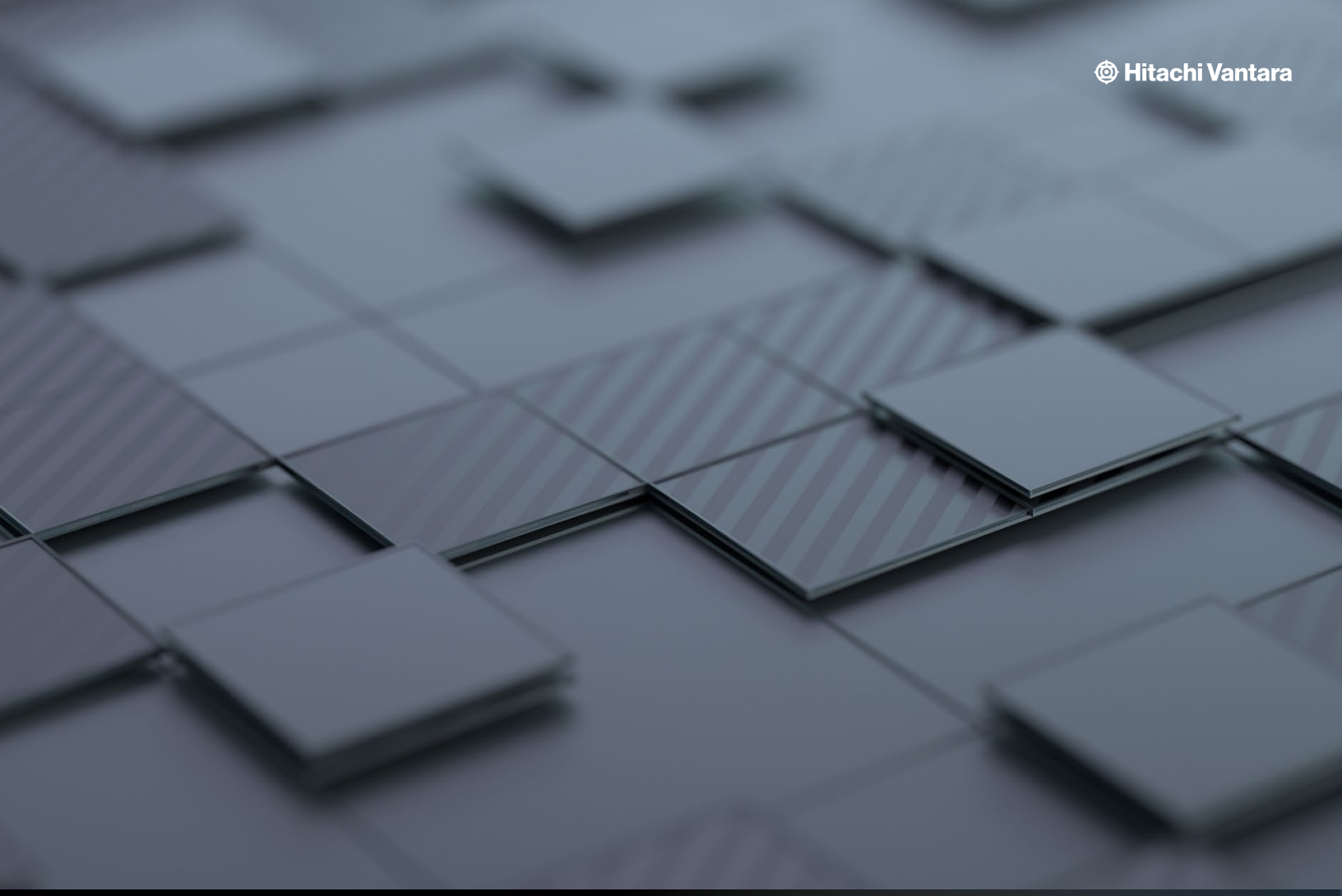


Hitachi Vantara

Ansible Automation for VSP E1090 Storage System and AWS EC2

Powered by Hitachi Cloud Connect for Equinix

Hitachi Vantara
January 2024

Hitachi Vantara

Table of Contents

Notices and Disclaimer 3

About This Guide 4

Intended Audience 4
Document Revisions 4
References 4
Comments 4

Executive Summary 5

Introduction 6

Solution Overview 6
Benefits 6
Key Components 6

Validation 8

Validation Method 8
High Level Diagram 8
Hardware and Software 9
Test Scenarios 10

Guidelines and Recommendations 12

Validation Results 13

Test 1: Prepare the Environment 13
Test 2: Create and Expand the Pool 15
Test 3: Create an iSCSI Target and Add the Host 16
Test 4: Provision the Volume 17
Test 5: Expand the DP Volume 18
Test 6: Add a CHAP User 18
Test 7: Add a New LUN 19

Test 8: Add a Host Nickname 19
Test 9: Change the CHAP Authentication Method 19
Test 10: Create an ADR Volume and Toggle Between ADR and Non-ADR Setting 20
Test 11: SI Pair Operations 21
Test 12: TI Pair Operations 24
Test 13: Volume-less TI PAIR operations 27
Test 14: Perform Resync and Restore operations of the TI Pair 28
Test 15: Revert All TI Operations 29
Test 16: Delete the Host 30
Test 17: Delete Volume Operation 30

Notices and Disclaimer

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 (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 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.

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, BCPii™ 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 screenshots 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.

About This Guide

This how-to white paper provides instructions for automating Hitachi Virtual Storage Platform (VSP) storage provisioning tasks in a hybrid cloud environment using Ansible with Hitachi Ops Center API Configuration Manager (CM REST).

Note: The information shared here is specific to our requirements. It can be used as a guideline or a starting point; however, you can conduct a proof-of-concept in a non-production, isolated test environment matching your production environment before implementing this solution.

Intended Audience

This document is intended for Hitachi Vantara staff and IT professionals of Hitachi Vantara customers and partners who are responsible for planning and deploying such solutions.

Document Revisions

Revision Number	Date	Author	Details
v1.0	January 2024	Hitachi Vantara LLC	Initial Release

References

- Ansible Playbook Documentation: https://docs.ansible.com/ansible/latest/index.html
- AWS Documentation: https://docs.aws.amazon.com
- Hitachi Configuration Manager REST API: https://knowledge.hitachivantara.com/Documents/Management_Software/Ops_Center/API_Configuration_Manager/10.9.x

Comments

Send any comments on this document to GPSE-Docs-Feedback@hitachivantara.com. Include the document title, including the revision level, and refer to specific sections and paragraphs whenever possible. All comments become the property of Hitachi Vantara Corporation.

Thank you.

Executive Summary

This how-to white paper provides instructions for automating VSP storage provisioning tasks in a hybrid cloud environment using Ansible with Hitachi Ops Center API Configuration Manager. Ansible is a widely used tool for configuration management that enables automation of the deployment, configuration, and management of IT infrastructure and applications. CM REST is a standalone module from the Hitachi Ops Center suite that enables automated management of Hitachi storage systems. It uses a REST API for getting information about storage systems or changing the configuration of storage systems.

