

Hitachi Storage Plug-in for Containers

Version 3.11.0

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.

© 2017, 2022 Hitachi, Ltd. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including copying and recording, or stored in a database or retrieval system for commercial purposes without the express written permission of Hitachi, Ltd., or Hitachi Vantara LLC (collectively "Hitachi"). Licensee may make copies of the Materials provided that any such copy is: (i) created as an essential step in utilization of the Software as licensed and is used in no other manner; or (ii) used for archival purposes. Licensee may not make any other copies of the Materials. "Materials" mean text, data, photographs, graphics, audio, video and documents.

Hitachi reserves the right to make changes to this Material at any time without notice and assumes no responsibility for its use. The Materials contain the most current information available at the time of publication.

Some of the features described in the Materials might not be currently available. Refer to the most recent product announcement for information about feature and product availability, or contact Hitachi Vantara LLC at https://support.hitachivantara.com/en_us/contact-us.html.

Notice: Hitachi products and services can be ordered only under the terms and conditions of the applicable Hitachi agreements. The use of Hitachi products is governed by the terms of your agreements with Hitachi Vantara LLC.

By using this software, you agree that you are responsible for:

1. Acquiring the relevant consents as may be required under local privacy laws or otherwise from authorized employees and other individuals; and
2. Verifying that your data continues to be held, retrieved, deleted, or otherwise processed in accordance with relevant laws.

Notice on Export Controls. The technical data and technology inherent in this Document may be subject to U.S. export control laws, including the U.S. Export Administration Act and its associated regulations, and may be subject to export or import regulations in other countries. Reader agrees to comply strictly with all such regulations and acknowledges that Reader has the responsibility to obtain licenses to export, re-export, or import the Document and any Compliant Products.

Hitachi and Lumada are trademarks or registered trademarks of Hitachi, Ltd., in the United States and other countries.

AI/AS/400e, DB2, Domino, DS6000, DS8000, Enterprise Storage Server, eServer, FICON, FlashCopy, GDPS, HyperSwap, IBM, Lotus, MVS, OS/390, PowerHA, PowerPC, RS/6000, S/390, System z9, System z10, Tivoli, z/OS, z9, z10, z13, z14, z/VM, and z/VSE are registered trademarks or trademarks of International Business Machines Corporation.

Active Directory, ActiveX, Bing, Excel, Hyper-V, Internet Explorer, the Internet Explorer logo, Microsoft, Microsoft Edge, the Microsoft corporate logo, the Microsoft Edge logo, MS-DOS, Outlook, PowerPoint, SharePoint, Silverlight, SmartScreen, SQL Server, Visual Basic, Visual C++, Visual Studio, Windows, the Windows logo, Windows Azure, Windows PowerShell, Windows Server, the Windows start button, and Windows Vista are registered trademarks or trademarks of Microsoft Corporation. Microsoft product screen shots are reprinted with permission from Microsoft Corporation.

All other trademarks, service marks, and company names in this document or website are properties of their respective owners.

Copyright and license information for third-party and open source software used in Hitachi Vantara products can be found in the product documentation, at <https://www.hitachivantara.com/en-us/company/legal.html> or https://knowledge.hitachivantara.com/Documents/Open_Source_Software.

Contents

Preface.....	5
Intended audience.....	5
Product version.....	5
Release notes.....	5
Document conventions.....	5
Conventions for capacity values.....	7
Comments.....	7
Getting help.....	8
Chapter 1: Overview.....	9
About Hitachi Storage Plug-in for Containers.....	9
About the environment setup tasks.....	10
Requirements.....	10
Container orchestrators to be supported.....	10
Server requirements.....	11
Storage requirements.....	11
Network requirements.....	12
Pre-installation tasks.....	12
Server pre-installation.....	13
Multipath settings.....	13
Storage pre-installation for VSP family.....	15
Host group and iSCSI target naming rules.....	16
Storage pre-installation for VSSB.....	17
Multitenancy functionality settings.....	17
Chapter 2: Installation.....	19
Installation on OpenShift.....	19
Installation on Kubernetes.....	20
Configuration of Storage Plug-in for Containers instance.....	21
Environment variables.....	23
Chapter 3: Usage.....	25
Secret settings.....	25
StorageClass settings.....	26
PersistentVolumeClaim settings.....	28

Pod settings.....	29
Command examples.....	30
Volume snapshot.....	33
Volume cloning.....	36
Volume expansion.....	37
Raw block volume.....	38
ReadOnlyMany.....	40
Resource partitioning.....	40
Chapter 4: Technology preview.....	45
Stretched PVC.....	45
Requirements for using the Stretched PVC feature.....	46
Creating a Stretched PVC.....	49
Checking information on Stretched PVCs.....	51
Deleting a Stretched PVC.....	52
Forcibly deleting a Stretched PVC.....	52
Connecting a Stretched PVC to a Pod.....	52
Failover.....	53
Troubleshooting.....	53
Chapter 5: Upgrade.....	54
Upgrade on OpenShift.....	54
Upgrade on Kubernetes.....	54
Chapter 6: Uninstallation.....	55
Uninstallation on OpenShift.....	55
Uninstallation on Kubernetes.....	55
Chapter 7: Troubleshooting.....	56
Collecting information for troubleshooting.....	56
Information needed when contacting support.....	56
Collecting logs for Storage Plug-in for Containers.....	57
Collecting storage system information for VSP family.....	57
Collecting storage system information for VSSB.....	58
Viewing the volume properties of PersistentVolume.....	58
Node failures.....	59
Initial setup for Fibre Channel environment.....	59
Creating and deleting PersistentVolumeClaim simultaneously	59
Host group settings.....	60

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.

Please read this document carefully to understand how to install and use the plug-in, and maintain a copy for your reference.

Intended audience

This document is intended for system administrators, Hitachi Vantara representatives, and authorized service providers who install, configure, and run Hitachi Storage Plug-in for Containers.

Readers of this document should be familiar with the following:

- Containerized environments and their basic functions
- Hitachi storage systems

Product version

This document applies to Hitachi Storage Plug-in for Containers version 3.11.0.

