Hitachi Vantara

Business Continuity for Containerized Applications in a Hybrid Cloud Environment

Using a Hitachi VSP 5200 storage system, Red Hat OpenShift Cluster, and Kasten K10

Hitachi Vantara
September 2023

Table of Contents

Notices and Disclaimer

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.

About This Guide

This reference architecture documents how to set up backup and restoration operations between two OpenShift clusters using Kasten K10 Multi-Cluster Manager and Hitachi Storage Plug-in for Containers. Additionally, the document includes test procedures to validate the resiliency of the solution, which you can leverage for your own proof-of-concept before deploying the 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	September 2023	Hitachi Vantara LLC	Initial Release

References

- Red Hat OpenShift Container Platform Installing on AWS v4.11
- Red Hat OpenShift Container Platform Installing on vSphere v4.11https://access.redhat.com/documentation/en-us/openshift_container_platform/4.11/html/installing/installing-on-vsphere
- Hitachi Storage Plug-in for Containers Quick Reference Guide v3.11.0
- Hitachi Virtual Storage Platform 5000 Series: System Administrator Guide
- Veeam Kasten K10 Guide

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 reference architecture documents the process of cloud-based migration of a containerized application with the Kubernetes volume snapshot function using Hitachi Storage Plugin for Containers (HSPC) and Kasten K10 Multi-Cluster Manager by Veeam when a Hitachi Virtual Storage Platform (VSP) storage system is used as a storage backend. HSPC leverages Thin Image (TI) point-in-time snapshots that are instantaneous and space efficient. Using the MySQL stateful application as an example, this document describes how to use HSPC for backup and restore, disaster recovery, and data mobility. In addition, it includes some real-world use cases. The environment used for this validation includes two Red Hat® OpenShift clusters, one at the near-cloud VMware environment and another in Amazon Web Services (AWS). For both the clusters, storage is provided from a VSP 5200 storage system located at the near-cloud data center. Keeping the application data in a centralized location has a number of benefits including costs, performance, and security. The near-cloud data center is a colocation operated by Equinix.

The Equinix colocation was selected because it offered high-speed and low latency connections to the major hyperscalers, such as AWS. Hitachi Vantara collaborated with Equinix to offer a near-cloud hybrid offering called **Hitachi Cloud Connect for Equinix**.

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

Red Hat OpenShift is a hybrid-cloud application platform that leverages the power of Kubernetes and combines reliable and proven services to make the process of developing, modernizing, deploying, running, and managing applications more streamlined. OpenShift ensures a uniform user experience, whether applications are deployed on public-cloud, on-premises, hybrid-cloud, or edge architecture.

The installation program of OpenShift Container Platform offers flexibility to deploy on a wide range of platforms. You can deploy OpenShift Container Platform on bare metal, AWS, Azure, GCP, VMware vSphere, and so on.

You can install OpenShift Container Platform using either installer-provisioned (IPI) or user-provisioned infrastructure (UPI) methods. In this solution, Red Hat OpenShift cluster version 4.11.25 was deployed using the installer-provisioned method.

The Hitachi Storage Plug-in for Containers is a software component comprising of libraries, settings, and commands that enable you to create a container for running stateful applications. The software enables stateful applications to persist and maintain data after the lifecycle of the container has ended. HSPC provides persistent volumes (PV) from Hitachi storage systems.

Kasten K10 is an enterprise grade robust data management platform by Veeam that helps organizations to back up and restore container-based applications on Kubernetes/OpenShift. The capabilities include automating and orchestrating data backup, recovery, disaster recovery, and application mobility across multiple Kubernetes clusters and cloud environments. Kasten K10 offers support for a variety of Kubernetes distributions, as well as public and private cloud providers and storage solutions.

The environment used for this validation includes a Red Hat OpenShift cluster, at the near-cloud data center, and a Red Hat OpenShift cluster in AWS. Both clusters share the same VSP 5200 storage system located in the near-cloud data center for persistent volume requirement for stateful applications. Keeping the data at the near-cloud location ensures data availability to any cloud vendor at close proximity and avoids cloud locking. The near-cloud data center is a colocation operated by Equinix.

