

# Hitachi NAS CSI Driver

v1.4.2

---

## User Guide

This document describes the usage, requirements, installation and configuration of the Hitachi NAS CSI driver on both Kubernetes and OpenShift systems. An explanation of the parameters used in the configuration files is also provided.

Direct deployment and deployment using an Operator are both described.

© 2024 Hitachi Vantara LLC. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, or stored in a database or retrieval system for commercial purposes without the express written permission of Hitachi, Ltd., or Hitachi Vantara Corporation (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 Corporation at [https://support.HitachiVantara.com/en\\_us/contact-us.html](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 Corporation.

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 to access relevant data; and
- 2) Verifying that 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.

**EXPORT CONTROLS** - Licensee will comply fully with all applicable export laws and regulations of the United States and other countries, and Licensee shall not export, or allow the export or re-export of, the Software, API, or Materials in violation of any such laws or regulations. By downloading or using the Software, API, or Materials, Licensee agrees to the foregoing and represents and warrants that Licensee is not located in, under the control of, or a national or resident of any embargoed or restricted country.

Hitachi is a registered trademark of Hitachi, Ltd., in the United States and other countries.

AIX, AS/400e, DB2, Domino, DS6000, DS8000, Enterprise Storage Server, eServer, FICON, Flash Copy, IBM, Lotus, MVS, OS/390, PowerPC, RS6000, S/390, System z9, System z10, Tivoli, z/OS, z9, z10, z13, z/VM, BCPI™ 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, the Microsoft Corporate 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 web site are properties of their respective owners.

# Table of Contents

<b>Table of Contents</b> .....	<b>3</b>
<b>Preface</b> .....	<b>5</b>
About this document.....	5
Document conventions .....	5
Intended audience .....	5
Accessing product downloads.....	5
Getting Help .....	6
Comments.....	6
<b>Chapter 1: Introduction</b> .....	<b>7</b>
Container Storage Interface (CSI).....	7
Overview of the Hitachi NAS CSI driver .....	7
How the Driver works .....	7
<b>Chapter 2: System Requirements</b> .....	<b>9</b>
Storage Platforms.....	9
Port Requirements.....	9
Hitachi NAS License Requirements .....	9
Kubernetes Requirements .....	10
<b>Chapter 3: Installation and Configuration of the Driver</b> .....	<b>11</b>
Install using the Deployment File .....	13
Install using the Operator.....	13
Installing on Red Hat OpenShift.....	14
Check Installed Components .....	15
Configuring the Hitachi NAS System.....	15
Configuring the Hitachi NAS CSI Driver .....	16
Static Volumes.....	17
Dynamic Volumes.....	17
Limitations .....	18
<b>Chapter 4: Using The Driver</b> .....	<b>19</b>
Create a New Static Volume.....	19
Create a New Dynamic Volume .....	19
Expand an Existing Dynamic Volume.....	19
Delete a Dynamic Volume .....	20
Using a Volume .....	20

Creating a Volume Snapshot .....	20
Deleting a Volume Snapshot.....	20
Creating a Volume from an Existing Source .....	21
<b>Chapter 5: Configuration Files .....</b>	<b>22</b>
Secret Settings .....	22
StorageClass Settings .....	23
VolumeSnapshotClass Settings .....	24
PersistentVolume Settings .....	25
PersistentVolumeClaim Settings .....	26
VolumeSnapshot Settings.....	28
Pod Settings .....	29
<b>Chapter 6: Data Locations and Admin Tasks .....</b>	<b>30</b>
Populating a Volume with Data .....	30
Access Volume Data for Backup.....	30
Access Volume Snapshot Data for Backup .....	30
Manual Deletion of Volume Data.....	31
Update Hitachi NAS Credentials .....	31
Uninstall CSI Driver .....	32
<b>Chapter 7: Troubleshooting .....</b>	<b>33</b>
Obtaining Logs .....	33
Changing the Log Level .....	34
CSI Driver cannot connect to the Hitachi NAS Server .....	34
Kubernetes Node Unable to NFS Mount the Hitachi NAS Server .....	34
Intermittent Connection Failure .....	35
Operation Timeout.....	36
Volume creation/deletion operations not able to complete.....	36
Snapshot creation failing.....	37
<b>Appendix 1: Hitachi NAS Driver Parameters.....</b>	<b>38</b>
<b>Appendix 2: Restricting API Key Access .....</b>	<b>39</b>
<b>Appendix 3: Options Available when Installing with the Operator.....</b>	<b>41</b>

# Preface

## About this document

This document describes how to use the Hitachi NAS CSI driver for Kubernetes, and explains the parameters used in the configuration files.

## Document conventions

This document uses the following typographic convention:

Convention	Description
<b>Bold</b>	<ul style="list-style-type: none"><li>Indicates text in a window, including window titles, menus, menu options, buttons, fields, and labels. Example: Click <b>OK</b>.</li><li>Indicates emphasized words in list items.</li></ul>
<i>Italic</i>	Indicates a document title or emphasized words in text.
Monospace	Indicates text that is displayed on screen or entered by the user. Example: <code>pairdisplay -g oradb</code>

## Intended audience

This document is intended for system administrators, Hitachi Vantara representatives, and authorized service providers who install, configure, and run the Hitachi NAS Container Storage Interface (CSI) driver for Kubernetes.

Readers of this document should be familiar with the following:

- Containerized environments and their basic functions
- Hitachi NAS server platforms (Hitachi NAS platform, NAS module and VSP One File)

## Accessing product downloads

Product software, drivers, and firmware downloads are available on Hitachi Vantara Support Connect: <https://support.hitachivantara.com/>.

Log in and select Product Downloads to access the most current downloads, including updates that may have been made after the release of the product.

## Getting Help

[Hitachi Vantara Support Connect](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 Hitachi Vantara Support Connect 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 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](https://community.hitachivantara.com), register, and complete your profile.

## Comments

Please send comments to [doc.feedback@hitachivantara.com](mailto: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: Introduction

All instructions in this document assume that Kubernetes is already setup and working. For more details about Kubernetes, refer to the following documentation:

<https://kubernetes.io/docs/home/>

It also assumes all prerequisites for allowing CSI drivers to work within your environment have been met before attempting to install the Hitachi NAS driver.

The Hitachi NAS CSI driver is also supported on Red Hat OpenShift. Unless specifically stated, the term Kubernetes used in this document includes Red Hat OpenShift.

## Container Storage Interface (CSI)

The Container Storage Interface (CSI) is a standard for exposing arbitrary block and file storage systems to containerized workloads on Container Orchestration Systems (COs) like Kubernetes. Using CSI, third-party storage providers can write and deploy plugins exposing new storage systems in Kubernetes without ever having to touch the core Kubernetes code. For more details, refer to the following documentation:

<https://kubernetes.io/docs/concepts/storage/volumes/#csi>

## Overview of the Hitachi NAS CSI driver

The Hitachi NAS Container Storage Interface (CSI) driver is a software component that contains libraries, settings, and commands that can be used to create and manage persistent storage for your containers. It enables the stateful applications to persist and maintain data after the life cycle of the container has ended. The Hitachi NAS CSI driver provides persistent volumes on Hitachi NAS storage, and is able to clone those volumes, and take snapshots of them. It supports both static and dynamically provisioned volumes.

As the driver relies on the ability for containers/pods to access Hitachi NAS NFS exports, it can only be used on Linux based systems.

### How the Driver works

Kubernetes supports the hosting of volumes on an external NFS server, but Kubernetes has no ability to manage them. The Hitachi NAS CSI driver allows storage to be dynamically allocated on Hitachi NAS systems directly from Kubernetes, as well as consuming existing storage. It can manage the creation and deletion of NFS exports and virtual volumes and allows existing data already present on the Hitachi NAS to be cloned and presented to multiple different containers/pods. Volumes can also be snapshotted and the data re-used for new volumes or to take backups. If configured to do so, the driver can also delete the volume or snapshot contents when they are no longer needed.

Backup and restore of important data contained within container images can be an issue under normal circumstances. By providing NFS mounted remote storage, it is possible to use

normal backup/restore tools to ensure that the important data associated with the containers can be properly backed up and restored should it be necessary.