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 Support Website: <https://knowledge.hitachivantara.com/Documents>.


Related documents for storage adapters and drivers are available at https://knowledge.hitachivantara.com/Documents/Adapters_and_Drivers/Storage_Adapters_and_Drivers.






Document conventions

This document uses the following typographic conventions:

Convention	Description
Bold	<ul style="list-style-type: none"> Indicates text in a window, including window titles, menus, menu options, buttons, fields, and labels. Example: Click OK. Indicates emphasized words in list items.
<i>Italic</i>	<ul style="list-style-type: none"> Indicates a document title or emphasized words in text. Indicates a variable, which is a placeholder for actual text provided by the user or for output by the system. Example: <pre>pairedisplay -g group</pre> (For exceptions to this convention for variables, see the entry for angle brackets.)
Monospace	Indicates text that is displayed on screen or entered by the user. Example: <code>pairedisplay -g oradb</code>
< > angle brackets	<p>Indicates variables in the following scenarios:</p> <ul style="list-style-type: none"> Variables are not clearly separated from the surrounding text or from other variables. Example: <pre>Status-<report-name><file-version>.csv</pre> Variables in headings.
[] square brackets	Indicates optional values. Example: [a b] indicates that you can choose a, b, or nothing.
{ } braces	Indicates required or expected values. Example: { a b } indicates that you must choose either a or b.
vertical bar	<p>Indicates that you have a choice between two or more options or arguments. Examples:</p> <p>[a b] indicates that you can choose a, b, or nothing.</p> <p>{ a b } indicates that you must choose either a or b.</p>

This document uses the following icons to draw attention to information:

Icon	Label	Description
	Note	Calls attention to additional information.

Icon	Label	Description
	Tip	Provides helpful information, guidelines, or suggestions for performing tasks more effectively.
	Important	Highlights information that is essential to the completion of a task.
	Caution	Warns the user of adverse conditions and/or consequences (for example, disruptive operations, data loss, or a system crash).
	CAUTION	Warns the user of a hazardous situation that, if not avoided, could result in major or minor injury.
	WARNING	Warns the user of a hazardous situation which, if not avoided, could result in death or serious injury.

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

Comments

Please send comments to doc.comments@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!

Getting help

The [Hitachi Vantara Support Website](https://support.hitachivantara.com/en_us/contact-us.html) 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 contact information: https://support.hitachivantara.com/en_us/contact-us.html.

[Hitachi Vantara Community](https://community.hitachivantara.com) 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.

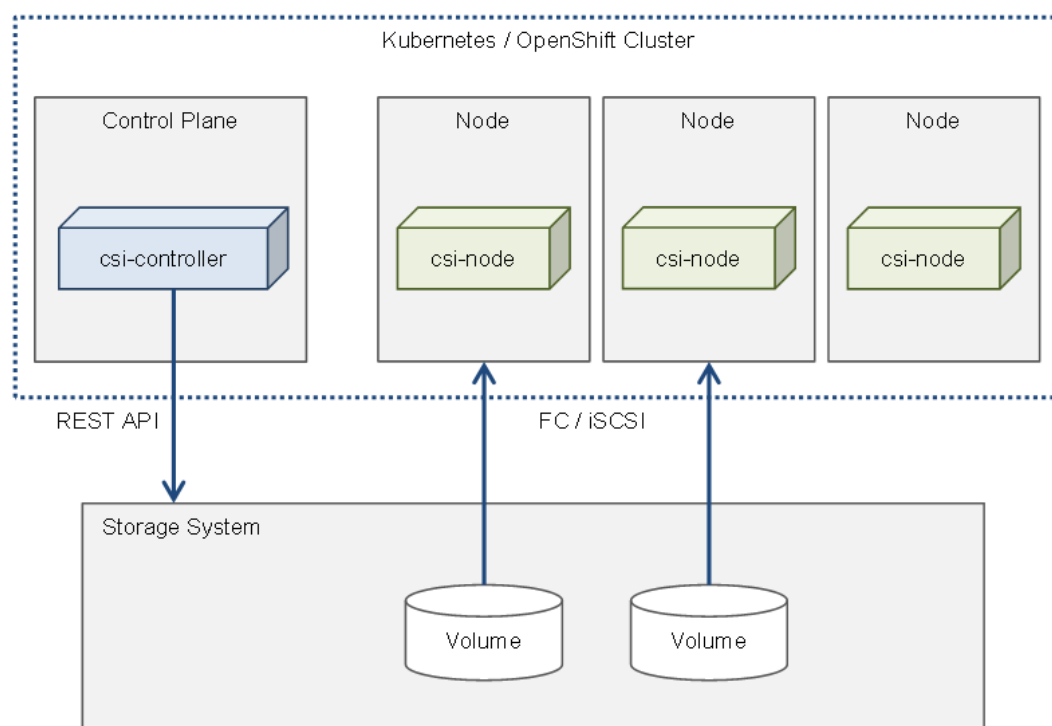
Chapter 1: Overview

Storage Plug-in for Containers is a software component that contains libraries, settings, and commands that you can use to create a container in order to run your stateful applications. It enables the stateful applications to persist and maintain data after the life cycle of the container has ended. Storage Plug-in for Containers provides persistent volumes from Hitachi storage systems.

About Hitachi Storage Plug-in for Containers

Storage Plug-in for Containers integrates Kubernetes or OpenShift with Hitachi storage systems 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 is mainly for storage operations. This is deployed as Deployment and runs only on a control plane.
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.

About the environment setup tasks

Storage Plug-in for Containers enables dynamic operation of storage systems when containers are used. In order to use Storage Plug-in for Containers pre-installation tasks must be completed.

Procedure

1. 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, and OpenShift.
2. Execute pre-installation tasks.
 - a. Set up the Kubernetes and OpenShift environment.
 - b. Configure the Hitachi storage systems.
3. Install Storage Plug-in for Containers.

Requirements

Before you install Storage Plug-in for Containers, check that your server and storage meet the following minimum requirements.