The procedures documented in this paper were specifically executed in a hybrid cloud environment to demonstrate the flexibility of the **Hitachi Cloud Connect for Equinix** solution. This offering allows clients to locate Hitachi products such as the VSP storage systems at Equinix International Business Exchange™ (IBX) data centers worldwide. Moreover, there is an option for clients to procure this solution through one agreement and invoice, greatly simplifying and accelerating their time to market. By using Equinix IBX data centers and Equinix Fabric™ to interconnect sources of data to applications, organizations can locate their data residing on VSP storage systems next to clouds to leverage hybrid- or multi-cloud capabilities while still maintaining physical control of the data.

If you want to discuss hosting these types of solutions at Equinix, contact your Hitachi Vantara sales team. For more information, visit the Hitachi Cloud Connect for Equinix webpage at: https://hitachivantara.com/en-us/products/storage/flash-storage/cloud-connect-for-equinix.html.

Introduction

Combining Ansible with Hitachi Ops Center API Configuration Manager can greatly streamline the management of IT infrastructure. Ansible is a widely used tool for configuration management that enables automation of the deployment, configuration, and management of IT infrastructure and applications. CM REST is a standalone module from the Hitachi Ops Center suite that enables automated management of Hitachi storage systems. It uses a REST API for getting information about storage systems or changing the configuration of storage systems. By integrating CM REST with Ansible, you can manage on-premises VSP storage systems and provision enterprise-grade storage to applications in on-premise or hybrid- and multi-cloud destination.

Solution Overview

Ansible is an open-source tool for automating IT infrastructure configuration and deployment. To automate tasks, Ansible uses playbooks that consist of a series of instructions written in YAML format. Ansible playbooks offer benefits such as:

- Simple and declarative syntax that facilitates the definition of tasks and configurations.
- Scalability to manage large and distributed systems with ease.
- A rich set of built-in modules and plugins that enhance the extensibility.
- An active community of users and contributors that share best practices and tips for using the tool effectively.

Additionally, you can customize Ansible playbooks to suit different use cases, such as server configuration, application deployment, and network automation, among others. By automating routine tasks, you can improve the efficiency, reduce costs, and increase productivity, making the Ansible playbook an essential tool for any modern IT infrastructure.

Hitachi Ops Center API Configuration Manager provides a method to manage Hitachi storage systems, network devices, and servers from a single interface. Essentially, it allows administrators to automate the configuration and management of Hitachi devices.

Ansible playbooks can be integrated with CM REST by creating Ansible modules that use the CM REST API. Ansible modules are reusable units of code that can be called from Ansible playbooks. These modules can perform tasks such as creating, modifying, or deleting resources in Hitachi devices.

Benefits

The following describes the benefits of Ansible:

- Automates tasks that would otherwise require manual intervention, reducing the time and resources required for IT operations. This results in lower operational costs and enables IT teams to focus on strategic activities.
- Provides consistent and standardized completion of tasks, reducing errors and improving quality. Additionally, it allows for easy scalability to manage large and complex IT environments, enabling organizations to quickly adapt to changing business needs and market conditions.

The following describes the benefits of CM REST:

- Enables storage administrators to automate configuration, management, and monitoring tasks, reducing the need for manual intervention and improving operational efficiency.
- Enables the creation of customized automation workflows and the integration of Hitachi storage systems with various IT tools and platforms, providing flexibility and scalability to keep up with the changing demands of IT environments.
- Enables organizations to scale their IT infrastructure management operations by automating tasks. As a result, additional devices and systems can be managed without increasing the IT team size, which reduces manual labor and leads to significant cost savings over time.

Key Components

The following lists the major components of the solution. For specifications, see the Hardware and Software section.

- Ansible Playbooks: Configuration files written in YAML format that describe a set of tasks to complete.
- Hitachi Ops Center API Configuration Manager: A self-contained, agile module that facilitates the automated management of Hitachi storage systems.
- VSP Storage System: A VSP E1090 storage system provided storage capacity for hybrid-cloud consumers.
- Network Switch: Cisco Nexus 9000 Series switch was used to connect to AWS Direct Connect. The following accessories are required for establishing a WAN between the two sites:
  - 10/25Gbase-LR-S Optics: Long Range transceivers are required to connect long distances.

  - Single-Mode Fiber Cables: Required for long-distance communications.
- Equinix Fabric: Connected equipment at the Equinix near-cloud data center to AWS cloud.
- AWS Cloud: Equipment at Equinix was connected to AWS cloud using a 10 Gbps Direct Connect link. On AWS, a Virtual Private Cloud was created in the region us-west-1.

Validation

This section describes the method, test environment, hardware and software, and test scenarios used in the validation.

Validation Method

To validate the solution, an Ansible control node and a CM REST server were used. All module utility files, Ansible playbook tasks, and parameter files were written to the Ansible node before each operation.

After running the playbook tasks, we verified whether the task result was reflected on either the VSP E1090 storage system or storage consumer.

The test environment was prepared as follows:

- On-premises data center at Equinix SV5 in San Jose, CA:
  - Provisioned a 2 TB volume from the VSP E1090 storage system to serve as a VMware datastore.
  - Deployed one Red Hat Enterprise Linux (RHEL) 8.6 virtual machine on the datastore. Installed Ansible control node.
  - Deployed one RHEL 8.6 virtual machine on the datastore. Installed CM REST.
- AWS cloud using us-west-1 region (Northern California):
  - Deployed one RHEL 9.0 EC2 instance as storage consumer.

High Level Diagram

Figure *1* shows the test environment used to run the validation.