The driver does not get involved with the normal operation of the containers, and NFS mounted exports behave in exactly the same way as if they were created and mounted manually. The driver makes it easier to create external storage on the Hitachi NAS systems, without the need to directly interact with the Hitachi NAS management interfaces.

In addition to creating specific NFS exports for use by Kubernetes, the driver performs the NFS mount on behalf of Kubernetes. Specific mount options can be supplied, for example, which NFS version to use. NFSv3 is recommended when mounting volumes, as it is stateless, generally faster than NFSv4, and has less overhead.

# Chapter 2: System Requirements

## Storage Platforms

The Hitachi NAS CSI driver requires access to the storage platform REST API, and can make use of versions 7, 8 or 9 – at least one of the API versions is supported on the following platforms, using the specified software versions or newer. The use of API keys for authentication was added to a later release. To use API keys instead of a username/password combination, make sure that you are using at least the minimum supported software version from the table below:

Hitachi storage platform	Software version	Software version with API key support
Hitachi VSP One File 3x	15.1	15.1
Hitachi NAS Platform 5000	13.3	13.7
Hitachi NAS Platform 4000	13.3	13.7
VSP Nx00 with NAS Module (N400/N600/N800)	83-06-01-x0/00	83-06-08-x0/00
VSP Gx00 with NAS Module (G400/G600/G800)	83-04-61-x0/00	83-05-36-x0/00

The term "Hitachi NAS" used in the context of this document implies the supported Hitachi storage platforms shown in the table above.

**Note:** It is recommended to use the highest API version supported by the Hitachi NAS that will host the volumes, as updated/enhanced functionality is only added to the latest API version. Support for older API versions may be removed in the future.

## Port Requirements

All network communications are performed over TCP port 8444, using HTTPS, between the Kubernetes cluster and the Hitachi NAS system being used to host the storage volumes.

## Hitachi NAS License Requirements

The Hitachi NAS CSI driver makes use of NFS and file cloning functionality within the Hitachi NAS and requires valid license keys for both **NFS** and **File Clone**.

# Kubernetes Requirements

The Hitachi NAS CSI driver can be installed into Kubernetes environments using Kubernetes version v1.28 or higher. It relies on the Volume Snapshot functionality to be able to snapshot and clone volumes.

Refer to <https://kubernetes.io/releases/> for maintained Kubernetes release versions.

The driver must be installed on a system that supports the **x86\_64-v2** processor instruction set. This is a requirement of the base operating system used by the Hitachi NAS driver container. If installing the driver on physical hardware, this should not cause an issue, but if the driver is installed in a virtualized environment, the processor instruction set exposed from the hypervisor may need to be changed from its default if the Hitachi NAS driver fails to start (`hnas-driver`).

Mount Propagation must be enabled on the Kubernetes cluster, otherwise it's not possible for the driver to perform mounts on behalf of pods/containers.

**Note:** Volume Snapshot functionality may need to be installed separately into the Kubernetes environment, depending on the distribution used – this process is not covered in this document.

## Chapter 3: Installation and Configuration of the Driver

The Hitachi NAS CSI driver includes example yaml files that demonstrate how to create each different resource type. There are two ways to install the driver:

- Using a deployment yaml file, which installs all the required components at the time of deployment
- Using an Operator, which manages the lifecycle of the driver with the creation/deletion of an instance of a Kubernetes CustomResource.

The following table summarizes the yaml deployment file supplied:

Deployment file for the Hitachi NAS CSI driver	
File	Details
hnas-csi-k8s.yaml	Contains information that installs the Hitachi NAS CSI driver into the Kubernetes CSI framework, along with the required sidecar containers.  <i>At the time of writing, the versions of Hitachi NAS CSI driver and sidecar containers specified in this file are suitable for installation on all versions of Kubernetes from 1.26 onwards.</i>

**Note:** Under normal circumstances, this file should not be modified - Refer to the *Troubleshooting* section, which outlines how the file can be updated to change log levels, connection/operation timeout and retry values from their default values.

The following tables summarize the yaml files that will deploy the Operator, and the example yaml files that will create an `HNAS` CustomResource instance used to manage the driver lifecycle:

Deployment file for the Hitachi NAS CSI driver Operator	
File	Details
hnas-operator.yaml	Contains information that installs the Hitachi NAS CSI driver Operator onto Kubernetes.  The Operator is installed into the <code>hnas-csi-operator-system</code> namespace.

CustomResource creation files for the Hitachi NAS CSI driver Operator	
File	Details
hnas_v1_hnas.yaml	Creates an instance of the <code>HNAS</code> CustomResource, called <code>hnas</code> , and uses all the default values.
hnas_v1_hnas_with-spec.yaml	Creates an instance of the <code>HNAS</code> CustomResource, called <code>hnas</code> .  The file contains all the parameters that can be changed – most are commented out, but the ones specifically mentioned in other parts of this document are left uncommented.

All the parameters that can be changed when installing the Hitachi NAS CSI driver using the Operator are described in *Appendix 3: Options Available when Installing with the Kubernetes Operator*.

The following table summarizes the sample configuration yaml files supplied:

Configuration files for the Kubernetes CSI environment	
File	Details
hnas-secret-sample.yaml	A sample file which contains information about each Hitachi NAS system and authentication information
hnas-sc-sample.yaml	A sample file which contains information about the Kubernetes StorageClass, which specifies where the data is stored on the Hitachi NAS system, and how it's mounted
hnas-pvc-sample.yaml	A sample file which contains information that will create a new dynamic PersistentVolume and associated PersistentVolumeClaim, on a Hitachi NAS system
hnas-pvc-clone-sample.yaml	A sample file which contains information that will create a dynamic PersistentVolume and associated PersistentVolumeClaim, where the new volume is pre-populated with data cloned from an existing PersistentVolume host on Hitachi NAS
hnas-pod-sample.yaml	A sample file which contains information about a Kubernetes Pod which will consume a PersistentVolume created by the Hitachi NAS CSI driver
hnas-snapclass-sample.yaml	A sample file which contains information used to create a VolumeSnapshotClass, which provides details on how snapshots are created of existing PersistentVolumes on a Hitachi NAS system
hnas-snapshot-sample.yaml	A sample file which contains information that can be used to create a VolumeSnapshot from an existing PersistentVolumeClaim
hnas-pvc-from-snapshot-sample.yaml	A sample file which contains information that will create a dynamic PersistentVolume and associated

	PersistentVolumeClaim, where the new volume is pre-populated with data from an existing Hitachi NAS VolumeSnapshot
hnas-static-pv-sample.yaml	A sample file which contains information that allows an existing Hitachi NAS NFS export to be used as a static PersistentVolume
hnas-static-pvc-sample.yaml	A sample file which contains information that will create a PersistentVolumeClaim on a static Hitachi NAS PersistentVolume

## Install using the Deployment File

The Hitachi NAS CSI driver can be installed using the deployment file contained within the package.

**Note:** If the driver was previously installed using the Operator, it must be uninstalled before installing with the deployment file. Refer to *Uninstall CSI Driver* in the *Data Locations and Admin Tasks* section.

1. Extract the Hitachi NAS CSI driver package and move to the **deploy** directory.
- 2a. Complete the following step for a **new install**:

```
# kubectl create -f hnas-csi-k8s.yaml
```

- 2b. Complete the following step to **upgrade an existing install**. Note there should be no need to delete any existing configuration or volumes that are already in use:

```
# kubectl replace --force -f hnas-csi-k8s.yaml
```

3. Verify whether the Hitachi NAS CSI driver is running – it may take a short while before the pods finish starting – the command output shown below is for a single node Kubernetes cluster, so only includes one instance of the `hnas-csi-node`. For a multiple node Kubernetes cluster, there should be an instance of `hnas-csi-node` for each cluster node:

```
# kubectl get pod -n kube-system | grep hnas
hnas-csi-controller-76c864f9f5-m5xf5    6/6    Running    0    62s
hnas-csi-node-qv22j                    3/3    Running    0    62s
```

## Install using the Operator

The Hitachi NAS CSI driver can be installed using the Operator. A separate deployment file for the Operator is contained within the package.