Container orchestrators to be supported

Container orchestrator	Remarks
Red Hat OpenShift Container Platform	—
Kubernetes	—

Container orchestrator	Remarks
Rancher Kubernetes Engine 1 (RKE1) Rancher Kubernetes Engine 2 (RKE2)	If you use RKE1 or RKE2, in this manual, read "Kubernetes" as "RKE1" or "RKE2" and act accordingly.

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. <div>  Note: You can also use Red Hat Enterprise Linux CoreOS as a worker node of OpenShift environments. </div>

Storage requirements

Storage requirements for Virtual Storage Platform (VSP) family.

Component	Requirement
Model	Refer to the release notes for details.
SVOS	Refer to the release notes for details.
Interface	Fibre Channel and iSCSI for bare metal servers. iSCSI for vSphere virtual machines.
Host group	Must be dedicated to Storage Plug-in for Containers. Do not use a host group used for Storage Plug-in for Containers for purposes other than 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.

Component	Requirement
License	<ul style="list-style-type: none"> ▪ Hitachi Dynamic Provisioning (HDP), required. ▪ Hitachi Thin Image (HTI), required.
SVP	Single and dual SVP configurations are supported.

Storage requirements for Virtual Storage Software block (VSSB).

Component	Requirement
Version	Refer to the release notes for details.
Interface	Fibre Channel and iSCSI for bare metal servers. iSCSI for vSphere virtual machines.
User account	<ul style="list-style-type: none"> ▪ 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 17) and set a user.

Network requirements

Storage Plug-in for Containers uses the following ports. Use this information for reference when configuring the firewall.

Component	Port	Usage	Remarks
Storage	80 or 443	REST API connection	None



Pre-installation tasks

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

Server pre-installation

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

If you are using VSSB, for information on the server settings, see the *Hitachi Virtual Storage Software Block Storage Administrator Guide*.

Component	Tasks
Hypervisor	<p>If you want to use virtual machines, setup the hypervisor.</p> <div>  Note: Storage Plug-in for Containers is tested with VMware vSphere 7.0. </div>
Fibre Channel	Verify if Fibre Channel HBA is installed on all hosts.
iSCSI	<p>Verify if iSCSI initiator software is installed on all hosts. If not, refer to: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/storage_administration_guide/osm-create-iscsi-initiator</p> <div>  Note: <p>Storage Plug-in for Containers does not support the following items:</p> <ul style="list-style-type: none"> ▪ IQN including upper-case letters. ▪ IPv6 (same for components other than iSCSI). </div>
multipathd	See Multipath settings (on page 13) .

Multipath settings

Enable multipathd 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 VSSB, for information on VSSB-specific settings, see the sections describing the operating environment settings and the ALUA settings in the *Hitachi Virtual Storage Software Block Operation Guide*.



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
- 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/post_installation_configuration/machine-configuration-tasks.html

The following is an example:

Procedure

1. Obtain the base64 contents from the multipath.conf file.

```
# echo 'defaults {
user_friendly_names yes
find_multipaths yes
}
blacklist {
}' | base64 -w0
```

2. Specify the base64 contents for `spec.config.storage.files.contents.source` in the MachineConfig YAML file that is provided as a sample.

```
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,
ZGVmYXVsdHMgewp1c2VyX2ZyaWVuZGx5X25hbWVzIH11cwpmaW5kX211bHRpcGF0aHMgeWVzCn0KCmJsY
WNrbG1zdCB7Cn0K
          verification: {}
        filesystem: root
        mode: 400
        path: /etc/multipath.conf
```

3. Run the following command:


```
# oc create -f <MachineConfig YAML file>
```




Note: MachineConfig applies to compute nodes only. After MachineConfig is created, all compute nodes are automatically restarted one by one.

Storage pre-installation for VSP family

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

Component	Task
Program products	<ul style="list-style-type: none"> Enable Hitachi Dynamic Provisioning (HDP) license Enable Hitachi Thin Image (HTI) license
Pool	<p>Create an HDP pool</p> <p>Hitachi Dynamic Tiering is not supported.</p>
Fibre Channel connection	<p>Use a Fibre Channel switch for communication between the storage and servers. Set the following parameters for storage ports using Storage Navigator:</p> <ul style="list-style-type: none"> Connection Type: P-to-P Fabric: ON Security: Enabled <p>Host groups required for Storage Plug-in for Containers are automatically created by Storage Plug-in for Containers.</p> <p>If you want to use existing host groups, rename them according to the naming rule. For details, see Host group and iSCSI target naming rules (on page 16).</p> <p>To the host group, add a WWN for each host that joins the Kubernetes cluster. For example, if you have 10 hosts in your Kubernetes cluster, you must prepare 10 host groups.</p> <div>  <p>Note: Storage Plug-in for Containers will overwrite host mode options even if existing host groups have other host mode options.</p> </div>
iSCSI connection	Enable port security by Storage Navigator.

Component	Task
	<p>Storage Plug-in for Containers automatically performs the following actions:</p> <ul style="list-style-type: none"> Creates iSCSI targets for each host if there is no iSCSI target. <p>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 16)).</p> <ul style="list-style-type: none"> Adds an IQN to the iSCSI target of each host that will join the Kubernetes cluster. Logs in to the iSCSI target on each host. <p>If you want to use CHAP, do the following:</p> <ul style="list-style-type: none"> Create an iSCSI target (see Host group and iSCSI target naming rules (on page 16)). Set CHAP for the port and iSCSI target. Log in to the iSCSI target with CHAP authentication. <div>  Note: Storage Plug-in for Containers will overwrite host mode options even if existing iSCSI targets have other host mode options. </div>

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 VSSB

Component	Task
Fibre Channel connection	Use the Fibre Channel switch for communication between storage and servers.
Server resource	<p>If you have already created a Server resource in VSSB, verify the following:</p> <p>Storage Plug-in for Containers automatically performs the following actions:</p> <ul style="list-style-type: none"> ▪ Finds an existing Server resource with the host WWN or IQN configured. ▪ If an existing Server resource is not found, it creates a new Server resource and configures it with a host WWN or IQN. ▪ Verifies and configures the Server resource to connect to the all compute ports. <p>If you want to use CHAP, do the following:</p> <ul style="list-style-type: none"> ▪ Set CHAP for the compute port. ▪ Log in to the iSCSI target with CHAP authentication. <div>  Note: Do not add a WWN or IQN of multiple hosts to the same Server resource. Only one Server resource can be associated with a single host. </div>

Multitenancy functionality settings

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



Note: A VSSB used in one Kubernetes/OpenShift cluster 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/installing/disconnected_install/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: **A specific namespace on the cluster** and `<any namespace>`
- Update approval: **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 \(on page 21\)](#).

