Hitachi NAS Modules for Red Hat® Ansible®

V1.0.0

User Guide

This document describes how to use the Hitachi NAS (HNAS) modules for Red Hat® Ansible®

MK-92HNAS096-00

April 2021

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.

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.

Table of Contents

Preface

About this document

This document describes how to use the Hitachi NAS (HNAS) modules for Red Hat® Ansible.

Document conventions

This document uses the following typographic convention:

Convention	Description
Bold	• Indicates text in a window, including window titles, menus, menu options, buttons, fields, and labels. Example: **Click OK**. • Indicates emphasized words in list items.
Italic	Indicates a document title or emphasized words in text.
`Monospace`	Indicates text that is displayed on screen or entered by the user. Example: `pairdisplay -g oradb`

Intended audience

This document is intended for IT and data center administrators in order to automate and manage the configuration of Hitachi NAS systems. This includes the following platforms:

- Hitachi NAS Platform 5000
- Hitachi NAS Platform 4000
- Hitachi Unified NAS Platform VSP F/G/Nx00

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 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 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, register, and complete your profile.

Chapter 1: Introduction

Overview

Hitachi NAS Modules for Red Hat® Ansible® allow IT and data center administrators to automate and manage some of the configuration of Hitachi NAS systems.

With Hitachi NAS Modules for Red Hat® Ansible®, the administrator can create playbooks together with logic and other Ansible modules to automate complex tasks. Administrators can filter, sort and group the information by piping the output from one module to another.

Hitachi NAS Modules for Red Hat® Ansible® tasks can be executed by running simple playbooks written in yaml syntax and are idempotent. For example, the create virtual server task can be used to create a new virtual server or ensure that the specified server exists. Executing the same task multiple times should produce the same result.

The modules are supported on Ansible version 2.9 and newer, and can be used with Python2 or Python3.

Components

The Hitachi NAS Modules for Red Hat® Ansible® are supplied as a set of Python files which make up the module collection. Some yaml examples, which demonstrate how to use the modules are also included, as is this document.

Chapter 2: System Requirements

Storage platforms

The Hitachi NAS modules for Red Hat® Ansible® require a minimum HNAS REST API version of 7, which 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 the minimum supported version from the table below:

Hitachi storage platform	Software version	Software version with API key support
Hitachi NAS Platform 5000	13.5	13.7
Hitachi NAS Platform 4000	13.5	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)	83-05-33-x0/00	83-05-36-x0/00

Port requirements

All network communications are performed over TCP port 8444, using HTTPS, between the Ansible server and the HNAS system being managed.

Chapter 3: Installation and Configuration

There is no need to install the modules on anything other than the Ansible Server – there is no benefit from running the modules on a remote system.

Installing the Hitachi NAS modules

The Hitachi NAS modules are distributed as an Ansible collection, and should be installed into an existing Ansible installation, using the `ansible-galaxy` tool, as shown below. The installation should be run by the user that will be using the modules:

```
# ansible-galaxy collection install hitachi-hnas-1.0.0.tar.gz
```

By default, the modules will be installed into the following folder, under the current users home directory:

```
.ansible/collections/ansible_collections/hitachi/hnas
```

To install the modules to a different location, the `-p` option can be used with `ansible-galaxy` to specify a different path. This option should be used if multiple users are going to be using the Hitachi NAS modules.

To check the modules are accessible by Ansible, use the `ansible-doc` command below, which should show the documentation for the `hnas_facts` module, and should include the installation location as the first line of output:

```
# ansible-doc hitachi.hnas.hnas_facts
```

In addition to modules, the install process will add some example playbooks under the installation path. Some of these examples are referenced later in this document. The examples playbooks should not be directly run without modification, as they require system specific settings before they can be used.

To remove the Hitachi NAS modules, delete the `hnas` specific folder and all files it contains. To obtain the full path location of the modules, use the `ansible-doc` example above.

Configuring the Hitachi NAS system

All communications between the Ansible server and the HNAS system is done using the HNAS 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.

```
hnas:$ apikey-create "ansible"
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.Nj2TOgxiOYgpTu2kjzEGS4QmIJIeLmF3aXKg6FhY9vC
```

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

```
hnas:$ user add ansible abcd1234 SUPERVISOR
```

If the HNAS system is using self-signed certificates, it will be necessary to set the `validate_certs` option to `false`.

Chapter 4: Hitachi NAS Modules

All parameters returned from the Hitachi NAS modules are of the form **virtualServerId**, whereas all parameters that can't be directly derived from response data, would be ambiguous, or instruct Ansible to perform a specific task are of the form **fact_type**.

The character case of values in string parameters should be used consistently, otherwise unpredictable results may occur. Some HNAS objects allow items to be created with similar names but different cases, others do not. Using inconsistent case may result in the wrong resources being identified, or duplicates being created.