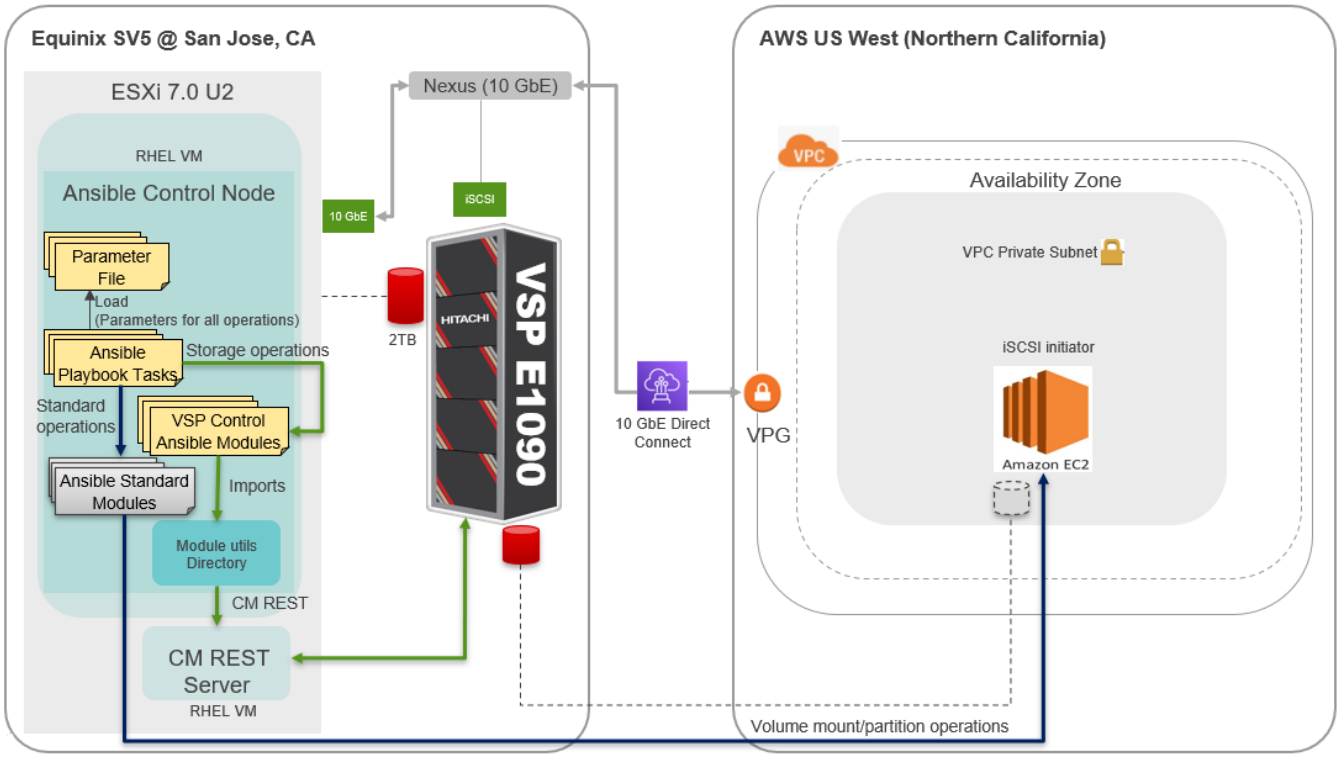


Figure 1: *High Level Diagram*

Hardware and Software

Table 1 provides the hardware specifications used in this validation.

Table 1. *Hardware Components*

Item	Description	Version	Function
Hitachi VSP E1090	1 TB cache (2) 32-core MPUs (3) RAID6 6D-2P parity groups (2) 10 GbE iSCSI ports	SVOS 9.8.2 93-06-42-80/01	Primary storage system
Hitachi Advanced Server DS220	(2) 18-core Intel Xeon Gold 6140 @ 2.3 GHz 128GB memory (1) 10GbE for management & iSCSI communication	BMC 4.70.06 BIOSS5BH3B22.H00	4-node primary vSphere cluster and attached to the VSP E1090 storage system.
Configuration Manager REST API Server	Virtual Machine: (4) Virtual CPUs 4 GB RAM 100 GB virtual disk	VM Version 19 Guest OS: RHEL 8.6 (64-bit)	Configuration Manager API server at the Equinix data center. Managed all activity by Ansible playbook to the VSP E1090 storage system.
Ansible Node	Virtual Machine: (4) Virtual CPUs 4 GB RAM 100 GB virtual disk	VM Version 19 Guest OS: RHEL 8.6 (64-bit)	Ansible control node at the Equinix data center. Completed all Ansible playbook tasks for the VSP E1090 storage system and test servers.
Cisco Nexus C93180YC-FX	(48) 1/10/25-Gbps fibre ports (6) 40/100-Gbps QSFP28 ports	NXOS 9.3(4)	Provided network connectivity between equipment at Equinix and AWS cloud.
Storage Consumer	Instance type: t2.micro	AMI: RHEL-9.0.0_HVM-20230313-x86_64-43-Hourly2-GP2	EC2 instance used as the storage consumer for Ansible playbook tasks.

Table 2 provides the software specifications used in this validation.

Table 2. *Software Components*

Item	Version	Function
VMware vSphere	ESXi 7.0 U2 (17867351)	Hypervisor operating system at the Equinix data center.
VMware vCenter Server Appliance	7.0 U3 (18700403)	Management interface for vSphere clusters at the Equinix data center.
RHEL	RHEL 8.6	Guest operating system of 2x VMs for Configuration Manager REST API Server and Ansible control node. Virtual CPUs: 4 Virtual Memory: 4 GB OS Disk: 100 GB

Test Scenarios

Table 3 lists the test scenarios performed in the validation.

Table 3. *Test Scenarios*

Test	Description	Success Criteria
1	Prepare the environment: 1. Deploy the Configuration Manager REST API Server: a. Provision one 2 TB volume from the VSP E1090 storage system to use as datastore. b. Create one RHEL 8.6 virtual machine on the datastore. c. Install Configuration Manager REST API on the virtual machine. d. Register the storage system. e. Verify whether the storage system is accessible. 2. Deploy the Ansible Server: a. Provision one 2 TB volume from the VSP E1090 storage system to use as datastore. b. Create one RHEL 8.6 virtual machine on the datastore. c. Install or upgrade Ansible* on the virtual machine. d. Configure Ansible node to work with the Configuration Manager REST API Server and test server EC2 instance. e. Configure modules, module utility, parameter files, and playbook tasks. 3. Deploy an RHEL 9.0 virtual machine as an EC2 instance for the test server and configure the iSCSI initiator. **Note**: The Ansible core package is already included in RHEL 8.6 version. You must upgrade to the latest ansible core version. For more information, see: https://access.redhat.com/articles/6325611.	Environment is set up as per the specifications.
2	Create and expand pool.	Pool creation and expansion is successful.
3	Create an iSCSI Target and add the host.	iSCSI target creation and host iqn added for the EC2 instance is successful.
4	Provision the volume: 1. Create a DP-VOL. 2. Add a LUN path.	DP-VOL is created and attached to the iSCSI target successfully.
5	Expand the DP volume.	DP volume expansion is successful.
6	Add a CHAP user.	Chap user addition ends normally, and all the LUNs are visible on the EC2 instance (one way chap).
7	Add a new LUN.	Added new DP-VOL to the existing iSCSI target and detected successfully. Test case 4 task is used again.
8	Add a host nickname.	Added a host nickname for the specified iSCSI target.
9	Change the CHAP authentication method.	Added a target-side CHAP user to an existing iSCSI Target and set the iSCSI target to mutual.

Test	Description	Success Criteria
10	Create an ADR Volume and toggle between ADR and Non-ADR setting.	Operations of ADR volume is successful.
11	ShadowImage copy operations: 1. Create an SI PAIR and map the Volumes to the host groups. 2. Split the SI PAIR. 3. Mount the S-VOL to the EC2 host. 4. Resync the SI PAIR. 5. Restore the SI PAIR.	All SI PAIR operations are successfully completed.
12	Thin Image copy operations: 1. Create a TI PAIR with S-VOL. 2. Split the TI PAIR. 3. Map the secondary volume to the host group. 4. Mount the S-VOL to the EC2 host.	All TI PAIR operations are successfully completed.
13	Volume-less Thin Image copy operations: 1. Create a volume-less TI PAIR. 2. Split the TI PAIR. 3. Map the snapshot data to the S-VOL. 4. Map the secondary volume to the host group. 5. Mount the S-VOL to the EC2 host.	All volume-less TI PAIR operations are successfully completed.
14	Perform resync and restore operations of the Thin Image pair.	Resync and restore operation of the TI PAIR is successfully completed.
15	Revert all TI operations: 1. Unmount the S-VOL from the host. 2. Delete the S-VOL LUN path from the iSCSI target. 3. Unmap the snapshot data. 4. Delete the TI PAIR.	All revert operations completed successfully.
16	Delete the host.	Host iqn deletion from the iSCSI target is successful.
17	Delete volume operation: 1. Delete the LUN path. 2. Delete the iSCSI Target. 3. Delete the DP volume.	LUN paths, iSCSI target, and DP-VOL are deleted.