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

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

Installation on Kubernetes

For Kubernetes, you can install Storage Plug-in for Containers using Operator. Perform the following steps to install Storage Plug-in for Containers and to get container images from Red Hat registry.



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

Procedure

1. Extract the Storage Plug-in for Containers package and move to the directory **yaml/operator**.
2. Create the namespace for the Operator:

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

3. Create the Secret for Red Hat registry. Create it with the `regcred-redhat-com` Secret name. (see <https://kubernetes.io/docs/tasks/configure-pod-container/pull-image-private-registry/#create-a-secret-by-providing-credentials-on-the-command-line>).

For example:

```
# kubectl create secret docker-registry regcred-redhat-com \
--namespace=hspc-operator-system \
--docker-server=registry.connect.redhat.com \
--docker-username=<user> \
--docker-password=<password>
```

4. 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
```

5. Create two Secrets for Red Hat registry. Create them with the `regcred-redhat-com` and `regcred-redhat-io` Secret names.

For example:

```
# SPC_NAMESPACE=<any namespace>

# kubectl create secret docker-registry regcred-redhat-com \
```

```
--namespace=${SPC_NAMESPACE} \
--docker-server=registry.connect.redhat.com \
--docker-username=<user> \
--docker-password=<password>

# kubectl create secret docker-registry regcred-redhat-io \
--namespace=${SPC_NAMESPACE} \
--docker-server=registry.redhat.io \
--docker-username=<user> \
--docker-password=<password>
```

6. Modify the namespace if you specified a namespace other than kube-system for **SPC_NAMESPACE**:

```
apiVersion: csi.hitachi.com/v1
kind: HSPC
metadata:
  name: hspc
  namespace: <SPC_NAMESPACE>
spec:
  imagePullSecrets:
    - regcred-redhat-com
    - regcred-redhat-io
```

If you want to make an advanced configuration, refer to [Configuration of Storage Plug-in for Containers instance \(on page 21\)](#).

7. Deploy Storage Plug-in for Containers and confirm the status READY is **true** using the following command:

```
# kubectl create -f hspc_v1_hspc.yaml

# kubectl get hspc -n ${SPC_NAMESPACE}
NAME READY AGE
hspc true 30s
```

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:

Parameters	Description
spec.imagePullSecrets	ImagePullSecrets is for pulling images from Red Hat registries.

Parameters	Description
<code>spec.controller.containers.name</code>	<p>Name of the Storage Plug-in for Containers that you want to configure in <code>hspc-csi-controller</code> pods.</p> <p>For example, <code>hspc-csi-driver</code>, <code>csi-provisioner</code>, and so on are the key to the container name inside the <code>hspc-csi-controller</code>.</p> <p>The <code>kubectl describe deployment hspc-csi-controller -n <SPC_NAMESPACE></code> command is used to get the container names.</p>
<code>spec.controller.containers.image</code>	The image name of <code>hspc-csi-controller</code> .
<code>spec.controller.containers.imagePullPolicy</code>	The image pull policy of <code>hspc-csi-controller</code> . The default value is <code>IfNotPresent</code> .
<code>spec.controller.containers.env</code>	List of environment variables to set in <code>hspc-csi-controller</code> container. Refer to Environment variables (on page 23) .
<code>spec.controller.containers.args</code>	Arguments to the entry point for <code>hspc-csi-controller</code> . This replaces all parameters at <code>spec.template.spec.containers.args</code> in a deployment of the container <code>hspc-csi-controller</code> .
<code>spec.node.containers.name</code>	<p>Name of the container that you want to configure in <code>hspc-csi-node</code> pods.</p> <p>For example, <code>hspc-csi-driver</code>, <code>liveness-probe</code>, and so on are the key to the container name inside <code>hspc-csi-node</code>.</p> <p>The <code>kubectl describe daemonset hspc-csi-node -n <SPC_NAMESPACE></code> command is used to get the container names.</p>
<code>spec.node.containers.image</code>	The image name of <code>hspc-csi-node</code> .
<code>spec.node.containers.imagePullPolicy</code>	The image pull policy of <code>hspc-csi-node</code> . The default value is <code>IfNotPresent</code> .

Parameters	Description
<code>spec.node.containers.env</code>	List of environment variables to set in <code>hspc-csi-node</code> container.
<code>spec.node.containers.args</code>	Arguments to the entry point for <code>hspc-csi-node</code> . This replaces all parameters at <code>spec.template.spec.containers.args</code> in a deployment of the container <code>hspc-csi-node</code> .
<code>spec.node.affinity.nodeAffinity</code>	Scheduling rule for Node Affinity of the Pod that runs <code>csi-node</code> . 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/ .

Environment variables

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

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

The following is an example to enable certificate verification of the `hspc-csi-driver`.

1. Check the current settings using the following command:

```
# kubectl get deployment -n <SPC_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 the parameter `env: SPC_VERIFY_CERTIFICATE` to Storage Plug-in for Containers manifests.

```

apiVersion: csi.hitachi.com/v1
kind: HSPC
metadata:
  name: hspc
  namespace: <SPC_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 \(on page 19\)](#) and [Uninstallation \(on page 55\)](#).
4. Check the changes.