**Note:** If the driver was previously installed using the deployment file, it must be uninstalled before installing with the Operator. Refer to *Uninstall CSI Driver* in the *Data Locations and Admin Tasks* section.

1. Extract the Hitachi NAS CSI driver package and move to the **operator** directory.
- 2a. Complete the following step for a **new install**:

```
# kubectl create -f hnas-csi-operator.yaml
```

- 2b. Complete the following step to **upgrade an existing install**. Note there should be no need to delete any existing configuration or volumes that are already in use:

```
# kubectl replace --force -f hnas-csi-operator.yaml
```

3. Verify that the Hitachi NAS CSI operator is running – it may take a short while before the pod finishes starting:

```
# kubectl get pods -n hnas-csi-operator-system
NAME                                                    READY   STATUS    RESTARTS   AGE
hnas-csi-operator-controller-manager-5d5cb7bc5-v8gpj  1/1     Running   0           60s
```

4. If you want to change the Hitachi NAS CSI driver settings, edit `hnas_v1_hnas.yaml`. For details about the configuration settings, refer to *Appendix 3: Options Available when Installing with the Operator*
5. Create an instance of the `HNAS CustomResource` using the following command:

```
# kubectl create -f hnas_v1_hnas.yaml
```

And then check that the new instance of the CustomResource has been created:

```
# kubectl get hnas -n kube-system
NAME      AGE
hnas     51s
```

Once created, the Hitachi NAS CSI driver controller and node pods should start to be created.

## Installing on Red Hat OpenShift

To install the Hitachi NAS CSI driver on Red Hat OpenShift, either of the install processes can be used as described in the previous parts of this section. The OpenShift specific `oc` command may be used instead when issuing the `kubectl` commands, but the same results should be achieved when deploying the driver/operator using either the `oc` or `kubectl` commands.

## Check Installed Components

Each component of the Hitachi NAS CSI driver is made up from multiple containers, which form a pod. Check that the following pods are present and in the running state. These should exist within the **kube-system** namespace, unless that was changed during the install process:

### hnas-csi-controller – single instance

The CSI controller is responsible for the creation, deletion and management of the volumes on the Hitachi NAS system. It also supplies the volume information to Kubernetes.

Container	Image location and version tag
csi-provisioner	registry.k8s.io/sig-storage/csi-provisioner:v5.2.0
external-attacher	registry.k8s.io/sig-storage/csi-attacher:v4.8.0
csi-resizer	registry.k8s.io/sig-storage/csi-resizer:v1.12.0
csi-snapshotter	registry.k8s.io/sig-storage/csi-snapshotter:v8.2.0
liveness-probe	registry.k8s.io/sig-storage/livenessprobe:v2.15.0
hnas-driver	registry.hitachivantara.com/fpgroup-docker-public/hnas-csi-driver:v1.4.2

### hnas-csi-node – an instance on each Kubernetes worker node

Each CSI node is responsible for mounting and unmounting the volumes on the required pods/containers on the specific Kubernetes node they reside on. This is why there is a CSI node running on each Kubernetes node.

Container	Image location and version tag
driver-registrar	registry.k8s.io/sig-storage/csi-node-driver-registrar:v2.13.0
liveness-probe	registry.k8s.io/sig-storage/livenessprobe:v2.15.0
hnas-driver	registry.hitachivantara.com/fpgroup-docker-public/hnas-csi-driver:v1.4.2

It is not possible to test if the driver is working until it has been configured.

## Configuring the Hitachi NAS System

The Hitachi NAS CSI driver requires an existing file serving EVS (Enterprise Virtual Server), which has an IP address that is accessible from all the nodes within the Kubernetes cluster. The EVS also requires at least one file system configured, which is allocated to the EVS. The file system is where the CSI driver will store the volume data, and the EVS IP address will be used by the Kubernetes nodes to communicate using the NFS file serving protocol. These details need to be supplied when configuring the StorageClass(s) later on.

The following example commands are from the Hitachi NAS CLI interface.

All communications between the Kubernetes cluster and the Hitachi NAS are done using the Hitachi NAS File Storage REST API. To check if the API is enabled, use the `rest-server-status` command. If the REST server is not running, use the `rest-server-start` command to enable it.

Create a new API key with the `apikey-create` command – this is the preferred authentication method.

```
nas:$ apikey-create csi-driver
Please make a note of this new API Key, as it is not possible to display the full
key again.
Only the prefix and description can be displayed in the future.
New key: xIAdbgTNVP.Nj2TOgxiOYgpTu2kjjzEGS4QmIJIeLmF3aXKg6FhY9vC
```

To restrict the API key access to just the operations required for the Hitachi NAS CSI Driver, refer to *Appendix 2: Restricting API Key Access*.

Alternatively, a new user can be created, using the `user` command. The example below will create a new supervisor level user called `csi-driver` with a password of `abcd1234`:

```
nas:$ user add csi-driver abcd1234 SUPERVISOR
```

## Configuring the Hitachi NAS CSI Driver

Once the Hitachi NAS CSI driver is installed and running, the following tasks need to be performed before the Hitachi NAS system can be used to host storage volumes:

Configure Hitachi NAS details – The address and credentials for each Hitachi NAS system to be used must be stored within Kubernetes **secret** files. Storing the details this way means they are not easily accessible by other processes. Refer to the *Secret Settings* in the *Configuration Files* section.

Configure one or more StorageClass resources – The details about the Hitachi NAS filesystem are configured in the StorageClass. A different StorageClass may be created depending on the underlying disk types, data retain policy, backup policy, etc. The StorageClass name should make it easy to distinguish between the different types. Refer to *StorageClass Settings* in the *Configuration Files* section.

Configure one or more VolumeSnapshotClass resources if you need to be able to take snapshots of the Hitachi NAS hosted volumes. Refer to the *VolumeSnapshotClass Settings* in the *Configuration Files* section.

For further details about the parameters that can be changed for the Hitachi NAS CSI driver, refer to *Appendix 1: Hitachi NAS Driver Parameters*.

## Static Volumes

Static volumes refer to storage that already exists before being configured within Kubernetes. An existing Hitachi NAS NFS export can be made available for consumption within Kubernetes as a PersistentVolume.

For a static volume to make use of the other CSI driver features, cloning, snapshotting, etc, it must be associated with a StorageClass, as that is where the Hitachi NAS access details are stored. Static volumes to be used in this way **should not** be configured with NFS exports that export the root of a Hitachi NAS filesystem, as that may cause unexpected failures or unexpected results when attempting snapshots or clones. The StorageClass must share the same filesystem as the existing NFS export being used to create the static volume.

If none of the advanced functionality is needed for the static volume, then do not associate it with a StorageClass. The root of a filesystem may be exported in this case and used as a static volume.

## Dynamic Volumes

Dynamic volumes are created on demand by Kubernetes, and generally the PersistentVolume and PersistentVolumeClaim are created at the same time. The volume is created on the Hitachi NAS filesystem specified in the StorageClass that is used to create the new volume.

Dynamic volumes allow space to be allocated/deleted on a Hitachi NAS system directly from the Kubernetes environment, and do not require an administrator to manually create them.

# Limitations

## File Cloning and Snapshotting

The Hitachi NAS CSI driver uses Hitachi NAS file clones for both cloning and snapshotting of volumes, and is not able to clone/snapshot the following file types:

- Files that are not regular (excluding symlinks) i.e. sockets, FIFOs, block special devices, character devices
- Hard links, cross volume links

If these files are encountered within a volume being cloned or snapshotted, they will be ignored and will not be reported as errors. For more details about this behavior, refer to the *tree-clone-job-submit man page* via the *Hitachi NAS Command Line Interface*.

This is a limitation of the file cloning functionality.