Guidelines and Recommendations

This section describes the lessons learned from this validation, along with guidelines and recommendations.

- Ensure the various components involved, including the Configuration Manager REST API server and Ansible node, resolve hostnames through a common DNS server.
- Ensure that the network ports required by the Configuration Manager REST API server are open on the firewall of the Configuration Manager REST API server. For a list of port requirements, see the *Hitachi Ops Center API Configuration Manager* document.
- Always use the parameters that are well defined in the module utility; otherwise, any arbitrary parameters do not work. Each task requires pre-defined parameters for storage provisioning through the Configuration Manager REST API server.

Validation Results

This section shows specific steps and screenshots for each test scenario.

Test 1: Prepare the Environment

This test case describes the configuration of the components used in the validation.

Note: It is important to ensure that the various components involved, including the Configuration Manager REST API server and Ansible node, can resolve hostnames through a common DNS server, and that the necessary network ports are opened on the firewall.

1. Install Configuration Manager REST API on the virtual machine:
   a. Download the Configuration Manager Rest API installation media from the following link: https://support.hitachivantara.com/en/user/answers/downloads/downloads-detail.html?d=Ops%20Center%20Installation%20Media&pptype=Software%20Version.
   b. Mount the media.
   c. Navigate to Linux directory and find the installer.
   d. Run the following command:
   ```
   ./install.sh
   ```
   e. When prompted, specify the required information.
   If the installation is successful, the following message is displayed:
   ```
   Configuration Manager REST API installation completed successfully.
   ```
2. Register the storage system by running the following command:
   ```
   curl -v -H "Accept:application/json" -H "Content-Type:application/json" -u username:password -
   X POST -d " { \"model\" : \"VSP E1090\",\"serialNumber\" : \"7xxxxxx\", \"ctl1Ip\" :
   \"172.xx.xx.xx\",\"changeNotificationSetting\" : {\"isNotifiable\" : false }}"
   http://172.xx.xx.xx:23450/ConfigurationManager/v1/objects/storages
   ```
3. Verify whether the storage system is accessible by running the following command:
   ```
   curl -v -H "Accept:application/json" -H "Content-Type:application/json" -u username:password -
   X GET http://172.xx.xx.xx:23450/ConfigurationManager/v1/objects/storages
   ```

   The following shows the output:

   ```
   <
   {
     "data" : [ {
       "storageDeviceId" : "9380007xxxxx",
       "model" : "VSP E1090",
       "serialNumber" : 7xxxxxx,
       "ctl1Ip" : "172.23.xx.xx",
       "ctl2Ip" : "172.23.xx.xx",
       "targetCtl" : "CTL1"
     } ]
   * Connection #0 to host 172.xx.xx.xx left intact
   ```

4. Upgrade Ansible to the latest version:
   The Ansible Core package (ansible-core), which facilitates Red Hat-provided automation content, is included in RHEL v8.6 and later. You must upgrade to the latest ansible-core version for the ansible server.

The following screenshot shows the VM ansible-core version (2.12.2):

```
[root@ansible_controller_demo ~]# ansible --version
ansible [core 2.12.2]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3.8/site-packages/ansible
  ansible collection location = /root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.8.12 (default, Sep 16 2021, 10:46:05) [GCC 8.5.0 20210514 (Red Hat 8.5.0-3)]
  jinja version = 2.10.3
  libyaml = True
[root@ansible_controller_demo ~]#
```

a. Register subscription manager to the RHEL 8.6 virtual machine.

```
[root@ansible_controller_demo ~]#  subscription-manager register
Registering to: subscription.rhsm.redhat.com:443/subscription
Username: HDS_ILAB_KOL
Password:
The system has been registered with ID: 73747d33-0829-414b-91d7-1370e74b9a46
The registered system name is: ansible_controller_demo
[root@ansible_controller_demo ~]#
```

b. Check the subscription manager list and attach a pool ID.

c. Enable the Ansible repository by running the following command:

```
subscription-manager repos –enable ansible*
```

d. Upgrade to the latest version of Ansible core by running the following command:

```
yum install ansible*
```

Check the upgraded version (2.14.5).

```
[root@ansible_controller_demo ~]# ansible --version
ansible [core 2.14.5]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/root/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3.9/site-packages/ansible
  ansible collection location = /root/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.9.13 (main, Nov  9 2022, 13:16:24) [GCC 8.5.0 20210514 (Red Hat 8.5.0-15)] (/usr/bin/python3.9)
  jinja version = 3.1.2
  libyaml = True
[root@ansible_controller_demo ~]#
```

5. Configure the Ansible node to work with the Configuration Manager REST API Server and the EC2 instance test server.

a. Add the host information (Configuration Manager REST API Server and EC2 instance test server) in the `/etc/ansible/hosts` file:

```
172.[redacted]ansible_ssh_user=root ansible_ssh_pass=[redacted]

[target]
10.7[redacted] ansible_user=ec2-user ansible_ssh_private_key_file=/root/.ssh/sayanpem.pem
```

b. Place the private key of the EC2 instance in any location and list the path in the `/etc/ansible/hosts` file of the ansible node.

6. Configure modules, module utilities, parameter files, and playbook tasks.

a. Place the Ansible module files in the standard module placement directory or any other directory.

- Standard module placement directory: Obtain the Ansible version to access this path.
- Any directory: When placing a module in any directory, you must add the directory path to the `/etc/ansible/ansible.cfg` file.

b. Install any additional modules:

- To install ansible.posix (for mount operations):

```
ansible-galaxy collection install ansible.posix
```

- To install community.general (for partition operations):
  `ansible-galaxy collection install community.general`

c. When placing the module utility and logs in any directory, add the directory path to the `/etc/ansible/ansible.cfg` file.

```
[defaults]
module_utils = /ansible/src/module_utils
log_path=/ansible/playbook/logs/ansible.log
```

In this scenario, we used the following two module utility python files:

- **hitachi_block_constant**: This consists of API, Endpoints, http, ModuleArgs, State, ErrorMessages, AutomationConstants, LogMessages, and Log class.
- **hitachi_block_client** : A Python module that defines various classes and functions used in automating tasks on a Hitachi storage system. The module imports various other modules such as datetime, functools, json, logging, time, urllib, HTTPStatus, and custom modules such as hitachi_block_constant.

d. Place the playbooks and parameter files in any directory.

The following lists the common parameters for all tasks:

Parameters	Description
cmrest_server	(Required) Specifies the IP address and hostname of the Configuration Manager REST server for storage operations. Use localhost by default.
cmrest_server_port	(Required) Specifies the communication port number of the Configuration Manager REST server. Use the default of 23450.
storage_device_id	(Required) Specifies the storage device ID of the storage system on where the provisioning task will run.
storage_user	(Required) Specifies the account name for storage operations.
storage_pass	(Required) Specifies the password for the storage operations account.

Example of specifications:

```
cmrest_server: 172.[redacted]
cmrest_server_port: 23450
storage_device_id: "938000[redacted]"
storage_user: opscenter
storage_pass: hitachi
```

7. Configure the iSCSI initiator on the RHEL 9.0 EC2 instance.
   a. Download and install the iSCSI initiator package.
   b. Set the IP address in the network interface and verify that the iSCSI storage ports are reachable through network connectivity.
   c. Use the default iface for the iSCSI connection.
   d. Note the iSCSI initiator name so you can include it in the Ansible parameter file or add in the storage system iSCSI target.