```

# kubectl get deployment -n <SPC_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.

Secret settings

The Secret file contains the storage URL, user name, and password 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)
```

Legend:

(1) Secret name

(2) base64-encoded storage URL.

Use the IP address of the SVP for the following: VSP 5000 series, VSP F400, F600, F800, VSP F1500, VSP G200, G400, G600, G800, VSP G1000, VSP G1500, and VSP N400, N600, N800. 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.

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
```

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

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.
- (5) Connection type between storage and nodes. `fc` and `iscsi` are supported. If blank, `fc` is set.
- (6) Activation of adaptive data reduction. `"Compression"`, `"CompressionDeduplication"`, and `"Disabled"` are supported. If blank, `Disabled` is set. 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.
- (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 blank, the default value is set. The default value depends on the storage system. 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.
- Adaptive data reduction cannot be used together with the Stretched PVC function.

- (8) Filesystem type. `ext4` and `xfs` are supported. If blank, `ext4` is set.
- (9) Secret name
- (10) Secret namespace

Storage Class for VSSB

Parameter references for sc-sample.yaml

```

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

```

Legend:

- (1) StorageClass name
- (2) Storage type. This field must be set to "vssb" when using VSSB.
- (3) Connection type between storage and nodes. `fc` and `iscsi` are supported. If blank, `fc` is set.
- (4) Filesystem type. `ext4` and `xf`s are supported. If blank, `ext4` is set.
- (5) Secret name
- (6) Secret namespace

PersistentVolumeClaim settings

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.

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`. To use `ReadOnlyMany`, see [ReadOnlyMany \(on page 40\)](#).

(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 command `kubectl delete pvc <PVC_NAME>`.
- 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 \(on page 58\)](#) and obtain the volume ID of the storage. Delete the PersistentVolume using the command `kubectl delete pv <PV_NAME>`. 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-cf8c6089-0386-4c39-8037-e1520a986a7d  1Gi       RWO           Delete
Bound   default/pvc-sample  sc-sample                                35s

# kubectl describe pv pvc-cf8c6089-0386-4c39-8037-e1520a986a7d
Name:          pvc-cf8c6089-0386-4c39-8037-e1520a986a7d
Labels:        <none>
Annotations:   pv.kubernetes.io/provisioned-by: hspc.csi.hitachi.com
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
  VolumeHandle:  60060e8018117a000051117a0000053f--spc-c44a7dc0f5
  ReadOnly:      false
  VolumeAttributes: autoHG=true
                  connectionType=fc
                  hostModeOption=91
                  ldevIDDec=1343
                  ldevIDHex=05:3F
                  nickname=spc-c44a7dc0f5
                  ports=CL1-A,CL2-A
                  size=1Gi
                  storage.kubernetes.io/csiProvisionerIdentity=1585728584906-
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
```

Volume snapshot

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

**Note:**

- If the volume is expanded, confirm for completion before executing this feature. See [Volume expansion \(on page 37\)](#) for more details.
- Flush the data before creating a snapshot for data consistency. For example, temporarily remove the pod.
- This feature is not supported in VSSB.

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. For Snapshot Controller, use 4.x.x.



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)
  csi.storage.k8s.io/snapshotter-secret-name: "secret-sample" # (3)
  csi.storage.k8s.io/snapshotter-secret-namespace: "default" # (4)
```

Legend:

- (1) VolumeSnapshotClass name
- (2) Same poolID as the StorageClass
- (3) Same Secret name as the StorageClass
- (4) 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

```

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 command `kubectl get pv <PV_NAME> -o yaml`, which is displayed in the parameter **size**.



Note: If the volume is expanded, obtain the size by using the command `kubectl get pv <PV_NAME>`, which is displayed in the parameter **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
```

- Confirm the completion of creating VolumeSnapshot by verifying that the parameter `readyToUse` is true:

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



Note: If the parameter `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
```

Volume cloning

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. It is different from a standard volume in that the backend device creates an exact duplicate of the specified volume at provisioning.



Note:

- If the volume is expanded, confirm for completion before executing this feature. Refer to [Volume expansion \(on page 37\)](#) for details.
- Flush the data before cloning for data consistency. For example, temporarily remove the pod.
- This feature is not supported in VSSB.

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 command `kubectl get pv <PV_NAME> -o yaml`, which is displayed in the parameter **size**.



Note: If the volume is expanded, obtain the size by using the command `kubectl get pv <PV_NAME>`, which is displayed in the parameter **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
```

Volume expansion

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 command `kubectl get pvc`, which is displayed in the parameter **CAPACITY**. Do not shut down the OS or drain the node before volume expansion completes.

Before you begin

This feature requires the following resources:

- StorageClass
- PersistentVolumeClaim



Note: Volume expansion has the following restrictions:

- 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, execute the command again.
- Volume capacity cannot be reduced.
- The PersistentVolume created by the StorageClass without parameters for volume expansion cannot be expanded.
- The size obtained by the command `kubectl get pv <PV_NAME> -o yaml` is not updated after the volume is expanded. If the volume is expanded, obtain the size by using the command `kubectl get pv <PV_NAME>`, which is displayed in the parameter **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 with the parameter 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: If you want to change the size of an LDEV created with Storage Plug-in for Containers, use the `kubectl patch pvc` command.

Raw block volume

Kubernetes supports 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
volumeMode: Block
resources:
  requests:
    storage: 1Gi      # (2)
  storageClassName: sc-sample    # (3)

```

Legend:

(1) PersistentVolumeClaim name

(2) Volume size

(3) 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
```

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.

Use the PersistentVolumeClaim manifest file used in the [Volume cloning \(on page 36\)](#) section and specify ReadOnlyMany, as shown in the following example.