## Virtual Volumes

The Hitachi NAS CSI driver makes use of Hitachi NAS tree delete functionality to remove volumes and snapshots. This functionality can be used to remove regular folders, and virtual volumes as a whole, but not folders within virtual volumes. Virtual volumes are also used by the Hitachi NAS CSI driver to enforce size restrictions/limits on dynamic volumes that it hosts, and it is not possible to create a virtual volume within an existing virtual volume. As a consequence, none of the root folder locations used for hosting volumes or snapshots should be within a virtual volume – this refers to the *folder-prefix* value.

For more details about the deletion behavior, refer to the *tree-delete-job-submit man page* via the *Hitachi NAS Command Line Interface*.

This is a limitation of the tree delete functionality.

## Number of Supported Volumes

Each volume created will be accessible via its own NFS export. There is currently a limit of 10000 NFS exports on a Hitachi NAS system, so it will not be possible to create more than 10000 volumes associated with each system. The maximum number of volumes may be less than that, depending on any other configuration on the Hitachi NAS system, and NFS exports used for other purposes.

## Chapter 4: Using The Driver

This section describes how to work with volumes and snapshots managed by the Hitachi NAS CSI driver using the `kubectl` command.

### Create a New Static Volume

An existing NFS export must already be present on a Hitachi NAS system before creating the static volume. The **name**, **volumeHandle**, **server** and **share** parameters must be specified in the yaml file - refer to *PersistentVolume Settings* in the *Configuration Files* section, and optionally the PersistentVolume can be associated with a StorageClass.

```
# kubectl create -f hnas-static-pv-sample.yaml
```

Once the PersistentVolume has been created within Kubernetes, a PersistentVolumeClaim needs to be made on the volume before it can be consumed.

```
# kubectl create -f hnas-static-pvc-sample.yaml
```

Once the PersistentVolumeClaim has been bound, the static volume can be used by Kubernetes in the same way as any other PersistentVolumeClaim.

### Create a New Dynamic Volume

CSI drivers perform this as a two stage process, which first involves the creation of a PersistentVolume, followed by a PersistentVolumeClaim on that volume. Most of the parameters that specify how the new volume is created are inherited from the StorageClass. The **name**, **size** and if the volume is to be shared are the parameters that must be specified in the yaml file - refer to *PersistentVolumeClaim Settings* in the *Configuration Files* section.

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

Once a CSI driver has created a PersistentVolumeClaim, it can be used by Kubernetes in the same way as any other PersistentVolumeClaim.

### Expand an Existing Dynamic Volume

If the StorageClass is created with the `allowVolumeExpansion` setting as true, then it is possible to expand the volume once it has been created. There is no need to delete and recreate the Pod for volume expansion. The following example expands the existing PersistentVolumeClaim, `pvc-sample-hnas`, to 5Gi.

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

## Delete a Dynamic Volume

When the volume is deleted, the Hitachi NAS is instructed to delete the NFS export. If the `PersistentVolume` has a `reclaimPolicy` of **Delete** (this parameter is defined in the `StorageClass` but can be changed once the volume is created), the Hitachi NAS will be instructed to delete all files contained within the exported folder. This is done as a background process, using Hitachi NAS tree delete functionality.

The example below deletes the `PersistentVolumeClaim` called `pvc-sample-hnas`:

```
# kubectl delete pvc pvc-sample-hnas
```

## Using a Volume

To create a new pod that makes use of a `PersistentVolumeClaim` as a volume, which has been created by the Hitachi NAS CSI driver, refer to the *Pod Settings* section in the *Configuration Files* section.

This process is identical to the way it would be done if the `PersistentVolumeClaim` were created manually.

## Creating a Volume Snapshot

When a new snapshot is created, two objects are associated with it – a `VolumeSnapshot` and a `VolumeSnapshotContent`. The `VolumeSnapshotClass` specifies some of the snapshot parameters, but the `yaml` file specifies the snapshot name and existing volume to snapshot - refer to *VolumeSnapshot Settings* in the *Configuration Files* section.

```
# kubectl create -f hnas-snapshot-sample.yaml
```

**Note:** Snapshots created by the CSI driver are not the same as Hitachi NAS filesystem snapshots.

## Deleting a Volume Snapshot

When the snapshot is deleted, the `deletionPolicy` setting from the `VolumeSnapshotClass` determines whether the snapshot contents are deleted or retained. If the deletion policy is set to **Delete**, the Hitachi NAS will be instructed to delete all files contained within the snapshot, otherwise the files are retained on the Hitachi NAS. The deletion is done as a background process, using Hitachi NAS tree delete functionality.

The example below deletes the `VolumeSnapshot` called `snapshot-hnas`:

```
# kubectl delete volumesnapshot snapshot-hnas
```

Deleting a snapshot does not affect any volumes that were created using the snapshot as a data source.

## Creating a Volume from an Existing Source

CSI drivers allow new volumes to be created with a copy of the data from an existing volume or from an existing snapshot. The Hitachi NAS CSI driver supports creating a new volume as a clone of an existing volume and also creating a new volume from a copy of the data contained within a snapshot of an existing volume. For details on both these options, refer to *PersistentVolumeClaim Settings* in the *Configuration Files* section.

The creation command is the same as creating a new, empty volume (see *Create a New Volume* section) and can be used in exactly the same way as any other volume.

# Chapter 5: Configuration Files

This section contains details about each of the configuration files that are used to allow the Hitachi NAS CSI driver to function within Kubernetes.

## Secret Settings

The Secret file contains the Hitachi NAS REST API URL and credentials that are necessary for the Hitachi NAS CSI driver to work with your environment. The following sample provides information about the required and optional parameters. Refer to the *Troubleshooting* section for an example of how to verify that the REST API is functioning correctly.

### Parameter references for hnas-secret-sample.yaml

```
apiVersion: v1
kind: Secret
metadata:
  name: hnas-system-1 # (1)
type: Opaque
stringData:
  apiUrl: "https://172.16.1.1:8444" # (2)
  apiVersion: "9" # (3)
data:
  apiKey: 4oCTbiBlbmNvZGUtbWUK # (4)
  username: 4oCTbiBlbmNvZGUtbWUK # (5)
  password: 4oCTbiBlbmNvZGUtbWUK # (6)
```

### Legend:

- (1) **name** – used to uniquely identify the Hitachi NAS system. If multiple systems are to be used, then create multiple secret files
- (2) **apiurl** – Hitachi NAS REST API URL, including protocol and port number
- (3) **apiVersion** – Hitachi NAS REST API version to be used – versions 7, 8 and 9 are currently supported. This parameter is optional, and defaults to version 7 if omitted
- (4) **apiKey** – base64 encoded Hitachi NAS API Key – this parameter is optional, and can be used instead of providing **username** and **password** parameters

If the **apiKey** parameter is specified, then the following parameters are ignored:

- (5) **username** – base64 encoded Hitachi NAS supervisor level username
- (6) **password** – base64 encoded Hitachi NAS password

Example of base64 encoding a parameter:

```
# echo -n "encode-me" | base64
4oCTbiBlbmNvZGUtbWUK
```

The output from the echo command should be used within the secret file instead of the plain text version.

# StorageClass Settings

The StorageClass file contains parameters related to the hosting filesystem, how it's to be mounted, and some details about how Kubernetes can manage it. The following sample provides information about the required parameters.