Test 2: Create and Expand the Pool

To create and expand pools, complete the following steps:

1. Specify the LDEV.
2. To expand the pool capacity, add LDEVs to the pool.
   Tip: You can specify LDEVs by LDEV numbers or a range of consecutive LDEV numbers.
3. Create a pool by entering the following request line:
   `ansible-playbook create-Pool.yml`

Input Parameters:

Parameters	Description
ldev_ids	(Required) LDEV number (decimal number).
pool_id	(Required) Pool number (decimal number).
pool_name	(Required) Pool name, which is a character string consisting of 1 to 32 characters.
Pool_type	(Required) Pool type. The specifiable values for Pool type are as follows: • HDT: HDT pools • HDP: HDP pools

Example of specifications:

```
ldev_ids:
  - 65031
pool_id: 1
pool_type: HDP
pool_name: ansiblePool
```

4. Expand the pool by entering the following request line:

```
ansible-playbook expand-Pool.yml
```

Input Parameters:

Parameters	Description
ldev_ids	(Required) LDEV numbers to add to a Pool. This specifies a decimal number that is passed as an object ID in the command line to the Configuration Manager REST API server.
pool_id	(Required) Pool number (decimal number).

Example of specifications:

```
ldev_ids:
  - 65030
pool_id: 1
```

Output Parameters:
Output the Pool information to the outputs key.

For details on the information of output, see the *Configuration Manager REST API Reference Guide*.

Test 3: Create an iSCSI Target and Add the Host

To create an iSCSI target and iSCSI name, specify the host mode and the host mode option when creating the iSCSI target.

Create the iSCSI Target and add the host by entering the following request line:

```
ansible-playbook iscsi-createtarget_addhost.yml
```

Input Parameters:

Parameters	Description
port_settings [Append]	Specifies the storage port information for which the LU path is created. • (Required) port_id: Specifies the port ID number of the storage port on which you want to create the iSCSI target. • (Optional) port_ip: Specifies the IP address of the storage port specified in the port_id. • (Required) host_group_name: Specifies the name of the iSCSI Target to create on the storage port. • (Optional) host_group_number: Specifies the number of the iSCSI target to create on the storage port. • (Required) iscsi_name: Specifies the iSCSI initiator name to associate with the iSCSI target. • (Optional) host_mode: Specifiable values are as follows: HP-UX, SOLARIS, AIX, WIN, LINUX/IRIX, TRU64, OVMS, NETWARE, VMWARE, VMWARE_EX, WIN_EX If set to none, 'LINUX/IRIX' is set by default.

Example of specifications:

```
port_settings:
  - port_id: CL2-D
    port_ip: 172.2[redacted]
    host_group_number: 253
    host_group_name: 2D-Ansible-0253
    iscsi_name: iqn.1994-05.com.redhat:ffe99b6c97f0t1
    host_mode: LINUX/IRIX
```

Output Parameters:
Output the iSCSI Target information to the outputs key.

For details on the information to output, see the *Configuration Manager REST API Reference Guide*.

Test 4: Provision the Volume

To create Dynamic Provisionng volumes and add the volumes to an iSCSI target, enter the following request line:
`ansible-playbook vol_provisioning.yml`

Input Parameters:

Parameters	Description
ldev_id	(Required) LDEV number (decimal number). You can assign or add one LDEV. For multiple LDEV assignments, you can use the ldev_ids parameter.
pool_id	(Required) Specifies the pool number (decimal number) from which the DP pool is created.
Block_capacity	(Required) Capacity of the volume to create in blocks (1 block = 512 bytes).
port_settings [Append]	Specifies the information of the storage port for which the LU path is to be created. • (Required) port_id: Specifies the port ID number of the storage port on which you want to add the LUN. • (Optional) port_ip: Specifies the IP address of the storage port specified in the port_id. • (Required) host_group_name: Specifies the name of the iSCSI Target in which the LUNs are added.

Example of specifications:

```
pool_id: 0
block_capacity: 10000000
ldev_id: 9712
port_settings:
  - port_id: CL2-D
    port_ip: 172.2[redacted]
    host_group_number: 253
```

Output Parameters:
Output the LDEV information result to the outputs key.

For details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

Test 5: Expand the DP Volume

To expand the capacity of a DP volume, enter the following request line:

```
ansible-playbook expand-vol.yml
```

Input Parameters:

Parameters	Description
ldev_id	(Required) LDEV number, which specifies the number of LDEVs to expand.
Block_capacity	(Required) Capacity of the volume to add in blocks (1 block = 512 bytes).

Example of specifications:

```
block_capacity: 10000000
ldev_id: 9712
```

Output Parameters:
Output the LDEV information result to the outputs key.

For details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

Test 6: Add a CHAP User

To create a CHAP user with secret for the iSCSI target, enter the following request line:

```
ansible-playbook create-CHAPuser.yml
```

In this scenario, we created a CHAP user for one way CHAP.

Input Parameters:

Parameters	Description

chap_settings: (Append)	Specifies the information of the CHAP user to create.
	• (Required) chap_user_name: Specifies the CHAP username (1 to 223 characters).
	• (Required) port_id: Port number on which to add the chap user.
	• (Required) hosthost_group_number: Target ID of the iSCSI target.
	• (Required) way_of_chap_user: Type of CHAP username:
	  ○ TAR: The CHAP username of the iSCSI target side.
	  ○ INI: The CHAP username of the host bus adapter (iSCSI initiator) side.
	• (Required) chap_password: Specifies a secret consisting of 12 to 32 characters for the specified CHAP user.

Example of specifications:

```
chap_settings:
  - chap_user_name: "user12"
    port_id: CL2-D
    hosthost_group_number: 253
    way_of_chap_user: "INI"
    chap_password: "user12_password"
```

Output Parameters:
Output the iSCSI Target information to the outputs key.

For details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

Test 7: Add a New LUN

For creating a new DP-Vol and adding it to the iSCSI target, enter the following request line:
`ansible-playbook vol_provisioning.yml`

Note: This task is similar to Test 4: Provision the Volume. Only ldev_id and block_capacity parametes must be changed according to the requirement.

Output Parameters:
Output the information of the newly created LUN to the outputs key.

For more details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

Test 8: Add a Host Nickname

To set or change the nickname for iSCSI initiators registered in an existing iSCSI target, enter the following request line:
`ansible-playbook add_hostname.yml`

Input Parameters:

Parameters	Description
port_settings [Append]	Specifies the information of the storage port for which the LU path is to be created. • (Required) port_id: Specifies the port ID number of the storage port on which you want to set the nickname. • (Required) host_group_number: Specifies the number of the iSCSI target on which you want to set the nickname. • (Required) iscsi_name: iSCSI name of the host bus adapter (iSCSI initiator) registered for the iSCSI target. • (Required) nick_name: Specifies the nickname consisting of 1 to 32 characters for the iSCSI name.

Example of specifications:

```
port_settings:
  - port_id: CL2-D
    host_group_number: 253
    iscsi_name: iqn.1994-05.com.redhat:ffe99b6c97f0t1
    nick_name: HOST2
```

Output Parameters:
Output the information of the iSCSI Target to the outputs key.