Accessing the modules in a playbook

There are two ways to refer to the modules inside yaml playbooks. Either by using the Fully Qualified Collection Name (FQCN) or by using just the module name, and using an additional `collections` statement. The collections statement is only needed once, and after that, just the module names are needed. See the two examples below:

FQCN example:

```
- hosts: localhost
  tasks:
  - hitachi.hnas.hnas_facts:
    - system_facts
```

Collections example:

```
- hosts: localhost
  collections:
  - hitach.hnas
  tasks:
  - hnas_facts:
    - system_facts
```

The examples in this document use the FQCN instead of specifying the collection name separately.

Restrictions

Due to the potential for user data loss with some of the operations that delete items, there are some situations where a deletion may fail. Performing the delete operations in an appropriate order, rather than attempting to delete a top level item is more likely to succeed. For example, the correct order for deleting items would be:

1. Delete shares/exports
2. Delete filesystems
3. Delete virtual servers
4. Delete storage pools

Deleting a virtual server will not delete any assigned filesystems, but will leave them in the unassigned state, and not accessible via Ansible – this is done intentionally.

Deleting a storage pool must be done after all the filesystems it hosts have been deleted, otherwise the operation will fail – this is done intentionally.

These restrictions are in place to try to avoid accidental data loss.

Common Configuration Parameters

All modules require the following common input parameters, which provide details on how to communicate between the Ansible Server and the HNAS system. Either an API key or username/password combination can be used for authentication, but authentication using an API key is the preferred method, as it allows the HNAS administrator greater control over system access.

Common input parameters:

Name	Type	Required	Description
api_url	String	Yes	The URL of the HNAS REST API. It must be of the form: https://10.1.2.3:8444/v7 which includes the protocol, management IP address, port used, and API version. Note that version 7 is the lowest supported version.
api_key	String	No	API authentication key associated with the HNAS management interface
api_username	String	No	HNAS management username
api_password	String	No	Password associated with the HNAS management username
validate_certs	Bool	No	Whether to validate SSL certificates when communicating between the Ansible Server and the HNAS system. The default value is True.

Get NAS System Facts

Module: hnas_facts

This module can be used to gather details about an HNAS system. It includes physical details and file serving details. In addition to the input parameters below, the connection details covered in the “common input parameters” section, at the beginning of this chapter, must be included.

Input parameters:

Name	Type	Required	Description
fact_type	List	Yes	A list of the type of “facts” to retrieve from the HNAS system – see “fact_type list items” below
data	Dict	No	When retrieving some types of data, additional parameters are required – see “data dictionary items” below

fact_type list items:

Name	Description
system_facts	Gather details about the HNAS cluster, including node information
virtual_server_facts	Gather details about the virtual servers hosted on the cluster
system_drive_facts	Gather details about the system drives visible to the cluster
storage_pool_facts	Gather details about the storage pools hosted on the cluster
filesystem_facts	Gather details about the filesystems hosted on the cluster
nfs_export_facts	Gather details of NFS exports hosted on a particular virtual server
cifs_share_facts	Gather details of SMB/CIFS shares hosts on a particular virtual server
snapshot_facts	Gather a list of snapshots present on s particular filesystem
network_port_facts	Gather a list of the physical network ports available to each cluster node
aggregate_port_facts	Gather a list of the aggregate network ports available to each cluster node

data dictionary items:

Name	Type	Description
filesystemId	String	filesystemId of a filesystem - required when retrieving `snapshot_facts`, otherwise not required
virtualServerId	Int	virtualServerId parameter of a virtual server - required when retrieving `nfs_export_facts` or `smb_share_facts`, otherwise not required

Example 1 – Get HNAS system information:

```
- name: Get Hitachi NAS system information
  hosts: localhost
  vars:
    login: &login
      api_url: https://172.27.5.11:8444/v7
      api_key: BgB2qWZVkE.e53OLShtF3If9UIVdTNmvW9dS7ObPqYNPM83OQoeAj9
      validate_certs: false
  tasks:
  - hitachi.hnas.hnas_facts:
      <<: *login
      fact_type:
        - system_facts
    register: result
  - debug: var=result.ansible_facts
```