## Parameter references for hnas-sc-sample.yaml

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: hnas-gold #(1)
  annotations:
    kubernetes.io/description: High speed Hitachi NAS storage - Gold #(2)
provisioner: hnas.csi.hitachi.com
reclaimPolicy: Delete #(3)
volumeBindingMode: Immediate #(4)
allowVolumeExpansion: true #(5)
mountOptions: #(6)
- "nfsvers=3"
parameters:
  filesystem: "fs1" #(7)
  folderPrefix: "/csi" #(8)
  accessConfig: "(norootsquash)" #(9)
  adminEmail: "email@example.com" #(10)
  preferredAddress: "192.168.0.100" #(11)
  csi.storage.k8s.io/provisioner-secret-namespace: "default" #(12)
  csi.storage.k8s.io/provisioner-secret-name: "hnas-system-1" #(13)
  csi.storage.k8s.io/controller-expand-secret-namespace: "default" #(12)
  csi.storage.k8s.io/controller-expand-secret-name: "hnas-system-1" #(13)
```

### Legend:

- (1) **name** – name of the new StorageClass
- (2) **description** – a description to be associated with the StorageClass (optional)
- (3) **reclaimPolicy** – **Delete** or **Retain**. If the StorageClass reclaim policy is Delete, all data associated with a volume will be deleted when the volume is deleted, but if the policy is set to Retain, it is the responsibility of the Administrator to manually delete the data
- (4) **volumeBindingMode** – **Immediate** or **WaitForFirstConsumer**. When the mode is set to WaitForFirstConsumer, any new volumes made with the StorageClass are not actually created on the Hitachi NAS until the first time it is used, otherwise it will be created immediately
- (5) **allowVolumeExpansion** – **true** or **false**. If true, the volume can be expanded once it has been created
- (6) **mountOptions** – these are the NFS client mount options – see below for more details
- (7) **filesystem** – Hitachi NAS filesystem, where the new volumes will be created
- (8) **folderPrefix** – Filesystem folder name, where all volumes will be created. This folder must not be within a virtual volume
- (9) **accessConfig** – NFS export mount options – refer to the *Hitachi NAS documentation* for a full list of options
- (10) **adminEmail** – a comma separated list of email addresses, which will receive notifications when the storage is getting full – a warning message will be sent at 75% full and a severe message at 95% full
- (11) **preferredAddress** – optional EVS IP address to use for connecting to the Hitachi NAS server. If omitted, a reachable EVS IP address will be chosen for the NFS mount

- (12) Namespace used to store the secrets file that contains the Hitachi NAS system details
- (13) Name specified in the Hitachi NAS system secrets file – this associates the StorageClass with a specific Hitachi NAS system

### mountOptions – more details

If the **mountOptions** are omitted, by default NFSv4 will be used to mount the Hitachi NAS NFS export – use the Hitachi NAS CLI command `nfs-max-supported-version` to check if NFSv4 is enabled or not, otherwise the mount will fail. If NFSv3 is required, then the `nfsvers=3` option should be specified.

For details on other mount options refer to the *Linux man page* for *mount*.

## VolumeSnapshotClass Settings

The VolumeSnapshotClass file contains parameters related to the folder location to store the snapshots, and some details about how Kubernetes can manage them. The following sample provides information about the required parameters.

### Parameter references for hnas-snapclass-sample.yaml

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: hnas-snapclass # (1)
driver: hnas.csi.hitachi.com
deletionPolicy: Delete # (2)
parameters:
  folderPrefix: "/csi-snapshots" # (3)
  exportPrefix: "snapshots" # (4)
  csi.storage.k8s.io/snapshotter-secret-namespace: "default" # (5)
  csi.storage.k8s.io/snapshotter-secret-name: "hnas-system-1" # (6)
```

### Legend:

- (1) **name** – name of the new VolumeSnapshotClass
- (2) **deletionPolicy** – **Delete** or **Retain**. If the VolumeSnapshotClass deletion policy is Delete, all data associated with the snapshot will be deleted when the snapshot is deleted, but if the policy is set to Retain, it is the responsibility of the Administrator to manually delete the data
- (3) **folderPrefix** – Filesystem folder name, where all snapshots will be created. Each snapshot will be stored on the same filesystem as its associated volume. This folder must not be within a virtual volume, otherwise later deletion of the snapshot data by the CSI driver will not be possible
- (4) **exportPrefix** – Optional prefix to use for the NFS export that allows access to the snapshots. If omitted, all snapshots will be grouped under the same NFS export
- (5) Namespace used to store the secrets file that contains the Hitachi NAS system details
- (6) Name specified in the Hitachi NAS system secrets file – this associates the VolumeSnapshotClass with a specific Hitachi NAS system

# PersistentVolume Settings

The PersistentVolume file contains information related to an existing Hitachi NAS NFS export that is to be used as a static volume, rather than allowing the Hitachi NAS CSI driver to create a new dynamic one.

## Parameter references for hnas-static-pv-sample.yaml

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: static-pv-hnas # (1)
spec:
  capacity:
    storage: 123Gi # (2)
  accessModes:
    - ReadWriteMany # (3)
  persistentVolumeReclaimPolicy: Retain # (4)
  mountOptions: # (5)
    - "nfsvers=3"
  storageClassName: "hnas-gold" # (6)
  csi:
    driver: hnas.csi.hitachi.com
    volumeAttributes:
      server: 172.27.146.72 # (7)
      share: 2_fs2 # (8)
      volumeHandle: dc9b1d2e-5ab4-11d1-9008-75728f512f9c # (9)
```

Legend:

- (1) **name** – meaningful name of the PersistentVolume, which will be used when creating a PersistentVolumeClaim
- (2) **capacity** – size that Kubernetes reports as being available for this volume. No check is performed as to whether this value matches the space available on the existing volume
- (3) **accessModes** – this specifies how the PersistentVolume can be accessed, and can be one of **ReadWriteOnce**, **ReadOnlyMany**, **ReadWriteMany**
- (4) **persistentVolumeReclaimPolicy** – should always be set to **Retain**, otherwise the CSI driver may attempt to delete the existing data when the PV is deleted
- (5) **mountOptions** – these are the NFS client mount options – see the *StorageClass Settings* in the *Configuration Files* section for more details
- (6) **storageClassName** – either the name of an existing StorageClass, which is being used to create dynamic volumes on the same Hitachi NAS filesystem, as this PersistentVolume is hosted, or an empty string. An empty string will not associate the PersistentVolume with any StorageClass. Omitting the storageClassName parameter completely will associate the PV with the default StorageClass, which is probably not the desired setting
- (7) **server** – EVS IP address to use for connecting to the Hitachi NAS server
- (8) **share** – the name of an existing NFS export hosted by the EVS referred to in the **server** parameter. This value **must not** start with a / character
- (9) **volumeHandle** – the volume handle needs to be the **Global identifier** parameter associated with the Hitachi NAS NFS export, and can be retrieved either via the Hitachi NAS CLI or via the REST API

If a `storageClassName` is supplied, and the NFS export being used is not exporting the root of a filesystem, it should be possible to create clones and snapshots of the static volume, in the same way that they can be created for dynamic volumes.

To obtain the NFS export global identifier needed for the `volumeHandle` parameter, use the following Hitachi NAS CLI command, which is retrieving the **Global identifier** associated with the NFS export called **2\_fs2**. The CLI must be in the correct EVS context:

```
nas[EVS02]:$ nfs-export list -v 2_fs2 | grep Global
Global identifier: dc9b1d2e-5ab4-11d1-9008-75728f512f9c
```

## PersistentVolumeClaim Settings

The `PersistentVolumeClaim` file contains volume information that is used by the Hitachi NAS CSI driver to create new dynamic `PersistentVolumes`, or claim static `PersistentVolumes`. The dynamically created volumes can either be empty or populated with data cloned from an existing `PersistentVolume` of the same class and size, or the contents of a snapshot.

The following sample provides information about the required parameters for a new, empty `PersistentVolumeClaim`.

### Parameter references for `hnas-pvc-sample.yaml`

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

Legend:

- (1) **name** – meaningful name, which will be used to access the `PersistentVolumeClaim` from a pod
- (2) **accessModes** – this specifies how the `PersistentVolumeClaim` can be accessed, and can be one of **ReadWriteOnce**, **ReadOnlyMany**, **ReadWriteMany**
- (3) **storage** – initial size to be allocated for the storage. If the `StorageClass` allows expansion, this can be changed later
- (4) **storageClassName** – the name of the `StorageClass` used to create the `PersistentVolumeClaim`

The following file shows how to create a new `PersistentVolumeClaim`, which is created by cloning the contents of the volume called `pvc-sample-hnas`, as the data source. This creates a second, writable copy of the source data.

## Parameter references for hnas-pvc-clone-sample.yaml

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-clone-hnas # (1)
spec:
  accessModes:
    - ReadWriteMany # (2)
  resources:
    requests:
      storage: 1Gi # (3)
  storageClassName: hnas-gold # (4)
  dataSource:
    name: pvc-sample-hnas # (5)
    kind: PersistentVolumeClaim
    apiGroup: ""
```

### Legend:

- (1), (2), (3), (4) – same as definitions above
- (5) **dataSource name** – the name of an existing PersistentVolumeClaim of the same StorageClass. Parameter (3) should be either the same size of the existing volume being used as the data source, or can be a larger value

The following file shows how to create a new PersistentVolumeClaim, which is created by cloning the contents of the VolumeSnapshot called `snapshot-hnas`, as the data source. This creates a writable copy of the source snapshot data.