Note: This feature is not supported in VSSB.

```
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.

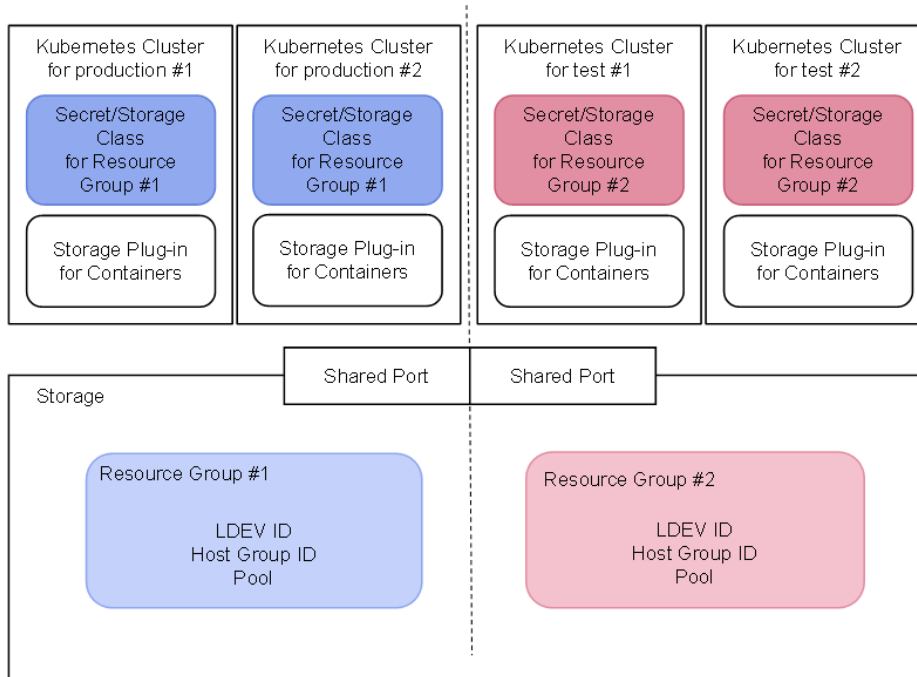


Note: Resource partitioning is not supported for VSSB.

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.



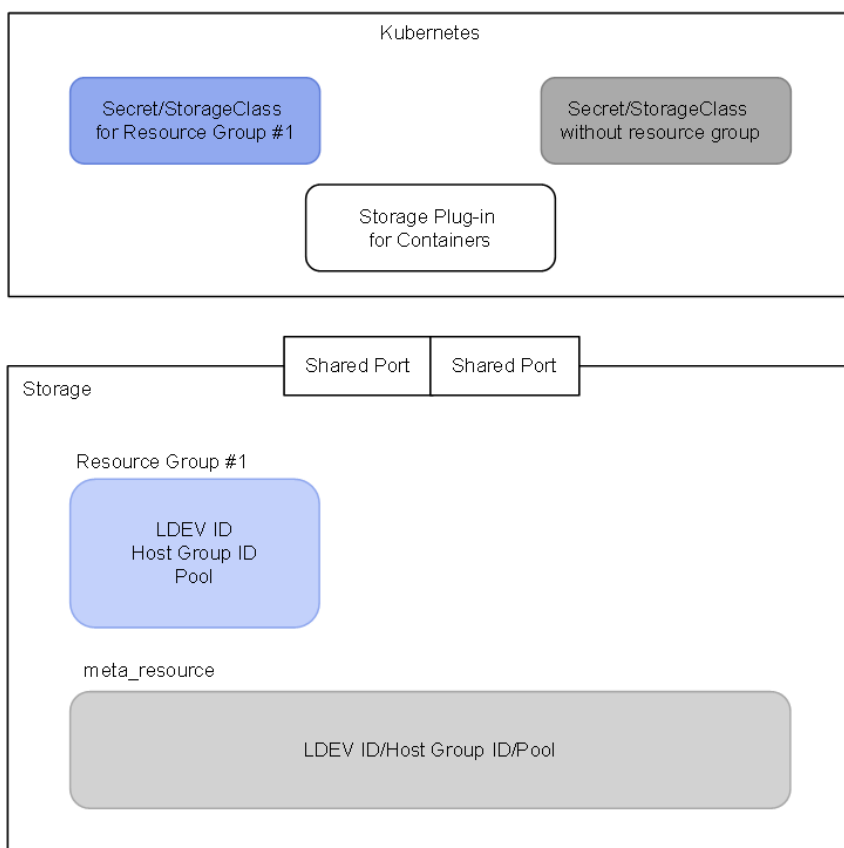
Unsupported configurations

The following are examples of configurations that are not supported.

Example 1

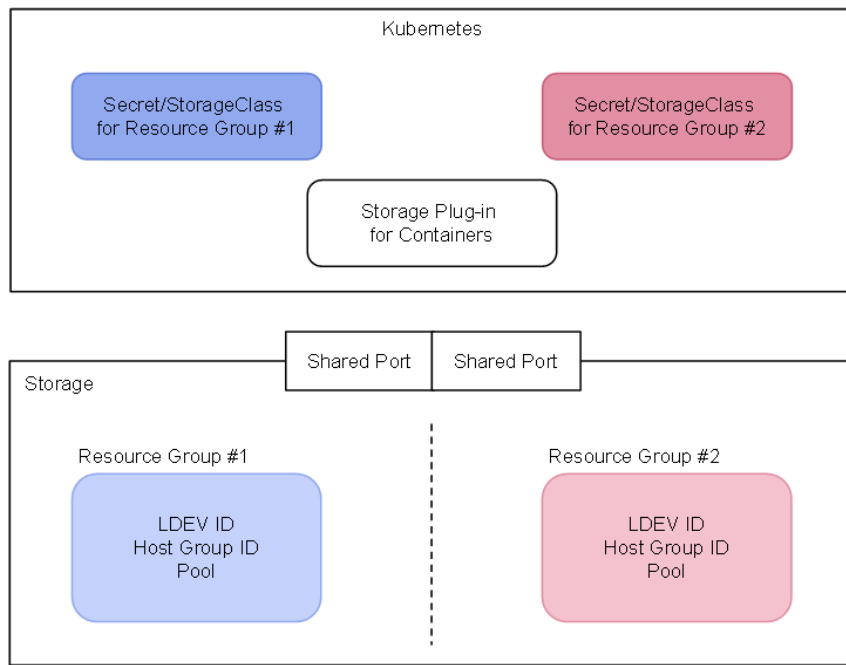
You cannot include both the following configurations in the same Kubernetes cluster.

- 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. Virtual storage machines are not supported.
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 pool from pool volumes with the resource group that you have created.
LDEV	Allocate the necessary number of undefined LDEV IDs to the resource group.
Host Group	Allocate the necessary number of undefined host group IDs to the resource group for each storage system port defined in StorageClass. The number of host group IDs must be equal to the number of hosts for all ports.

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