To summarize, our hybrid cloud environment consists of the following two domains. The relationship between the two sites is shown in *Figure 1*.

- A near-cloud Equinix colocation data center (named SV5), located in San Jose, California.
- A cloud hosted by AWS in Northern California.

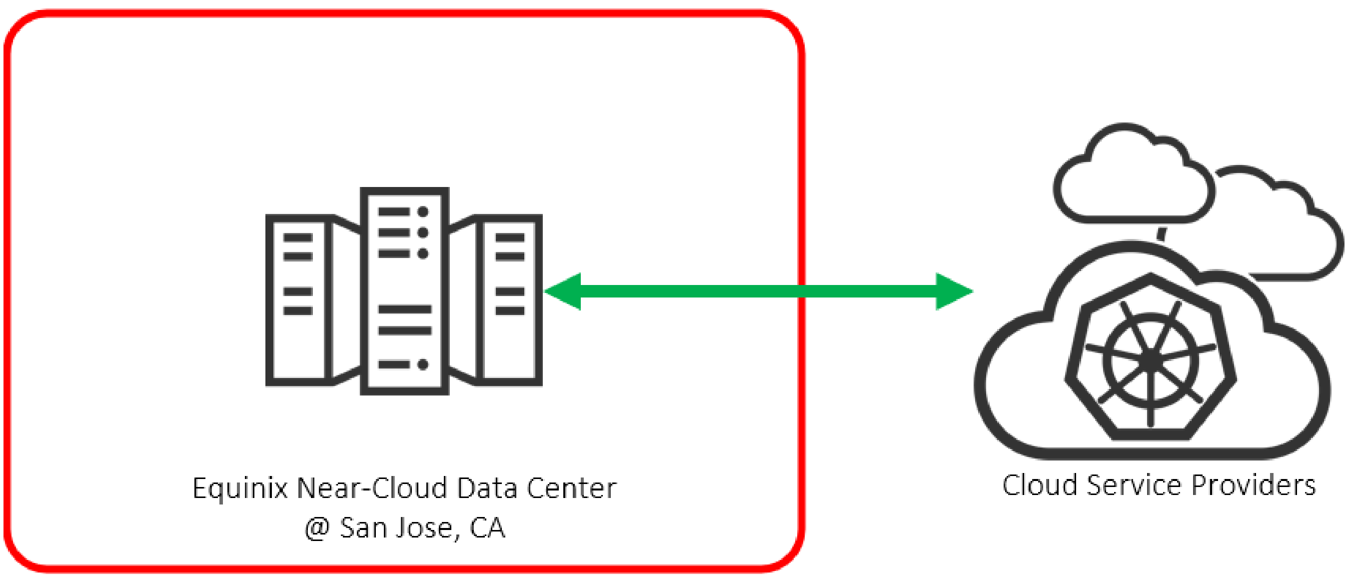


Figure 1: *Hybrid Cloud Environment*

> **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.

Solution Overview

HSPC integrates the OpenShift Container Platform with the Hitachi storage system by using the Container Storage Interface (CSI). Integrating backup software Kasten K10 with HSPC enables protection from data loss and on demand application mobility in the OpenShift Container Platform by using the Hitachi storage system functions (such as Thin Image snapshots and ShadowImage clones).

In addition, implementing HSPC enables the availability of high-performance and high-reliability persistent volumes.

Benefits

The following lists the benefits of business continuity solution using Red Hat OpenShift Cluster, Hitachi storage system, and Kasten K10 Multi-Cluster:

- The solution allows business to resume operations quickly when a disaster brings down a cluster environment.
- On-demand application mobility: Provides the flexibility to quickly snap data copies in multiple environments for on-demand analytics, data mining, DR testing, development testing, and similar use cases.
- The backup and restore operation of Kubernetes clusters in a hybrid cloud environment can be centralized with a single pane of glass UI provided by Kasten K10 Multi-Cluster manager.
- Recover against ransomware attacks: Granular, schedule-based snapshots with immutability (using Data Retention Utility) enables the administrator to recover from a point-in-time snapshot before the attack.
- The substantial reduction in cloud egress costs can be achieved by sharing the same near-cloud storage between AWS and the near-cloud cluster.