For more details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

Test 9: Change the CHAP Authentication Method

To change the CHAP authentication method (set iSCSI target direction and target side CHAP user), enter the following request line:

`ansible-playbook mutual-authentication.yml`

Input Parameters:

Parameters	Description
port_settings [Append]	Specifies the information of the storage port for which the LU path is to be created. • (Required) port_id: Specifies the port ID number of the storage port on which you want to set the nick name. • (Required) host_group_number: Specifies the number of the iSCSI target on which you want to set the nick name. • (Required) host_mode: Host mode of the specified iSCSI target. • (Required) iscsi_target_direction: Direction of CHAP authentication for the iSCSI target. • The specifiable types are as follows: o S: Unidirectional authentication mode (The iSCSI target authenticates the iSCSI initiator.) o D: Bidirectional authentication mode (The iSCSI target and the iSCSI initiator authenticate each other.)
chap_settings (Append)	Specifies the information of the CHAP user to be created. • (Required) chap_user_name: Specifies the CHAP username consisting of 1 to 223 characters. • (Required) port_id: Port number on which the chap user is to be added. • (Required) hosthost_group_number: Target ID of the iSCSI target. • (Required) way_of_chap_user: Type of the CHAP username: o TAR: The CHAP username of the iSCSI target side o INI: The CHAP username of the host bus adapter (iSCSI initiator) side. • (Required) chap_password: Specifies a secret consisting of 12 to 32 characters for the specified CHAP user.

Example of specifications:

```
port_settings:
  - port_id: CL2-D
    host_group_number: 253
    host_mode: LINUX/IRIX
    iscsi_target_direction: D
chap_settings:
  - chap_user_name: user21
    port_id: CL2-D
    hosthost_group_number: 253
    way_of_chap_user: TAR
    chap_password: user21_password
```

Output Parameters:
Output the information of the iSCSI Targets to the outputs key.

For more details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

Test 10: Create an ADR Volume and Toggle Between ADR and Non-ADR Setting

1. To create an ADR volume, enter the following request line:
   `ansible-playbook create-ADRldev.yml`

   Input Parameters:

Parameters	Description
ldev_id	(Required) LDEV number which specifies a decimal number. You can assign or add one LDEV. For multiple LDEV assignments, use the ldev_ids parameter.

pool_id	(Required) Specifies the pool number with a decimal number from which the DP pool is to be created.
Block_capacity	(Required) Capacity of the volume to be created in blocks (1 block = 512 bytes).
Data_reduction_mode	(Required) The specifiable values are as follows: • Compression: Enable the capacity saving function (compression). • Compression_deduplication: Enable the capacity saving function (compression and deduplication). • disabled: Disable the capacity saving function (compression and deduplication). You can use the compression accelerator for specifying compression or compression_deduplication and the storage system. The capacity saving function, that uses a compression accelerator, will be automatically enabled.

Example of specifications:

```
pool_id: 1
block_capacity: 20971520
ldev_id: 9705
data_reduction_mode: 'compression_deduplication'
```

Output Parameters:
Output the information of the ADR Volume to the outputs key.

For more details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

2. To toggle between ADR and Non-ADR, enter the following request line:
   `ansible-playbook toggle-adr-nonadr.yml`

   Input Parameters:

Parameters	Description
ldev_id	(Required) LDEV number that specifies the LDEV for which you can change the properties.
Data_reduction_mode	(Required) The specifiable values are as follows: • compression: Enable the capacity saving function (compression). • Compression_deduplication: Enable the capacity saving function (compression and deduplication). • disabled: Disable the capacity saving function (compression and deduplication).

   Example of specifications:

```
ldev_id: 9705
data_reduction_mode: 'disabled'
```

   Note: By changing the data_reduction_mode parameter, you can toggle any volume between ADR and non-ADR.

   Output Parameters:
   Output the information of the Volume to the outputs key.

   For more details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

Test 11: SI Pair Operations

To create SI pair mapping host groups, split SI pair, mount volume on the host, resync the SI pair, and then restore the SI pair, complete the following steps:

1. To create an SI pair and map the volumes to host groups, enter the following request line:

   `ansible-playbook create-SI.yml`

Note: This task creates P-VOLs, S-VOLs, host groups, map them to the host group, and create an SI pair**.** You can create a SI pair only if P-VOLs and S-VOLs are already created and assigned.

Input Parameters:

Parameters	Description
port_settings [Append]	Specifies the information of the storage port for which the LU path is to be created. • (Required) port_id: Specifies the port ID number of the storage port on which you want to create the iSCSI target. • (Optional) port_ip: Specifies the IP address of the storage port specified in the port_id. • (Required) host_group_name: Specifies the name of the iSCSI Target to be created on the storage port. • (Optional) host_group_number: Specifies the number of iSCSI targets to be created on the storage port. • (Required) iscsi_name: Specifies the iSCSI initiator name to associate with the iSCSI target. • (Optional) host_mode: The specifiable values are as follows: HP-UX, SOLARIS, AIX, WIN, LINUX/IRIX, TRU64, OVMS, NETWARE, VMWARE, VMWARE_EX, WIN_EX If set to none, 'LINUX/IRIX' will be set by default.
pool_id	(Required) Pool ID from which the P-VOL must be created.
block_capacity	(Required) Capacity of the P-VOL.
ldev_id	(Required) LUN ID of the P-VOL.
SI (Append)	Specifies the information of all the parameters required to create an SI PAIR: • (Required) pool_id: Pool ID from which the S-VOL must be created. • (Required) block_capacity: Capacity of the S-VOL. • (Required) ldev_id: LUN ID of the S-VOL. • (Required) copy_group_name: Specifies the copy group name. The value must not exceed 29 characters, and the name is case sensitive. • (Required) copy_pair_name: Specifies the copy pair name. The value must not exceed 31 characters, and the name is case sensitive. • (Required) copy_pace: Specifies a value in the range of 1 to 15 for copy speed. • (Required) consistency_group_id: To register the new pair in the consistency group, specify the consistency group ID (0 to 127). • (Required) port_settings (Append): o (Required) port_id: Specifies the port ID number of the storage port on which you want to create the iSCSI target. o (Optional) port_ip: Specifies the IP address of the storage port specified in the port_id. o (Required) host_group_name: Specifies the name of the iSCSI Target to be created on the storage port. o (optional) host_group_number: Specifies the number of iSCSI targets to be created on the storage port. o (Required) iscsi_name: Specifies the iSCSI initiator name to associate with the iSCSI target. o (optional) host_mode: The specifiable values are as follows: HP-UX, SOLARIS, AIX, WIN, LINUX/IRIX, TRU64, OVMS, NETWARE, VMWARE, VMWARE_EX, WIN_EX If set to none, 'LINUX/IRIX' will be set by default.