Output from example 1:

```
[root@localhost ~]# ansible-playbook hnas_get_system_facts.yml
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'

PLAY [Get Hitachi NAS system information]
*******************************************************************************
****************************************
```

```
TASK [Gathering Facts]
********************************************************************************
***********************************************************
ok: [localhost]

TASK [hitachi.hnas.hnas_facts]
********************************************************************************
***************************************************
ok: [localhost]

TASK [debug]
********************************************************************************
*********************************************************************
ok: [localhost] => {
    "result.ansible_facts": {
        "nodes": [
            {
                "UUID": "48fbe624-4c33-11d0-9001-9c5547075e75",
                "firmwareVersion": "13.9.6813.00",
                "ipAddresses": [
                    "172.27.5.10"
                ],
                "model": "3090-G2",
                "name": "mercury110n1-1",
                "nodeId": 1,
                "objectId": "313a3a3a3a3a3a303a3a3a4f49445f24232140255f56",
                "serial": "M2SEKW1238092",
                "status": "ONLINE"
            }
        ],
        "system": {
            "clusterUUID": "48fbe624-4c33-11d0-9000-9c5547075e75",
            "firmwareVersion": "13.9.6813.00",
            "isCluster": false,
            "licenses": [
                "CIFS",
                "NFS",
                "FILE_CLONE",
                "BASE_DEDUPLICATION",
            ],
            "model": "3090-G2",
            "name": "mercury110n1",
            "nodeCount": 1,
            "storageHealth": "ROBUST",
            "vendor": "HITACHI"
        }
    }
}

PLAY RECAP
********************************************************************************
***********************************************************************
localhost                  : ok=3    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

Example 2 – Get NFS exports for virtual server 1:

```
- name: Get NFS Export details for virtual server 1
  hosts: localhost
  vars:
    login: &login
```

```
      api_url: https://172.27.5.11:8444/v7
      api_key: BgB2qWZVkE.e53OLShtF3If9UIVdTNmvW9dS7ObPqYNPM83OQoeAj9
      validate_certs: false
  tasks:
  - hitachi.hnas.hnas_facts:
      <<: *login
      fact_type:
        - nfs_export_facts
      data:
        virtualServerId: 1
    register: result
  - debug: var=result.ansible_facts
```

Output from example 2:

```
[root@localhost ~]# ansible-playbook hnas_get_nfs_export_facts.yml
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'

PLAY [Get NFS Export details for virtual server 1]
*******************************************************************************
*******************************

TASK [Gathering Facts]
*******************************************************************************
***********************************************************
ok: [localhost]

TASK [hitachi.hnas.hnas_facts]
*******************************************************************************
***************************************************
ok: [localhost]

TASK [debug]
*******************************************************************************
*********************************************************************
ok: [localhost] => {
    "result.ansible_facts": {
        "nfsExports": [
            {
                "filesystemId": "075E4D861BBF9C360000000000000000",
                "name": "/nfse",
                "objectId":
"313a3a3a36333463396164632d306332312d313164372d396437642d3963353534373037356537
353a3a3a303a3a3a4f49445f24232140255f56",
                "path": "/",
                "settings": {
                    "accessConfig": "*(rw)",
                    "localReadCacheOption": "DISABLED",
                    "snapshotOption": "SHOW_AND_ALLOW_ACCESS",
                    "transferToReplicationTargetSetting": "USE_FS_DEFAULT"
                },
                "shareId": "634c9adc-0c21-11d7-9d7d-9c5547075e75",
                "virtualServerId": 1
            }
        ]
    }
}
```

```
PLAY RECAP
********************************************************************************
************************************************************************
localhost                  : ok=3    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

Manage Virtual Servers

Module: hnas_virtual_server

This module manages Hitachi NAS virtual servers. It can be used to ensure that a virtual server does or does not exist. IP addresses can also be added or removed from virtual servers.

If `state=present`, it ensures the existence of a virtual server, or ensures that IP addresses are assigned to a virtual server. If `state=absent`, it ensures that specific IP addresses are not assigned to a virtual server, or if no IP address details are supplied, it ensures that the virtual server is not present.

In addition to the **Input parameters** below, the connection details covered in the “common input parameters” section, at the beginning of this chapter, must be included.

Input parameters:

Name	Type	Required	Description
state	String	Yes	Absent or present
data	Dict	No	When retrieving some types of data, additional parameters are required – see “data dictionary items” below

data dictionary items:

Additional data to describe the specific virtual server. The `name` parameter is required for both create and delete operations. The `address_details` parameter can be used to add and remove multiple addresses from virtual servers. The `clusterNodeId` parameter is only valid for the create operation.

Name	Type	Required	Description
name	String	Yes	Name of the virtual server
clusterNodeId	Int	No	Node that initially hosts the virtual server – default is for node 1
address_details	List	Yes	A list of IP addresses that should be present/absent from a virtual server.

address_details list parameters:

A list of IP addresses that should be present/absent from a virtual server. If `state=present`, `netmask` and `port` need to be supplied for each address. If state=absent, just the address parameter is needed. If the address_details parameter is not present and state=absent, then the virtual server is deleted

Name	Type	Required	Description
address	String	Yes	IP address to be associated with/removed from the virtual server
netmask	String	No	Network mask associated with the address parameter
port	String	No	Aggregate network port for address association

Example 1 – Ensure a virtual server is present on an HNAS system:

```
- name: Create Hitachi NAS virtual server
  hosts: localhost
  gather_facts: false
  vars:
    login: &login
      api_url: https://172.27.5.11:8444/v7
      api_key: BgB2qWZVkE.e53OLShtF3If9UIVdTNmvW9dS7ObPqYNPM83OQoeAj9
  tasks:
  - hitachi.hnas.hnas_virtual_server:
      <<: *login
      state: present
      data:
        name: "evs-ansible"
        address_details:
        - address: "172.27.5.15"
          netmask: "255.255.192.0"
          port: "ag1"
        - address: "172.27.5.16"
          netmask: "255.255.192.0"
          port: "ag1"
    register: result
  - debug: var=result.virtualServer
```