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

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-from-snapshot # (1)
spec:
  accessModes:
    - ReadWriteMany # (2)
  resources:
    requests:
      storage: 1Gi # (3)
  storageClassName: hnas-gold # (4)
  dataSource:
    name: snapshot-hnas # (5)
    kind: VolumeSnapshot
    apiGroup: "snapshot.storage.k8s.io"
```

### Legend:

- (1), (2), (3), (4) – same as definitions above
- (5) **dataSource name** – the name of an existing VolumeSnapshot of the same StorageClass. Parameter (3) should be either the same size as the source of the snapshot, or can be a larger value

The following file shows how to create a PersistentVolumeClaim for a statically created Hitachi NAS PersistentVolume. Creating the PersistentVolumeClaim is required before the volume can be consumed.

### Parameter references for hnas-static-pvc-sample.yaml

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: pvc-sample-hnas #(1)
spec:
  accessModes:
    - ReadWriteMany #(2)
  resources:
    requests:
      storage: 1Gi #(3)
  storageClassName: "" #(4)
  volumeName: static-pv-hnas #(5)
```

Legend:

- (1), (2) – same as definitions above
- (3) **storage** – parameter is generally ignored for a static volume, as the size reported by Kubernetes is determined by the PersistentVolume. However, the parameter must be supplied, and valid, otherwise the creation will fail
- (4) **storageClassName** – must match the storageClassName used when the PersistentVolume was created – the example here shows how to specify a blank storage class
- (5) **volumeName** – name of the PersistentVolume the claim is being made against

## VolumeSnapshot Settings

The VolumeSnapshot file contains information that is used by the Hitachi NAS CSI driver to create a new VolumeSnapshot from an existing PersistentVolumeClaim. The following sample provides information about the required parameters for creating a new snapshot.

### Parameter references for hnas-snapshot-sample.yaml

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

Legend:

- (1) **name** – the name of the new snapshot
- (2) **volumeSnapshotClassName** – the name of the VolumeSnapshotClass used when creating the snapshot

- (3) **persistentVolumeClaimName** – name of the PersistentVolumeClaim that is to be snapshotted

## Pod Settings

The Pod file contains container and volume information that is used by Kubernetes to create a Pod that uses the PersistentVolumeClaim as a mounted volume within the container. The following sample provides information about the required parameters.

### Parameter references for hnas-pod-sample.yaml

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

#### Legend:

- (1) **name** – the name of the new Pod
- (2) **mountPath** – a path within the pod where the Hitachi NAS volume will be mounted. The name should match the name within the volumes section, which references the PersistentVolumeClaim
- (3) **claimName** – is the name of the PersistentVolumeClaim that is to be used with this pod

# Chapter 6: Data Locations and Admin Tasks

One of the advantages of storing the volumes on a Hitachi NAS system is that it's possible to access the volume data from other systems, outside of Kubernetes. This can be useful for pre-populating the volumes and backing up the data. It also allows data to be shared/manipulated by clients that are not run within a containerized environment.

## Populating a Volume with Data

Once a PersistentVolumeClaim has been created for a dynamic volume, it's possible to populate it with data before it's mounted by a pod/container. Either access the NFS export associated with the PersistentVolumeClaim directly (provided the `accessConfig` options specified in the StorageClass allow it) – its name will correspond to the name of the PersistentVolumeClaim, and will be of the form:

```
/pvc-9161d39b-82f6-417e-b435-2a79fc0ca49c
```

It's also possible to access the folder hosting the PersistentVolumeClaim if a higher level mount point exists on the same filesystem. The StorageClass `filesystem` and `folderPrefix` values can be combined to determine the location of PersistentVolumeClaims. All PersistentVolumeClaims will be stored in individual directories under the directory specified by the `folderPrefix` value.

## Access Volume Data for Backup

To access the data within a volume for backup purposes, use the same approach as shown in the *Populating a Volume with Data* section.

## Access Volume Snapshot Data for Backup

Snapshots associated with a specific VolumeSnapshotClass are grouped together in a common location on each filesystem, specified by the `folderPrefix` value. When the first snapshot is created, an NFS export is created that exports the `folderPrefix` directory, which allows NFS access to the snapshots. The export is created to allow read-only access to the snapshots. By default, the export is named:

```
/csi-snapshots@<fs_label>
```

If the `exportPrefix` is specified in the VolumeSnapshotClass, the export will be named as below. This allows multiple snapshot classes to exist on the same filesystem.

```
/<exportPrefix>@<fs_label>
```

Do not delete the snapshot exports, otherwise the CSI driver will not be able to locate the snapshot data.

If the VolumeSnapshotClass `deletionPolicy` is set to **Retain**, then once the snapshot is no longer needed, the folder containing the snapshot data will need to be manually removed.

Deletion of the data can be done using the Hitachi NAS `tree-delete-job-submit` CLI command or from another client.

## Manual Deletion of Volume Data

If a PersistentVolumeClaim has a Reclaim Policy of **Retain**, the data will not automatically get deleted when the PersistentVolumeClaim is deleted in Kubernetes. The NFS export associated with the PersistentVolumeClaim and Virtual Volume will also be retained on the Hitachi NAS and need to be removed using one of the Hitachi NAS management interfaces once they are no longer needed.

Deletion of the data can be done using the Hitachi NAS `tree-delete-job-submit` CLI command or from another client, accessing the data in the same way as shown in the *Populating a Volume with Data* section.

## Update Hitachi NAS Credentials

The credentials used to access the Hitachi NAS systems REST API can be updated by editing the appropriate `secret.yaml` file and reapplying it to Kubernetes. The Hitachi NAS CSI driver will pick up the new credentials when it is next used.

```
# kubectl replace -f hnas-secret.yaml
```

# Uninstall CSI Driver

To uninstall the Hitachi NAS CSI driver, perform the following steps below.

## Procedure

If you are uninstalling the driver to replace it with a new version, skip steps 1 and 2.

1. Delete all Pods which are using the volumes created by the Hitachi NAS CSI driver.
2. Delete all PersistentVolumeClaims, PersistentVolumes, VolumeSnapshots, StorageClasses, VolumeSnapshotClasses and any Secrets related to the Hitachi NAS CSI driver.
- 3a. Complete the following step if the HNAS CSI driver was installed with the **deployment file**.

Delete the Hitachi NAS CSI driver, using the same deployment file that was used to install the driver:

```
# kubectl delete -f hnas-csi-k8s.yaml
```

- 3b. Complete the following step if the HNAS CSI driver was installed using the **operator**.

Delete the `hnas` CustomResource which will cause the Hitachi NAS CSI driver to be deleted. Once the using the same deployment file that was used to install the driver:

```
# kubectl delete -f hnas_v1_hnas.yaml
```

Once the Hitachi NAS CSI driver controller and node pods have been removed, uninstall the operator using the same deployment file that was used to install it:

```
# kubectl delete -f hnas-csi-opertor.yaml
```

# Chapter 7: Troubleshooting

As a first step to troubleshooting any issues, ensure the Hitachi NAS REST API is enabled, as described in *Chapter 3*, and that the Kubernetes nodes are able to ping any IP addresses used in the configuration files.

**Note:** Any values changed in the installation yaml files require the driver to be replaced to make the new settings active.

```
# kubectl replace --force -f hnas-csi-k8s.yaml
```

## Obtaining Logs

All the components that make up the Hitachi NAS CSI driver can have their log level changed to help with troubleshooting. The log levels are from 0 (minimal logging) to 9 (large amount of logging, including full HTTP request). All logs are configured at Level 5 by default.