Example of specifications:

```
pool_id: 0
block_capacity: 10000000
ldev_id: 9700
port_settings:
  - port_id: CL2-D
    port_ip: 172.[redacted]
    host_group_number: 253
    host_group_name: 2D-Ansible-0253
    iscsi_name: iqn.1994-05.com.redhat:ffe99b6c97f0t2
    host_mode: LINUX/IRIX
SI:
  pool_id: 0
  block_capacity: 10000000
  ldev_id: 9701
  port_settings:
    port_id: CL1-D
    port_ip: 172.[redacted]
    host_group_number: 253
    host_group_name: 1D-HAD-0253
    iscsi_name: iqn.1994-05.com.redhat:ffe99b6c97f0t2
    host_mode: LINUX/IRIX
  copy_group_name: HAD-SI-9700-9701
  copy_pair_name: clone_0025E4_0025E5
  copy_pace: 4
  consistency_group_id: 50
```

Output Parameters:
Output the information of the iSCSI Targets and SI PAIR to the outputs key.

For more details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

2. To split the SI PAIR, enter the following request line:
   `ansible-playbook splitSI.yml`

   Input Parameters:

Parameters	Description
SI (Append)	Specifies the information of all the parameters required to create an SI PAIR: • (Required) copy_group_name: Specifies the copy group name. The value must not exceed 29 characters, and the name is case sensitive. • (Required) copy_pair_name: Specifies the copy pair name. The value must not exceed 31 characters, and the name is case sensitive. • (Required) copy_pace: Specifies a value in the range of 1 to 15 for the copy speed.

   Example of specifications:

```
SI:
  copy_group_name: HAD-SI-9700-9701
  copy_pair_name: clone_0025E4_0025E5
  copy_pace: 4
```

   Output Parameters:
   Output the information of the SI PAIR to the outputs key.

   For more details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

3. To mount the S-VOL to the EC2 host, enter the following request line:
   `ansible-playbook mount-volumeec2-part-filesystem-fstabentry.yml`

Note: This task creates a partition, filesystem, fstab entry, and then mounts the volume to the mount point. To only mount the volume to the mount point, enter the following request line:

```
ansible-playbook mount-volumeec2-mountpoint.yml
```

Note: Parameters are the same for both mount tasks.

Input Parameters:

Parameters	Description
filesystems (Append)	Specifies the information of the parameters required to mount the S-VOL. • (Required) fs_type: File system type which will be created. • (Required) dev_path: Device path on which the task will be run. • (Required) part_num: Partition number of the device. • (Required) mnt_path: Mount path where the device will be mounted.

Example of specifications:

```
---
filesystems:
  - fs_type: ext4
    dev_path: /dev/sdb
    part_num: "1"
    mnt_path: /mnt/dir05
```

Output Parameters:
Output the information of the mount operation to the outputs key.

4. To resync the SI pair, enter the following request line:
   ```
   ansible-playbook resync-SI.yml
   ```

5. To restore the SI pair, enter the following request line:
   ```
   ansible-playbook restoreSI.yml
   ```

   Note: The same parameters are used for both resync and restore operations for splitting the SI pair.

Test 12: TI Pair Operations

To create a TI pair with S-VOL, split the TI pair, map the secondary volume to the host group, and mount the S-VOL to EC2 instance, complete the following steps:

1. To create a Thin Image PAIR, enter the following request line:

   ```
   ansible-playbook createTI-snapshot-svol.yml
   ```

   Note: This task creates a TI pair between the P-VOL and S-VOL assuming that the volumes are already created, and P-VOL is assigned to the host group.

Input Parameters:

Parameters	Description
TI (Append)	Specifies the information of all the parameters required to create a TI PAIR:

	• (Required) snapshot_group_name: Specifies the name of the snapshot group for which the Thin Image pair is to be created. • (Required) snapshot_pool_id: Specifies the pool ID in which the snapshot data is to be created. • (Required) pvol_ldev_id: LUN ID of the P- VOL. • (Required) svol_ldev_id: LUN ID of the S- VOL. • (Optional) is_consistency_group: The snapshot group for which the Thin Image pair is to be created; specifies whether the snapshot group must be created in the consistency group mode (CTG mode). o true: Creates the snapshot group in the CTG mode. o false: Does not create the snapshot group in the CTG mode. If you do not specify this item, false is set automatically. • (Optional) generations: Number of snapshot generation.

Example of specifications:

```
TI:
  snapshot_group_name: HAD_TI_9711
  snapshot_pool_id: 1
  pvol_ldev_id: 9709
  svol_ldev_id: 9711
  is_consistency_group: false
  generations: 1
```

Output Parameters:
Output the information of the TI PAIR to the outputs key.

For more details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

2. To split the Thin Image PAIR, enter the following request line:
`ansible-playbook split-TI.yml`

Input Parameters:

Parameters	Description
TI (Append)	Specifies the information of all the parameters required to create a TI PAIR: • (Required) snapshot_group_name: Specifies the name of the snapshot group for which the Thin Image pair is to be created. • (Required) pvol_ldev_id: LUN ID of the P-VOL.

Example of specifications:

```
snapshot_group_name: HAD_TI_9711
pvol_ldev_id: 9709
```

Output Parameters:
Output the information of the TI PAIR to the outputs key.

For more details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

3. To map the secondary volume to the host group, enter the following request line:
`ansible-playbook addlunpath_secondary.yml`

Input Parameters:

Parameters	Description
port_settings (Append)	Specifies the information of all the parameters required to create a TI PAIR: • (Optional) port_id: Port ID number in which the LUN is to be mapped. • (Required) host_group_number: Host group number in which the LUN is to be mapped.
Ldev_id	(Required) Number of the S-VOL Ldev ID to be mapped to the host group.

Example of specifications:

```
ldev_id: 9711
port_settings:
  - port_id: CL2-D
    host_group_number: 254
```

Output Parameters:
Output the information of the LUN path to the outputs key.

For more details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

4. To mount the S-VOL to the EC2 host, enter the following request line:
   `ansible-playbook mount-volumeec2-part-filesystem-fstabentry.yml`

   Note: This task creates a partition, filesystem, fstab entry, and then mounts the volume to the mount point.

   To only mount the volume to the mount point, enter the following request line:
   `ansible-playbook mount-volumeec2-mountpoint.yml`

   Note: Parameters are the same for both mount tasks.

   Input Parameters:

Parameters	Description
filesystems (Append)	Specifies the information of the parameters required to mount the S-VOL. • (Required) fs_type: File system type which will be created. • (Required) dev_path: Device path on which the task will be run. • (Required) part_num: Partition number of the device. • (Required) mnt_path: Mount path where the device will be mounted.

   Example of specifications:

```
---
filesystems:
  - fs_type: ext4
    dev_path: /dev/sdb
    part_num: "1"
    mnt_path: /mnt/dir05
```

   Output Parameters:
   Output the information of the mount operation to the outputs key.

Test 13: Volume-less TI PAIR operations

To create a volume-less TI pair, split the TI pair, map the snapshot data to the S-VOL, map the secondary volume to the host group, and mount the S-VOL to the EC2 instance, complete the following steps:

1. To create a volume-less Thin Image pair, enter the following request line:
   `ansible-playbook createTI-snapshot-cascade-vless.yml`

   Note: This task creates a TI pair with only P-VOL assuming that the volumes are already created, and the P-VOL is assigned to a host group.