Output from example 1:

```
[root@localhost ~]# ansible-playbook hnas_create_evs.yml
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'

PLAY [Create Hitachi NAS virtual server]
*******************************************************************************
*****************************************

TASK [hitachi.hnas.hnas_virtual_server]
*******************************************************************************
******************************************
ok: [localhost]

TASK [debug]
*******************************************************************************
*********************************************************************
ok: [localhost] => {
    "result.virtualServer": {
        "UUID": "0cf05696-1b0e-11d7-965f-9c5547075e75",
        "ipAddresses": [
            "172.27.5.15"
            "172.27.5.16"
        ],
        "isEnabled": true,
        "name": "evs-ansible",
        "objectId": "333a3a3a3a3a3a303a3a3a4f49445f24232140255f56",
        "status": "ONLINE",
        "type": "File services",
        "virtualServerId": 3
    }
}

PLAY RECAP
********************************************************************************
************************************************************************
localhost                  : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

Example 2 – Ensure a virtual server is not present:

```
- name: Delete Hitachi NAS virtual server
  hosts: localhost
  gather_facts: false
  vars:
    login: &login
      api_url: https://172.27.5.11:8444/v7
      api_key: BgB2qWZVkE.e53OLShtF3If9UIVdTNmvW9dS7ObPqYNPM83OQoeAj9
  tasks:
  - hitachi.hnas.hnas_virtual_server:
      <<: *login
      state: absent
      data:
        name: "evs-ansible"
    register: result
  - debug: var=result.virtualServer
```

Output from example 2:

```
[root@localhost ~]# ansible-playbook hnas_delete_evs.yml
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'

PLAY [Delete Hitachi NAS virtual server]
********************************************************************************
*****************************************

TASK [hitachi.hnas.hnas_virtual_server]
********************************************************************************
******************************************
ok: [localhost]

TASK [debug]
********************************************************************************
**********************************************************************
ok: [localhost] => {
    "result.virtualServer": ""
}
```

```
PLAY RECAP
********************************************************************************
************************************************************************
localhost                  : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

Example 3 – Ensure an IP address is not present on a virtual server:

```
- name: Delete IP address from Hitachi NAS virtual server
  hosts: localhost
  gather_facts: false
  vars:
    login: &login
      api_url: https://172.27.5.11:8444/v7
      api_key: BgB2qWZVkE.e53OLShtF3If9UIVdTNmvW9dS7ObPqYNPM83OQoeAj9
  tasks:
  - hitachi.hnas.hnas_virtual_server:
      <<: *login
      state: absent
      data:
        name: "evs-ansible"
        address_details:
        - address: "172.27.5.15"
    register: result
  - debug: var=result.virtualServer
```

Output from example 3:

```
[root@localhost ~]# ansible-playbook hnas_delete_evs_address.yml
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'

PLAY [Delete HNAS virtual server IP address]
********************************************************************************
*************************************

TASK [hitachi.hnas.hnas_virtual_server]
********************************************************************************
******************************************
ok: [localhost]

TASK [debug]
********************************************************************************
**********************************************************************
ok: [localhost] => {
    "result.virtualServer": {
        "UUID": "b38c5e30-1b33-11d7-977a-9c5547075e75",
        "ipAddresses": [
            "172.27.5.16"
        ],
        "isEnabled": true,
        "name": "ansible",
        "objectId": "333a3a3a3a3a303a3a3a4f49445f24232140255f56",
        "status": "ONLINE",
        "type": "File services",
        "virtualServerId": 3
    }
}
```

```
PLAY RECAP
********************************************************************************
************************************************************************
localhost                  : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

Manage Storage Pools

Module: hnas_storage_pool

This module manages Hitachi NAS storage pools. It allows the creation and deletion of storage pools. The presence of a storage pool allows filesystems to be created.

If `state=present`, it ensures the existence of a storage pool. If `state=absent`, it ensures that a specific storage pool is not present on the server.

Note that deleting a storage pool can destroy user data, and cannot be undone. – can it be deleted with filesystems?

In addition to the input parameters below, the connection details covered in the “common input parameters” section, at the beginning of this chapter, must be included.

Input parameters:

Name	Type	Required	Description
state	String	Yes	Absent or present
data	Dict	No	When retrieving some types of data, additional parameters are required – see “data dictionary items” below

data dictionary items:

Additional data to describe the storage pool. The `label` parameter is required for both create and delete operations. The other parameters are only required for the create operation.

Name	Type	Required	Description
label	String	Yes	The label or name of the storage pool
chunkSize	Int	No	The chunk size determines the ultimate scalability of the storage pool and its filesystems. It also controls the size of the increments in which space can be added to filesystems. The `chunkSize` cannot be changed once a storage pool has been created. Default value is 19327352832 (19GB)
systemDrives	List	Yes	A list of system drive ID values. The minimum number of system drives needed to create a storage pool is 4, and the maximum is 16
allow_denied_system_drives	Bool	No	Allows the use of system drives that currently are denied access

Example1 – Create a storage pool:

```
- name: Create HNAS storage pool
  hosts: localhost
  gather_facts: false
  vars:
    login: &login
      api_url: https://172.27.5.11:8444/v7
      api_key: BgB2qWZVkE.e53OLShtF3If9UIVdTNmvW9dS7ObPqYNPM83OQoeAj9
      validate_certs: false
  tasks:
  - hitachi.hnas.hnas_storage_pool:
      state: present
      <<: *login
      data:
        label: "ansible-pool"
        systemDrives: [ 16, 17, 18, 19, 20, 21 ]
    register: result
  - debug: var=result.storagePool
```

Output from example 1:

```
root@localhost ~]# ansible-playbook hnas_create_storage_pool.yml
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'

PLAY [Create HNAS storage pool]
*******************************************************************************
**************************************************

TASK [hitachi.hnas.hnas_storage_pool]
*******************************************************************************
********************************************
ok: [localhost]

TASK [debug]
*******************************************************************************
*********************************************************************
ok: [localhost] => {
    "result.storagePool": {
        "chunkSize": 19327352832,
        "freeCapacity": 2145310998528,
        "isAssignedToLocalCluster": true,
        "isFilesystemExpansionAllowed": true,
        "isHealthy": true,
        "isTiered": false,
        "label": "ansible-pool",
        "objectId":
"3533313334363535303835333238303734353a3a3a3a3a303a3a3a4f49445f24232140255f56
",
        "storagePoolId": 531346550853280745,
        "totalCapacity": 2145310998528,
        "usedCapacity": 0
    }
}

PLAY RECAP
*******************************************************************************
***********************************************************************
```

```
localhost                  : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

Example 2 – Delete a storage pool:

```
- name: Delete HNAS storage pool
  hosts: localhost
  gather_facts: false
  vars:
    login: &login
      api_url: https://172.27.5.11:8444/v7
      api_key: BgB2qWZVkE.e53OLShtF3If9UIVdTNmvW9dS7ObPqYNPM83OQoeAj9
      validate_certs: false
  tasks:
    - hitachi.hnas.hnas_storage_pool:
        state: absent
        <<: *login
        data:
          label: "ansible-pool"
      register: result
    - debug: var=result.storagePool
```

Output of example 2:

```
[root@localhost ~]# ansible-playbook hnas_delete_storage_pool.yml
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'

PLAY [Delete HNAS storage pool]
*******************************************************************************
**************************************************

TASK [hitachi.hnas.hnas_storage_pool]
*******************************************************************************
********************************************
ok: [localhost]

TASK [debug]
*******************************************************************************
*********************************************************************
ok: [localhost] => {
    "result.storagePool": ""
}

PLAY RECAP
*******************************************************************************
***********************************************************************
localhost                  : ok=2    changed=0    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

Manage Filesystems

Module: hnas_filesystem

This module manages Hitachi NAS filesystems. It can be used to create or delete filesystems. It can also be used to expand an existing filesystem, by increasing the capacity. The state of filesystem can be set to mounted or unmounted by setting the `status` parameter appropriately.

If `state=present`, it ensures the existence of a filesystem, with the requested `status`, and that it is at least the requested `capacity`. If `state=absent`, it ensures that specific filesystem is not present on the server.

Note that deleting a filesystem will destroy user data, and cannot be undone.

In addition to the input parameters below, the connection details covered in the "common input parameters" section, at the beginning of this chapter, must be included.

Input parameters:

Name	Type	Required	Description
state	String	Yes	Absent or present
data	Dict	No	When retrieving some types of data, additional parameters are required – see "data dictionary items" below

data dictionary items:

Additional data to describe the filesystem. The `label` parameter is required for all operations, and is the only parameter required for the delete operation. Either the `storage_pool_name` or `storagePoolId` parameter needs to be specified for all operations where `state=present`. Either the `virtual_server_name` or `virtualServerId` parameter needs to be specified for all operations where `state=present`. The `capacity` value is required for all operations where `state=present`.

Name	Type	Required	Description
label	String	Yes	The label or name of the filesystem
storage_pool_name	String	No	The name of the storage pool that should contain the filesystem
storagePoolId	Int	No	The ID of the storage pool that should contain the filesystem
virtual_server_name	String	No	The name of the virtual server that should host the filesystem
virtualServerId	Int	No	The ID of the virtual server that should host the filesystem
capacity_unit	String	No	The unit to use as a multiplier for the `capacity` value. The value can be one of - b, bytes, k, kb, kib, m, mb, mib, g, gb, gib, t, tb, tib. The default value is 'bytes'

capacity	Int	No	The minimum capacity of the filesystem. The `capacity` value will be dependent on the `chunkSize` of the storage pool. See *hnas_storage_pool* module details. The `capacity` value should be used in conjunction with the `capacity_unit` value.
status	String	No	The required status of the filesystem – it can be either 'MOUNTED' or 'NOT_MOUNTED'. The default value is ‘MOUNTED’
blockSize	Int	No	The filesystem block size to be used to format an unformatted/new filesystem. **Note that an existing filesystem will NOT be reformatted with a different block size**. The default value is ‘4’

Example 1 – Ensure a filesystem exists, or expand an existing filesystem:

```
- name: Create or Expand an HNAS filesystem
  hosts: localhost
  gather_facts: false
  vars:
    login: &login
      api_url: https://172.27.5.11:8444/v7
      api_key: BgB2qWZVkE.e53OLShtF3If9UIVdTNmvW9dS7ObPqYNPM83OQoeAj9
      validate_certs: false
  tasks:
  - hitachi.hnas.hnas_filesystem:
      state: present
      <<: *login
      data:
        label: "ansible"
        virtualServerId: 1
        storage_pool_name: "Span0"
        capacity_unit: gib
        capacity: 20
    register: result
  - debug: var=result.filesystem
```

Output from example 1:

```
[root@localhost ~]# ansible-playbook hnas_create_fs.yml
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'

PLAY [Create or expand an HNAS filesystem]
*******************************************************************************
***************************************

TASK [hitachi.hnas.hnas_filesystem]
*******************************************************************************
**********************************************
changed: [localhost]

TASK [debug]
*******************************************************************************
*********************************************************************
ok: [localhost] => {
    "result.filesystem": {
        "blockSize": 4096,
        "capacity": 19293798400,
        "expansionLimits": 18446744073709551615,
        "filesystemId": "075FA5699D6C5B4F0000000000000000",
        "freeCapacity": 16986300416,
        "isDedupeEnabled": false,
        "isDedupeSupported": false,
        "isLogicalCapacityFreeCapacityValid": true,
        "isLogicalCapacityValid": true,
        "isNDMPRecoveryTarget": false,
        "isNonStrictWORM": false,
        "isObjectReplicationTarget": false,
        "isReadCached": false,
        "isReadOnly": false,
        "isSysLocked": false,
        "isThinProvisioningEnabled": false,
        "isThinProvisioningEnabledValid": true,
        "isTrueSparseFileEnabled": true,
        "isTrueSparseFileEnabledValid": true,
        "isUnlimitedExpansion": true,
        "isWORM": false,
        "label": "ansible",
        "logicalCapacity": 19293798400,
        "logicalFreeCapacity": 16986300416,
        "objectId":
"303735464135363939443643354234463030303030303030303030303030303a3a3a303a3a3a
4f49445f24232140255f56",
        "status": "MOUNTED",
        "storagePoolId": 530991019344831952,
        "usedCapacity": 2307497984,
        "virtualServerId": 1
    }
}

PLAY RECAP
********************************************************************************
************************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

Example 2 – Ensure a filesystem does not exist:

```
- name: Delete HNAS filesystem
  hosts: localhost
  gather_facts: false
  vars:
    login: &login
      api_url: https://172.27.5.11:8444/v7
      api_key: BgB2qWZVkE.e53OLShtF3If9UIVdTNmvW9dS7ObPqYNPM83OQoeAj9
      validate_certs: false
  tasks:
    - hitachi.hnas.hnas_filesystem:
        state: absent
        <<: *login
        data:
          label: "ansible"
      register: result
    - debug: var=result.filesystem
```

