployment hspc-csi-controller -n <Storage-Plug-in-for-Containers-namespace>
```

5. Locate the `cpu` and `memory` settings under `resources`.
6. Update the `cpu` and `memory` values as required:
   For example:

```
resources:
  limits:
    cpu: 1000m
    memory: 1Gi
  requests:
```

```
    cpu: 200m
    memory: 256Mi
```

7. Save the changes and exit the edit mode.

   Kubernetes will automatically roll out the updated configuration.

8. Verify the roll out status. Enter:

```
kubectl rollout status deployment hspc-csi-controller -n <Storage-Plug-in-for-Containers-namespace>
```

9. Verify the updated settings. Enter:

```
kubectl get pod -n <Storage-Plug-in-for-Containers-namespace> -l app=hspc-csi-controller -o jsonpath='{.items[*].spec.containers[*].resources}'
```

Note: Scaling the operator deployment to 1 replicas resets the CPU and memory settings to the default values defined in the CSI Controller deployment YAML file.

Hitachi Vantara

Corporate Headquarters
2535 Augustine Drive
Santa Clara, CA 95054 USA

HitachiVantara.com/contact