The log level for each component is specified when they are originally created – so unless it's needed, they should not be changed.

If the driver does not appear to be working correctly, the first places to look are the console logs:

1. If the new volumes are not being created, then look at the logs associated with the `hnas-driver` container, hosted as part of the **CSI Controller** pod.
2. If there are issues mounting the volumes by newly created pods, look at the logs associated with the `hnas-driver` container, hosted as part of the **CSI Node** pod on the Kubernetes node hosting the new pod.

The following example shows how to get the logs from the `hnas-driver` container from one of the CSI Node drivers:

```
# kubectl get pods -n=kube-system
NAME                                READY   STATUS    RESTARTS   AGE
coredns-66bff467f8-clgb5            1/1     Running   1           21d
coredns-66bff467f8-gtqj6            1/1     Running   1           21d
etcd-minikube                        1/1     Running   1           21d
hnas-csi-controller-67ccb6748f-j24pq 6/6     Running   0           20d
hnas-csi-node-4n59t                3/3     Running   0           20d
kube-apiserver-minikube              1/1     Running   2           21d
kube-controller-manager-minikube     1/1     Running   2           21d
kube-proxy-hvvr8                     1/1     Running   1           21d
kube-scheduler-minikube              1/1     Running   1           21d
storage-provisioner                  1/1     Running   1           21d
# kubectl logs -n=kube-system hnas-csi-node-4n59t hnas-driver
```

## Changing the Log Level

The log level can be changed for each component separately, and is specified in the installation yaml file, as the `--v` parameter.

The log level can be changed by adjusting the following value in the installation yaml file for the corresponding component that requires adjustment:

```
args:
- "--v=5"
```

## CSI Driver cannot connect to the Hitachi NAS Server

The connection parameters and credentials can be checked using a simple curl command, run on one of the Kubernetes nodes. Run the following command, and replace the `<username>`, `<password>` and `<admin_IP>` parameters with the ones to be used by the CSI driver. If the command works, it should return some information about the cluster nodes, and some additional debug info associated with the connection. If the credentials are wrong, a message should be returned with a "401 Unauthorized" error code. If the `<admin_IP>` address is wrong, the connection should either be refused, or timeout.

```
# curl -v -k -H "X-Subsystem-User: <username>" -H "X-Subsystem-Password: <password>" https://<admin_IP>:8444/v9/storage/nodes
```

If an API key is to be used instead of user/password combination, use the following curl command, replacing the `<api_key>` and `<admin_IP>` parameters as appropriate:

```
# curl -v -k -H "X-API-Key: <api_key>" https://<admin_IP>:8444/v9/storage/nodes
```

For more details on troubleshooting connectivity issues to the REST API, refer to the *"Hitachi NAS File Storage, REST API Reference"*.

## Kubernetes Node Unable to NFS Mount the Hitachi NAS Server

The most likely cause of a mount failure is an NFS version mismatch between the Hitachi NAS server and the Kubernetes node. By default, Hitachi NAS systems are not configured to support NFSv4, and if no mount options are specified in the StorageClass definition, the Kubernetes node will attempt to use NFSv4 to mount the NFS export. There are two solutions to this issue - enable NFSv4 on the Hitachi NAS or specify NFSv3 as an option when creating the StorageClass. NFSv3 as a mount option should be the preferred solution, as enabling NFSv4 on a Hitachi NAS system that does not already have it enabled, could potentially cause other NFS clients to mount via NFSv4 when next mounted, which may have unintentional behavior changes.

## Intermittent Connection Failure

Another possible reason for connection failures between the CSI driver and the Hitachi NAS REST API may be due to all connection resources being used up on the Hitachi NAS server, as it only supports a limited number of simultaneous connections. The `msstats` command accessible via the Hitachi NAS CLI can be used to determine how many current connections are open to the Hitachi NAS server, and if the connection limit has been reached, or if connections are being rejected:

```
nas:$ msstats rest
RestAPI

Listening on port 8444
Maximum simultaneous connections 50

Current connections: 1           Frames rxed: 184978
Max connections: 10           Frames txed: 378912
Total connections: 2308       Bytes rxed: 62302762
Rejected connections: 0       Bytes txed: 280864069
Successful logins: 2308       Requests rxed: 178432
Failed logins: 0              Responses txed: 178432

Current Connections

Address      Age (s)  Rx age (s)  Tx age (s)  Rx pkts  Tx pkts
172.27.141.190 801      21          21          37       73
```

The maximum simultaneous connections can be adjusted up to a maximum value of 50. The connections accepted can also be limited to specific addresses if there appear to be connections from unknown sources, although restricting the connections may cause other management tools to stop working.

By default, the CSI driver will attempt to retry to establish a new connection 5 times before it gives up. Each time a connection attempt fails, the CSI driver will back off for a random time between 1 and 30 seconds before it will try to make a new connection attempt.

These values can be changed by adjusting the following values in the installation yml file – find the `args` section of the `hnas-driver`, within the `hnas-csi-controller` section of the yml file:

```
args:
- "--v=5"
- "--nodeid=$(KUBE_NODE_NAME)"
- "--endpoint=$(CSI_ENDPOINT)"
- "--timeout=300s"
- "--retry-interval-max=30s"
- "--retry-count-max=5"
```

## Operation Timeout

By default, the CSI driver is set with an operation timeout of 300 seconds, which should be long enough for most operations to return a response. In the situation where an operation is still ongoing and the driver times out waiting for the response, the timeout value can be increased by adjusting the following value in the installation yaml file - find the `args` section of the `hnas-driver`, within the `hnas-csi-controller` section of the yaml file:

```
args:
- "--v=5"
- "--nodeid=$(KUBE_NODE_NAME) "
- "--endpoint=$(CSI_ENDPOINT) "
- "--timeout=300s"
- "--retry-interval-max=30s"
- "--retry-count-max=5"
```

**Note:** When adjusting the timeout value, bear in mind that other components within Kubernetes will also have timeout values, and if the CSI driver timeout is increased too much, then other parts of the system may timeout before the driver does. Investigating why the operation is taking so long would also be a good idea if the default timeout is too short.

## Volume creation/deletion operations not able to complete

If too many volume creation or deletion operations are attempted simultaneously, the timeout/retry settings for the various components within the CSI controller can be adjusted to allow operations to be retried (See previous sections), but if there are still failures, the number of worker threads the `csi-provisioner` runs can be lowered. The initial value is 20 – lowering it to 10 should halve the maximum number of operations that get sent simultaneously - find the `args` section of the `csi-provisioner`, within the `hnas-csi-controller` section of the yaml file:

```
args:
- "--csi-address=$(ADDRESS) "
- "--timeout=300s"
- "--worker-threads=10"
```

## Snapshot creation failing

If too many snapshot operations are attempted simultaneously, the timeout/retry settings for the various components within the CSI controller can be adjusted to allow operations to be retried (See previous sections), but if there are still failures, the number of worker threads the `csi-snapshotter` runs can be lowered. The container is run with the default value, which is 10 – specifically lowering it to 5 should halve the maximum number of operations that get sent simultaneously - find the `args` section of the `csi-snapshotter`, within the `hnas-csi-controller` section of the yml file:

```
args:
- "--v=5"
- "--csi-address=$(ADDRESS) "
- "--timeout=300s"
- "--worker-threads=5"
```

# Appendix 1: Hitachi NAS Driver Parameters

All parameters are specific to the controller and will have no effect when supplied to the node resources:

The following parameters related to the REST API requests sent from the CSI driver to the Hitachi NAS system.

Parameter	Default value	Description
timeout	120 seconds	Timeout for individual REST requests
retry-interval-max	30 seconds	Max length of time between retrying REST requests
retry-count-max	5	Max number of times to retry a REST request

When creating a new volume that is pre-populated with the contents of another volume or from a snapshot, the CSI spec is not clear if the driver should wait for the population to complete before the response is sent back to Kubernetes or not. If the operation to pre-populate the volume takes a long time, Kubernetes may timeout the operation and attempt it again, which can have undesired effects and will also probably keep failing. This option determines if the driver returns a success message when the volume is created, or when the volume is populated.