```

StorageClass settings

If you use iSCSI as a storage system connection, specify the port IP address in number order. If you use 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.
<...>

```

Chapter 4: Technology preview

This chapter describes a Technology Preview feature that is included with the software code.

Use of this feature is only recommended in a test environment as it is not covered under any Hitachi Vantara LLC support plans, may not be functionally complete, and is not intended for production use.

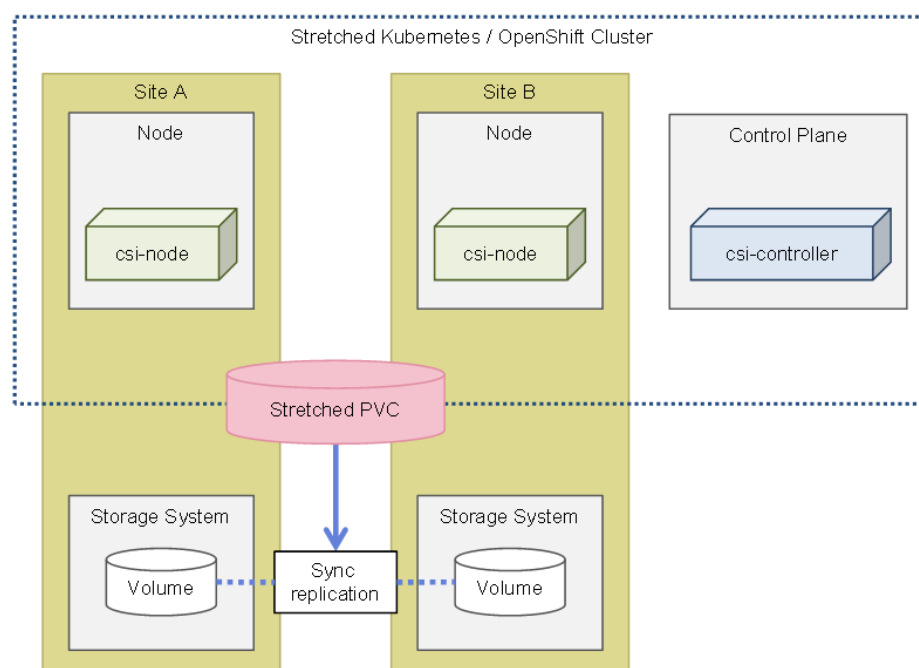
For full information about Technology Preview, review the release notes.

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

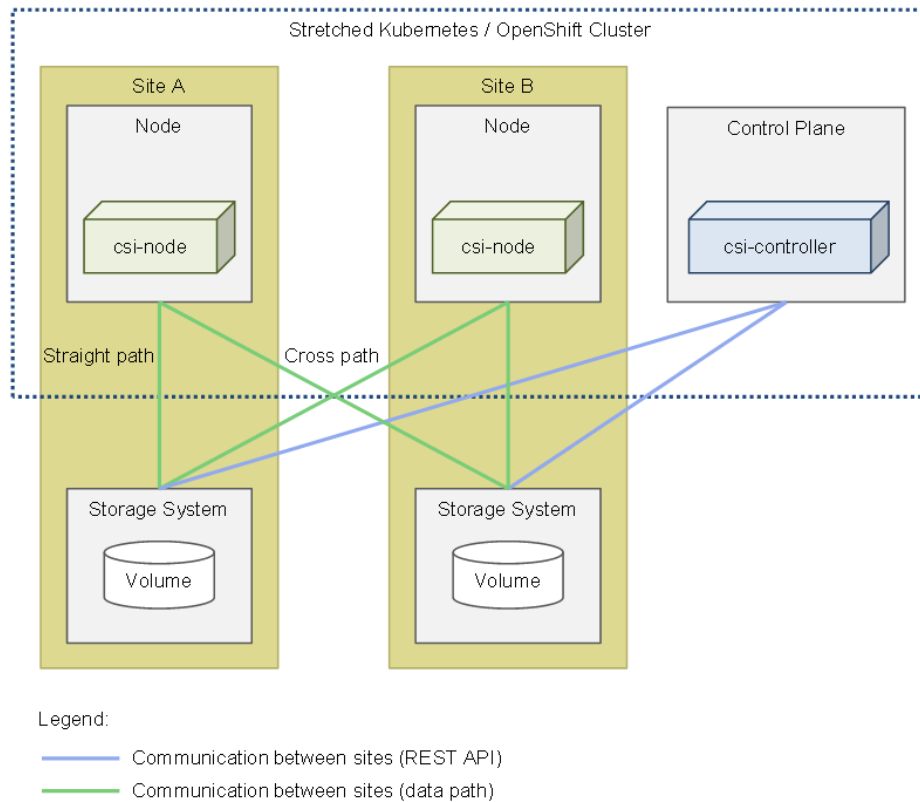
The Stretched PVC feature automates replication provisioning. Note that this feature includes neither the management and monitoring of the replication status, nor the management and monitoring of the status of the primary and secondary sites.

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.
Communication between sites (REST API)	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.
Communication between sites (data path)	Specify settings so that each node can perform data communication with the storage systems at the primary site and at the secondary site. Only FC 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	<p>The following storage systems are supported:</p> <ul style="list-style-type: none"> VSP E series VSP 5000 series VSP F350, F370, F700, F900 VSP G350, G370, G700, G900
License	<p>Apply the required license by referring to the <i>Global-Active Device User Guide</i>.</p>

Item	Description
Virtual storage machines	<p>For each storage system, create one virtual storage machine. The virtual storage machine must meet the following requirements:</p> <ul style="list-style-type: none"> 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. <p>Note the following when using the same combination of storage systems from multiple clusters:</p> <ul style="list-style-type: none"> 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 that has access permissions only for the resource group of the created virtual storage machine. In addition, create users to which only the created user group is assigned.
Pool	Create a pool from the pool volumes in the created virtual storage machines.
LDEV	Assign the required number of unused LDEV IDs to the virtual storage machines. For the assigned LDEVs, change the virtual LDEV ID to unassigned (65534).
Host groups	For each port that you want to use, assign the required number of unused host group IDs for the virtual storage machines. 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 \(on page 40\)](#). Note the items listed in the following table when specifying settings in advance:

Item	Description
Resource groups	Create regular resource groups instead of virtual storage machines.
Host groups	Use 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	No
Volume cloning	No
Volume expansion	No
Raw block volume	Yes
ReadOnlyMany	No
Adaptive data reduction	No



Note: Do not use the Stretched PVC feature and Hitachi Replication Plug-in for Containers together in the same Kubernetes cluster.

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, the 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 are to 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) 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) 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: false
parameters:
  connectionType: fc
  replicationType: stretched                # (1)
  quorumID: "30"                           # (2)
  primaryPoolID: "10"                      # (3)
  primaryPortID: CL1-A,CL2-A               # (4)
  secondaryPoolID: "20"                   # (5)
  secondaryPortID: CL1-F                   # (6)
  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"