   Input Parameters:

Parameters	Description
TI (Append)	Specifies the information of all the parameters required to create a TI PAIR: • (Required) snapshot_group_name: Specifies the name of the snapshot group for which the Thin Image pair is to be created. • (Required) snapshot_pool_id: Specifies the pool ID where the snapshot data is to be created. • (Required) pvol_ldev_id: LUN ID of the P-VOL. • (Optional) is_consistency_group: Snapshot group for which the Thin Image pair is to be created; specifies whether the snapshot group must be created in the consistency group mode (CTG mode). o true: Creates the snapshot group in the CTG mode. o false: Does not create the snapshot group in the CTG mode. If you do not specify this item, false is set automatically. • (Optional) generations: Number of snapshot generation.

   Example of specifications:

```
TI:
  snapshot_group_name: HAD_TI_9711
  snapshot_pool_id: 1
  pvol_ldev_id: 9709
  is_consistency_group: false
  generations: 1
```

   Output Parameters:
   Output the information of the TI PAIR to the outputs key.

   For more details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

2. To split the volume-less Thin Image pair, enter the following request line:
   `ansible-playbook split-TI.yml`

   Note: The parameters used here are the same as the parameters used for splitting a TI pair in Test 12: TI Pair Operations.

3. To map the snapshot to S-VOL, enter the following request line:
   `ansible-playbook assignsnapshot_SVOL.yml`

   Input Parameters:

Parameters	Description

TI (Append)	Specifies the information of all the parameters required to create a TI PAIR: • (Required) svol_ldev_id: LUN ID of the S-VOL. • (Required) pvol_ldev_id: LUN ID of the P-VOL. • (Required) mu_number: MU number of the P-VOL.

Example of specifications:

```
TI:
  svol_ldev_id: 9711
  pvol_ldev_id: 9709
  mu_number: 3
```

Output Parameters:
Output the information of the TI PAIR to the outputs key.

For more details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

4. To map the secondary volume to the host group, enter the following request line:
   `ansible-playbook addlunpath_secondary.yml`

   Note: The parameters used here are the same as the parameters used for mapping secondary volume to the host group in Test 12: TI Pair Operations.

5. To mount the S-VOL to the EC2 host, enter the following request line:
   `ansible-playbook mount-volumeec2-part-filesystem-fstabentry.yml`

   Note: This task creates a partition, filesystem, fstab entry, and then mounts the volume to the mount point.
   To only mount the volume to the mount point, enter the following request line:
   `ansible-playbook mount-volumeec2-mountpoint.yml`

   Note: The parameters used here are the same as the parameters used for mounting the S-VOL to the EC2 instance in Test 12: TI Pair Operations.

Test 14: Perform Resync and Restore operations of the TI Pair

To perform Resync and Restore operations of the TI pair, complete the following steps. The same parameters are used for both.

1. To perform Resync operation of Thin Image pair, enter the following request line:
   `ansible-playbook resync_TIPAIR.yml`

2. To perform Restore operation of Thin Image pair, enter the following request line:
   `ansible-playbook restore_TIpair.yml`

   Input Parameters:

Parameters	Description
snapshot_group_name	(Required) Specifies the name of the snapshot group for which the Thin Image pair is to be created.
pvol_ldev_id	(Required) LUN ID of the P-VOL.
mu_number	(Required) MU number of the P-VOL.

   Example of specifications:

```
snapshot_group_name: HAD_TI_9711
pvol_ldev_id: 9709
mu_number: 3
```

Output Parameters:
Output the information of the TI PAIR to the outputs key.

For details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

Test 15: Revert All TI Operations

To unmount the S-VOL from the EC2 instance, delete the S-VOL LUN path from the iSCSI target, unmap the snapshot data, and delete the TI PAIR, complete the following steps:

1. To unmount the S-VOL from the EC2 instance, enter the following request line:
   `ansible-playbook unmount-volumeec2-fromfstab-mountpoint.yml`

   Note: The parameters used here are the same as the parameters used for mounting the S-VOL to the EC2 instance in Test 12: TI Pair Operations.

2. To delete the LUN path of S-VOL from the iSCSI target, enter the following request line:
   `ansible-playbook delete-lunpathTI.yml`

   Input Parameters:

Parameters	Description
lun	(Required) Specifies the LUN number to be removed from the iSCSI target.
port_settings (Append)	• (Required) port_id: Port ID of the specified port. • (Required) host_group_number: Host group number from which the LUN must be removed.

   Example of specifications:

```
lun: 1
port_settings:
  - port_id: CL2-D
    host_group_number: 254
```

   Output Parameters:
   Output the information of the TI PAIR to the outputs key.

   For details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

3. To unmap the snapshot data from the S-VOL, enter the following request line:
   `ansible-playbook unassignsnapshot_SVOL.yml`

4. To delete the TI pair, enter the following request line:
   `ansible-playbook delete_TI.yml`

   Note: The same parameters are used for both unmap and delete operations.

   Input Parameters:

Parameters	Description

TI (Append)	Specifies the information of all parameters required to create a TI PAIR: • (Required) pvol_ldev_id: LUN ID of the P- VOL. • (Required) mu_number: MU number of the P-VOL.

Example of specifications:

```
TI:
  pvol_ldev_id: 9709
  mu_number: 3
```

Output Parameters:
Output the information of the TI PAIR to the outputs key.

For details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

Test 16: Delete the Host

To delete the host iqn from the specified iSCSI target and unmount the S-VOL from the EC2 instance, enter the following request line:

```
ansible-playbook delete-host.yml
```

Input Parameters:

Parameters	Description
port_settings (Append)	• (Required) port_id: Port ID of the specified port. • (Required) host_group_number: Host group number from which the LUN must be removed. • (Required) iscsi_name: iqn name which must be deleted from the iSCSI target.

Example of specifications:

```
port_settings:
  - portId: CL2-D
    host_group_number: 253
    iscsi_name: iqn.1994-05.com.redhat:ffe99b6c97f0t1
```

Output Parameters:
Output the information of the iSCSI Target to the outputs key.

For more details on the information to be output, see the *Configuration Manager REST API Reference Guide*.

Test 17: Delete Volume Operation

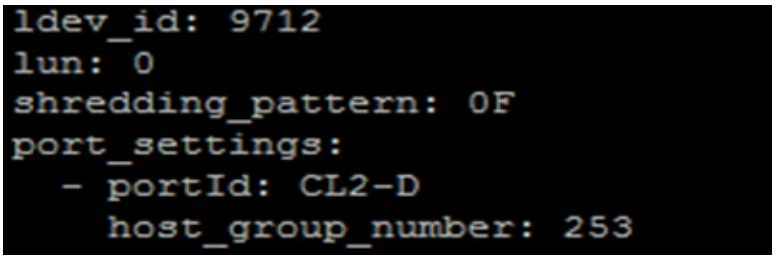
To delete the LUN path, iSCSI target, DP volume, enter the following request line:

```
ansible-playbook delete-volume.yml
```

Input Parameters:

Parameters	Description
port_settings (Append)	• (Required) portID: Port ID of the specified port. • (Required) host_group_number: Host group number from which the LUN must be removed.
ldev_id	(Required) Volume ID to be removed.
Lun	(Required) LUN number to be removed.
Shredding_pattern	(Optional) Specifies a pattern consisting of 1 to 8 characters in a hexadecimal format.

Example of specifications:



Output Parameters:
Output the information of the completion task to the outputs key.

For details on the information to be output, see the *Configuration Manager REST API Reference Guide*.