Parameter	Default value	Description
wait-for-volume-content-population	true	Whether to wait for a new volume to be populated with the defined content, or allow the data to be copied in the background

The following parameters are only relevant if using Hitachi NAS software 15.2 or greater. 15.2 introduced API calls that allow the CSI driver to lookup ongoing clone operations. The CSI driver snapshot functionality is based on creating clones of each file within the volume. The required APIs were added to Hitachi NAS software release 15.2.

Parameter	Default value	Description
use-async-operations-if-available	true	Whether to use asynchronous operations when creating snapshots or clones, if the APIs are available. Set to <b>false</b> to revert to the previous behavior
async-check-count-max	10	Max number of times to check if an asynchronous snapshot or clone operation has completed
async-check-interval-max	1	Max length of time between each check to see if an asynchronous snapshot or clone operation has completed

## Appendix 2: Restricting API Key Access

A valid set of username/password credentials for the REST API will also be a valid means to login to the Hitachi NAS CLI interface. API Keys are a more secure method to provide authentication to the Hitachi NAS File Storage REST API than the traditional username/password access and should be used in preference to a username/password combination, as they do not allow access to any management interfaces other than access to external APIs.

From Hitachi NAS software release 15.4 or later, API keys can be further restricted to limit access to a subset of API calls, and to add further restrictions, by limiting the API calls to one or more EVSs, rather than allowing full read/write access to all API calls.

To restrict an existing API key, use the following command to create a new API key group, which will be called **CsiDriver**. An API key group takes a list of API calls and puts them into a group that can be easily assigned to existing API keys to restrict their access to a subset of the available API calls. Attempting to access any other API calls with the restricted API key will return a **403 Forbidden** message, and the API call will fail.

The command below contains the full list of API calls that the Hitachi NAS CSI driver uses – the list covers calls that are applicable to all API versions.

```
nas:$ apikey-group-create --name CsiDriver --add-api-calls
getFileDevice,getFilesystemExports,createFilesystemExport,deleteFilesystemExport,updateFilesystemExport,getFilesystems,getFilesystem,getFilesystemVirtualServer,getVirtualServers,cloneFilesystemDirectory,submitTreeCloneJob,getRecentFilesystemTreeCloneJobs,getTreeCloneJob,createFilesystemDirectory,getFilesystemRootDirectory,getFilesystemDirectory,deleteFilesystemDirectory,createVirtualVolume,getVirtualVolumes,createVirtualVolumeQuota,modifyQuota,modifyVirtualVolumeQuota,getVirtualVolumeQuota,getVirtualVolumeQuotas
API Key group CsiDriver successfully created
```

Once the API key group is created, it can be used to restrict an existing API key. The following example restricts the API key which is associated with the description **csi-driver** to the API calls that are in the API key group **CsiDriver**:

```
nas:$ apikey-restrict --description csi-driver --apikey-groups CsiDriver
API Key restrictions updated
```

API keys can be further restricted to one or more EVSs, ensuring the allowed operations can only be performed on an allowed set of EVSs, rather than for all EVSs configured on the Hitachi NAS system. The following example will restrict the API key associated with the description **csi-driver** to only be able to perform EVS specific operations that relate to EVS **2**, rather than all EVSs:

```
nas:$ apikey-restrict --description csi-driver --evss 2
API Key restrictions updated
```

To check the restrictions that apply to an API key, use the following command:

```
nas:$ apikey-restrict --description csi-driver
The API Key is restricted to API calls in the following API Key groups:
```

Index	Group Name
1	CsiDriver

The API Key is restricted to the following EVSs:

EVS ID	Label
2	EVS-2

A restricted API key will be far more secure than one that is unrestricted, but the operations it can perform will be more limited.

## Appendix 3: Options Available when Installing with the Operator

When using the Operator to install the Hitachi NAS CSI driver, it's easy to modify the install parameters by adding options to the CustomResource definitions file.

**Note:** If specifying container arguments, any specified will replace all the existing default arguments, so ensure that all existing arguments are included when adding new ones, otherwise the driver may not work properly.

Parameters that can be changed and are referenced in other places in this document:

Parameter	Description
<code>spec.controller.hnasDriver.args</code>	Arguments provided to the controller instance of the <code>hnas-driver</code> container
<code>spec.controller.csiProvisioner.args</code>	Arguments provided to the <code>csi-provisioner</code> container
<code>spec.controller.csiSnapshotter.args</code>	Arguments provided to the <code>csi-snapshotter</code> container
<code>spec.node.hnasDriver.args</code>	Arguments provided to all the node instances of <code>hnas-driver</code> container

**Note:** `imagePullPolicy` will default to `Always` if the image tag is `latest`, irrespective of what it is set to.

Parameters which can be changed, but are unlikely to need changing:

Parameter	Description
<code>spec.controller.csiProvisioner.image</code>	The image name and tag of the <code>csi-provisioner</code> container
<code>spec.controller.csiProvisioner.imagePullPolicy</code>	The image pull policy for the <code>csi-provisioner</code> container
<code>spec.controller.externalAttacher.image</code>	The image name and tag of the <code>external-attacher</code> container
<code>spec.controller.externalAttacher.imagePullPolicy</code>	The image pull policy for the <code>external-attacher</code> container
<code>spec.controller.externalAttacher.args</code>	Arguments provided to the <code>external-attacher</code> container
<code>spec.controller.csiResizer.image</code>	The image name and tag of the <code>csi-resizer</code> container

<code>spec.controller.csiResizer.imagePullPolicy</code>	The image pull policy for the <code>csi-resizer</code> container
<code>spec.controller.csiResizer.args</code>	Arguments provided to the <code>csi-resizer</code> container
<code>spec.controller.csiSnapshotter.image</code>	The image name and tag of the <code>csi-snapshotter</code> container
<code>spec.controller.csiSnapshotter.imagePullPolicy</code>	The image pull policy for the <code>csi-snapshotter</code> container
<code>spec.controller.livenessProbe.image</code>	The controller image name and tag of the <code>liveness-probe</code> container
<code>spec.controller.livenessProbe.imagePullPolicy</code>	The controller image pull policy for the <code>liveness-probe</code> container
<code>spec.controller.livenessProbe.args</code>	Arguments provided to controller instance of the <code>liveness-probe</code> container
<code>spec.controller.hnasDriver.image</code>	The controller image name and tag of the <code>hnas-driver</code> container
<code>spec.controller.hnasDriver.imagePullPolicy</code>	The controller image pull policy for the <code>hnas-driver</code> container
<code>spec.node.driverRegistrar.image</code>	The image name and tag of the <code>driver-registrar</code> container
<code>spec.node.driverRegistrar.imagePullPolicy</code>	The image pull policy for the <code>driver-registrar</code> container
<code>spec.node.driverRegistrar.args</code>	Arguments provided to the <code>driver-registrar</code> container
<code>spec.node.livenessProbe.image</code>	The node image name and tag of the <code>liveness-probe</code> container
<code>spec.node.livenessProbe.imagePullPolicy</code>	The node image pull policy for the <code>liveness-probe</code> container
<code>spec.node.livenessProbe.args</code>	Arguments provided to node instances of the <code>liveness-probe</code> containers
<code>spec.node.hnasDriver.image</code>	The node image name and tag of the <code>hnas-driver</code> containers
<code>spec.node.hnasDriver.imagePullPolicy</code>	The node image pull policy for the <code>hnas-driver</code> containers

## Hitachi Vantara

Corporate Headquarters 2535 Augustine Drive  
Santa Clara, CA 95054 USA [www.HitachiVantara.com](http://www.HitachiVantara.com) [community.HitachiVantara.com](http://community.HitachiVantara.com)

### Regional Contact Information

Americas: +1 866 374 5822 or [info@hitachivantara.com](mailto:info@hitachivantara.com)

Europe, Middle East and Africa: +44 (0) 1753 618000 or [info.emea@hitachivantara.com](mailto:info.emea@hitachivantara.com)

Asia Pacific: +852 3189 7900 or [info.marketing.apac@hitachivantara.com](mailto:info.marketing.apac@hitachivantara.com)