```

Legend:

- (1) Parameter indicating that the replication type is Stretched PVC
 - (2) ID of the Quorum disk
 - (3) Pool ID of the storage system at the primary site
 - (4) Port ID of the storage system at the primary site
 - (5) Pool ID of the storage system at the secondary site
 - (6) 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. Therefore, by running `kubectl describe pv <pv-name>` and checking the `spec.csi.volumeAttributes` field, you can check the resources on the storage systems created by Storage Plug-in for Containers. You can also use this information for troubleshooting.

The following table lists the items displayed in `spec.csi.volumeAttributes`.

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

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 at 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, take action in accordance with the error message. If the problem persists, see [Forcibly deleting a Stretched PVC \(on page 52\)](#).

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 delete a Stretched PVC in this situation, perform the following steps:

Procedure

1. Check information about the resources created at the primary and secondary sites by referring to [Checking information on Stretched PVCs \(on page 51\)](#).
2. Delete the PVC and 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).

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]
```

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.

Troubleshooting

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}]
```

Chapter 5: 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.

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 \(on page 19\)](#).

Upgrade on Kubernetes

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

Procedure

1. Delete the 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 the new Storage Plug-in for Containers. See the topic [Installation on Kubernetes \(on page 20\)](#).

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 PersistentVolumeClaim, the StorageClass and the Secret related 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 PersistentVolumeClaim, the StorageClass, and the Secret related to Storage Plug-in for Containers.
3. Delete the Storage Plug-in for Containers and resource for the Operator:

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

4. Delete the Secret of your Docker credentials.

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
Cluster information	Run the following command: <pre># kubectl cluster-info dump -A > dump.txt</pre>
Operator logs	See Collecting logs for Storage Plug-in for Containers (on page 57) .
CSI controller logs	See Collecting logs for Storage Plug-in for Containers (on page 57) .
CSI node logs	See Collecting logs for Storage Plug-in for Containers (on page 57) .
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 Volume snapshot (on page 33) chapter.
Application manifests	Get the YAML files for applications that uses Storage Plug-in for Containers PVCs.

Information	Procedure
Storage logs	See Collecting storage system information for VSP family (on page 57) or Collecting storage system information for VSSB (on page 58) .

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/>

To retrieve logs from the operator, run the following command:

```
# kubectl logs -n <namespace> hspc-operator-controller-manager-<id>
```

To retrieve logs from the CSI controller, run the following command:

```
# kubectl logs -n <namespace> hspc-csi-controller-<id> -c <container name>
```



Note:

- Container `hspc-csi-driver` is the primary process.
- There are several sidecars for this Pod: `csi-provisioner`, `external-attacher`, `csi-resizer`, `csi-snapshotter`, and `liveness-probe`.

To retrieve logs from the CSI node, run the following command:

```
# kubectl logs -n <namespace> hspc-csi-node-<id> -c <container name>
```



Note:

- Container `hspc-csi-driver` is the primary process.
- There are several sidecars for this Pod: `driver-registrar`, `liveness-probe`
- You will see multiple CSI node Pods since this is deployed as a DaemonSet. Collect logs from all these Pods.

Collecting storage system information for VSP family

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 VSSB


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

Viewing the volume properties of PersistentVolume


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

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

Volume properties for VSP family.

Property	Description
ldevIDDec	LDEV ID.
ldevIDHex	Hexadecimal LDEV ID.
size	Capacity of the volume. <div>  Note: Capacity shown here is the original capacity used when creating the volume. </div>

Volume properties for VSSB.

Property	Description
volumeID	ID of the volume created in VSSB.
size	Capacity of the volume. <div>  Note: Capacity shown here is the original capacity used when creating the volume. </div>

Node failures

When a node failure occurs and the nodes leave a cluster, reboot the operating system of the nodes to clear unnecessary files, such as unpointed device files, before the nodes rejoin the cluster.

Initial setup for Fibre Channel environment

If you encounter an error code `0x0000c008` during the Pod creation phase, and if it is the first time after the initial setup of the Fibre Channel environment, consider deleting the Pod and rebooting the host to solve the issue.

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 100.

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 \(on page 21\)](#).

```
apiVersion: csi.hitachi.com/v1
kind: HSPC
metadata:
  name: hspc
  namespace: <SPC_NAMESPACE>
spec:
  imagePullSecrets:
    - regcred-redhat-com
    - regcred-redhat-io
  controller:
    containers:
      - name: csi-provisioner
        args:
          - --csi-address=/csi/csi-controller.sock
          - --timeout=300s
          - --v=5
          - --worker-threads=10
```

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 \(on page 16\)](#)). 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. Stop Storage Plug-in for Containers.
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. Start Storage Plug-in for Containers.



Note: These settings are not applicable for VSSB.

Hitachi Vantara

Corporate Headquarters
2535 Augustine Drive
Santa Clara, CA 95054 USA



HitachiVantara.com/contact