Output from example 2:

```
[root@localhost ~]# ansible-playbook hnas_delete_fs.yml
```

```
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'

PLAY [Delete HNAS filesystem]
********************************************************************************
****************************************************

TASK [hitachi.hnas.hnas_filesystem]
********************************************************************************
*********************************************
changed: [localhost]

TASK [debug]
********************************************************************************
**********************************************************************
ok: [localhost] => {
    "result.filesystem": ""
}

PLAY RECAP
********************************************************************************
************************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

Manage CIFS/SMB Shares and NFS Exports

Module: hnas_share_export

This module manages CIFS/SMB shares and NFS exports on Hitachi NAS servers. They can be created and deleted, and also updated by changing the supplied parameters. For CIFS/SMB shares, the share access authentications can also be updated.

In addition to the input parameters below, the connection details covered in the “common input parameters” section, at the beginning of this chapter, must be included.

Input parameters:

Name	Type	Required	Description
state	String	Yes	Absent or present
data	Dict	No	When retrieving some types of data, additional parameters are required – see “data dictionary items” below

data dictionary items – common for NFS Exports and CIFS/SMB Shares:

Additional data to parameters which are common to both CIFS/SMB shares or NFS exports. The `name`, `virtualServerId` and `type` parameters are required for all operations. The other parameters are only required for operations where `state=present`. If `type=nfs` the `filesystemPath` should be in UNIX format e.g. '/folder/sub-folder', but if `type=cifs` it should be in Windows format, with the \ character escaped e.g. '\\folder\\sub-folder'

Name	Type	Required	Description

name	String	Yes	name of the share/export
type	String	Yes	Type can be either ‘nfs’, which refers to an NFS export or ‘cifs’ to refer to a CIFS/SMB share
virtualServerId	Int	Yes	The `virtualServerId` parameter of the virtual server that hosts the share/export
filesystemId	Int	No	The `filesystemId` of the filesystem associated with the share/export
filesystemPath	String	No	The filesystem location that is exported by the share/export
accessConfig	String	No	Set the client access restrictions for the share/export. By default all clients have read and write access.
snapshotOption	String	No	Sets the accessibility and visibility of the snapshot directory. The value can be one of - ‘HIDE_AND_DISABLE_ACCESS’, ‘HIDE_AND_ALLOW_ACCESS’, ‘SHOW_AND_ALLOW_ACCESS’ The default value is ‘SHOW_AND_ALLOW_ACCESS’
transferToReplicationTargetSetting	String	No	Sets whether the share or export should be brought online when the replication target of this shares/exports file system is converted to read-write. The value can be one of – ‘DO_NOT_TRANSFER’, ‘TRANSFER’, ‘USE_FS_DEFAULT’, ‘INVALID’. The default value is ‘USE_FS_DEFAULT’

data dictionary items – NFS Export specific:

Name	Type	Required	Description
localReadCacheOption	String	No	Sets which files are candidates for read caching. The value can be one of – ‘DISABLED’, ‘ENABLED_FOR_ALL_FILES’, ‘ENABLED_FOR_TAGGED_FILES’, ‘ENABLED_FOR_CVLS’. The default value is ‘DISABLED’

data dictionary items – CIFS/SMB Share specific:

Name	Type	Required	Description
comment	String	No	A comment associated with the share.
userHomeDirectoryPath	String	No	Per-user home directories will be created using this path, relative to the share root.
isScanForVirusesEnabled	Bool	No	If virus scanning is enabled, scan files accessed via this share for viruses. The default value is False.
maxConcurrentUsers	Int	No	Controls the maximum allowed connections. -1 allows unlimited client connections. The default value is -1.

cacheOption	String	No	Specifies the share's cache options. The value can be one of – ‘MANUAL_CACHING_DOCS’, ‘AUTO_CACHING_DOCS’, ‘AUTO_CACHING_PROGS’, ‘CACHING_OFF’. The default value is ‘MANUAL_CACHING_DOCS’
userHomeDirectoryMode	String	No	Set the share's home directory behavior. The value can be one of – ‘OFF’, ‘ADS’, ‘USER’, ‘HIDDEN_USER’, ‘DOMAIN_AND_USER’, ‘UNIX’. The default value is ‘OFF’
isFollowSymbolicLinks	Bool	No	If symlinks are encountered when browsing this share, follow them automatically on the server. Note that if client side symlink handling is enabled, this setting does not affect clients using the SMB2 or SMB3 protocols. Default value is False.
isFollowGlobalSymbolicLinks	Bool	No	Allow clients that are connected to this share to follow global symlinks. Default value is False.
isForceFileNameToLowercase	Bool	No	Convert the names of all files and directories created to lower case. Default value is False.
isABEEnabled	Bool	No	Enable Access-based Enumeration, which makes visible only those files or folders that the user has rights to access. Default value is False.
cifsAuthentications	List	No	A list of share access authentication entries that must be `present` or `absent` from the share. See “cifsAuthentications list parameters” below.

cifsAuthentications list parameters:

When adding or checking for the existence of a user/group `(state=present)`, all parameters are required. When removing items `(state=absent)`, only the `name` parameter is needed.

Name	Type	Required	Description
name	String	Yes	The name of the Windows user, group or SID to associate the access requirements with.
permission	Integer	No	Bit representation of the permissions for access authentication. The following values should be used to grant the appropriate access: ▪ 0 no permission ▪ 8 grant read access ▪ 24 grant read and change access ▪ 56 grant read, change and full control ▪ 1 deny read access ▪ 3 deny read and change access ▪ 7 deny read, change and full control
type	String	No	The type of the user/group/SID specified by name. The value can be one of 'ALIAS', 'COMPUTER', 'DELETED', 'DOMAIN', 'GROUP', 'INVALID', 'UNKNOWN', 'USER', 'WELLKNOWN'

Example 1 – Ensure a CIFS/SMB share exists:

```
- name: Create HNAS CIFS share
  hosts: localhost
  gather_facts: false
  vars:
    login: &login
      api_url: https://172.27.5.11:8444/v7
      api_key: BgB2qWZVkE.e53OLShtF3If9UIVdTNmvW9dS7ObPqYNPM83OQoeAj9
      validate_certs: false
  tasks:
  - hitachi.hnas.hnas_share_export:
      state: present
      <<: *login
      data:
        name: "ansible-cifs-share"
        virtualServerId: 2
        filesystemId: "075E7582C745AEA10000000000000000"
        filesystemPath: "\\home\\frank"
        type: "cifs"
    register: result
  - debug: var=result.cifsShare
```

Output from example 1:

```
[root@localhost ~]# ansible-playbook hnas_create_share.yml
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'

PLAY [Create HNAS CIFS share]
*******************************************************************************
****************************************************

TASK [hitachi.hnas.hnas_share_export]
*******************************************************************************
********************************************
changed: [localhost]

TASK [debug]
*******************************************************************************
*********************************************************************
ok: [localhost] => {
    "result.cifsShare": {
        "filesystemId": "075E7582C745AEA10000000000000000",
        "name": "ansible-cifs-share",
        "objectId":
"323a3a3a353363646532313823d316633322d313164372d393964632d396335353437303735653
7353a3a3a303a3a3a4f49445f24232140255f56",
        "path": "\\home\\frank",
        "settings": {
            "accessConfig": "",
            "cacheOption": "MANUAL_CACHING_DOCS",
            "comment": "",
            "isABEEnabled": false,
            "isFollowGlobalSymbolicLinks": false,
            "isFollowSymbolicLinks": false,
            "isForceFileNameToLowercase": false,
            "isScanForVirusesEnabled": false,
            "maxConcurrentUsers": -1,
            "snapshotOption": "SHOW_AND_ALLOW_ACCESS",
            "transferToReplicationTargetSetting": "USE_FS_DEFAULT",
            "userHomeDirectoryMode": "OFF",
            "userHomeDirectoryPath": ""
        },
        "shareId": "53cde218-1f32-11d7-99dc-9c5547075e75",
        "virtualServerId": 2
    }
}
```

Example 2 – Ensure a CIFS/SMB share does not exist:

```
- name: Delete HNAS CIFS share
  hosts: localhost
  gather_facts: false
  vars:
    login: &login
      api_url: https://172.27.5.11:8444/v7
      api_key: BgB2qWZVkE.e53OLShtF3If9UIVdTNmvW9dS7ObPqYNPM83OQoeAj9
      validate_certs: false
  tasks:
    - hitachi.hnas.hnas_share_export:
        state: absent
        <<: *login
        data:
          name: "ansible-cifs-share"
          virtualServerId: 2
          type: "cifs"
      register: result
    - debug: var=result.cifsShare
```

Output from example 2:

```
[root@localhost ~]# ansible-playbook hnas_delete_share.yml
[WARNING]: provided hosts list is empty, only localhost is available. Note that
the implicit localhost does not match 'all'

PLAY [Delete HNAS CIFS share]
*******************************************************************************
****************************************************

TASK [hitachi.hnas.hnas_share_export]
*******************************************************************************
********************************************
changed: [localhost]

TASK [debug]
*******************************************************************************
*********************************************************************
ok: [localhost] => {
    "result.cifsShare": ""
}

PLAY RECAP
*******************************************************************************
***********************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

Chapter 5: Troubleshooting

As a first step to troubleshooting any issues, ensure the HNAS REST API is enabled, as described in Chapter 3, and that the Ansible server is able to ping the IP address of that is being used for communications.

Ansible Unable to Connect to HNAS

The connection parameters and credentials can be checked using a simple curl command, run on the Ansible Server machine. Run the following command, and replace the `<username>`, `<password>` and `<admin_IP>` parameters with the ones to be used by Ansible. 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/v7/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/v7/storage/nodes
```

For more details on troubleshooting connectivity issues to the REST API, refer to the *"Hitachi NAS File Storage, REST API Reference - MK-92HNAS088"*.

SSL Certificate Verification Failure

If any error messages returned contain a message similar to the following:

```
[[SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed
```

This means that the SSL certificate validation was not successful. To solve this issue, either add the `validate_certs: false` option to the playbook and see if it works, or ensure that the correct certificates are installed on the HNAS server. The command below can be used to provide additional information about certificate validation.

```
# curl -v -H "X-Api-Key: <api_key>" https://<admin_IP>:8444/v7/storage/nodes
```

Hitachi Vantara

Corporate Headquarters 2535 Augustine Drive
Santa Clara, CA 95054 USA www.HitachiVantara.com community.HitachiVantara.com

Regional Contact Information
Americas: +1 866 374 5822 or info@hitachivantara.com
Europe, Middle East and Africa: +44 (0) 1753 618000 or info.emea@hitachivantara.com
Asia Pacific: +852 3189 7900 or info.marketing.apac@hitachivantara.com