Key Components

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

- Red Hat OpenShift Container Platform: This solution involved two Red Hat OpenShift Clusters. The first cluster consisted of three Control Plane nodes and two Worker nodes that were configured in the VMware environment at the near-cloud data center. The second cluster consisted of three Control Plane nodes and two Worker nodes; however, this was configured in AWS. Some of the key components of Red Hat OpenShift Container Platform are:
  - OpenShift Control Plane node: Runs services required for controlling the OpenShift Container Platform cluster and manages node workloads.
  - OpenShift Worker node: Responsible for running containerized workloads, managing resources, and communicating with the control plane to ensure that the desired state of the cluster is maintained.
  - Namespace: Provides a way to organize and isolate resources within a cluster, making it easier to manage and secure workloads
  - Persistent Volume and Persistent Volume Claim (PVC): A part of the storage of the cluster that is statically provided by the cluster administrator or dynamically provided by using the "StorageClass" object.
- HSPC: A CSI plugin from Hitachi used to provision persistent volume from the Hitachi storage system to Red Hat OpenShift or Kubernetes cluster to preserve and maintain data after the container life cycle ends.
  - CSI-controller: Mainly incorporates the CSI controller service for storage operation. This service is deployed as "Deployment" and is run only on the control plane.
  - CSI-node: Mainly incorporates the CSI node service that manages volumes in each node. This service is deployed as "DaemonSet". This component is required for all nodes.
- Veeam Kasten K10 Multi-Cluster Manager: Kasten K10 provides a user-friendly data management platform to perform backup or restore, disaster recovery, and mobility of containerized applications. The K10 Multi-Cluster manager provides a platform for K10 operations across multiple OpenShift clusters in a hybrid-cloud environment.
- VSP Storage System: A VSP 5200 storage system was used for persistent volume in Red Hat OpenShift clusters deployed in near-cloud and AWS for stateful application.
- 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. Some of the key services used in AWS cloud are EC2, S3, Route53, Classic load balancer, and Network load balancer.

Validation

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

Validation Method

This solution consists of the following test cases.

Test case 1 involves setting up the environment, which includes two Red Hat OpenShift Container Platform clusters - one in near-cloud and the other in AWS.

To validate test case 2, a persistent volume was allocated from the VSP 5200 storage system located in near-cloud to deploy a stateful MySQL application in both Red Hat OpenShift Container Platforms in near-cloud and AWS.

To validate test case 3, fresh data was inserted into the MySQL application, and after restoring the backup, the database records were verified at the AWS location to ensure the data consistency. The Kasten K10 Multi-Cluster user interface was used to perform this use case. A Global Location Profile was created with AWS S3 bucket as the storage provider, followed by creating Global Policies to automate the workflows for managing data (such as snapshot and restore). The subsequent step was to add Distributions, which defines the clusters where K10 resources must be allocated. Finally, snapshot and restore operations were carried out using the Global Policies.

Instead of Kasten K10 Multi-Cluster, a manual approach with Kubernetes commands was used to validate test case 4. Before performing the backup operation, fresh data was inserted into the MySQL application. A snapshot of the persistent volume was created with Kubernetes Volume Snapshot function with HSPC. In the target cluster, a PVC of the snapshot volume was created and used as a source to create a clone volume. The stateful MySQL application was restored using this cloned PVC in the target cluster.

Test case 5 shows how business continuity can be performed if a ransomware attack corrupts the application data. To validate this test case, a stateful MySQL application was used and the Data Retention Utility (DRU) feature was set on the snapshot volume to restrict read and write. If a ransomware attack corrupts the application's data, the data can be restored from the snapshot. You can perform the recovery process in either of the Red Hat OpenShift Container Platform clusters (near-cloud or AWS). The process involves creating a PVC of the DRU-enabled snapshot, creating a snap-on-snap copy of that PVC, and then restoring the stateful MySQL application using the cloned PVC in the target cluster.

High Level Diagrams

Figure 2 shows the test environment used to run the validation.

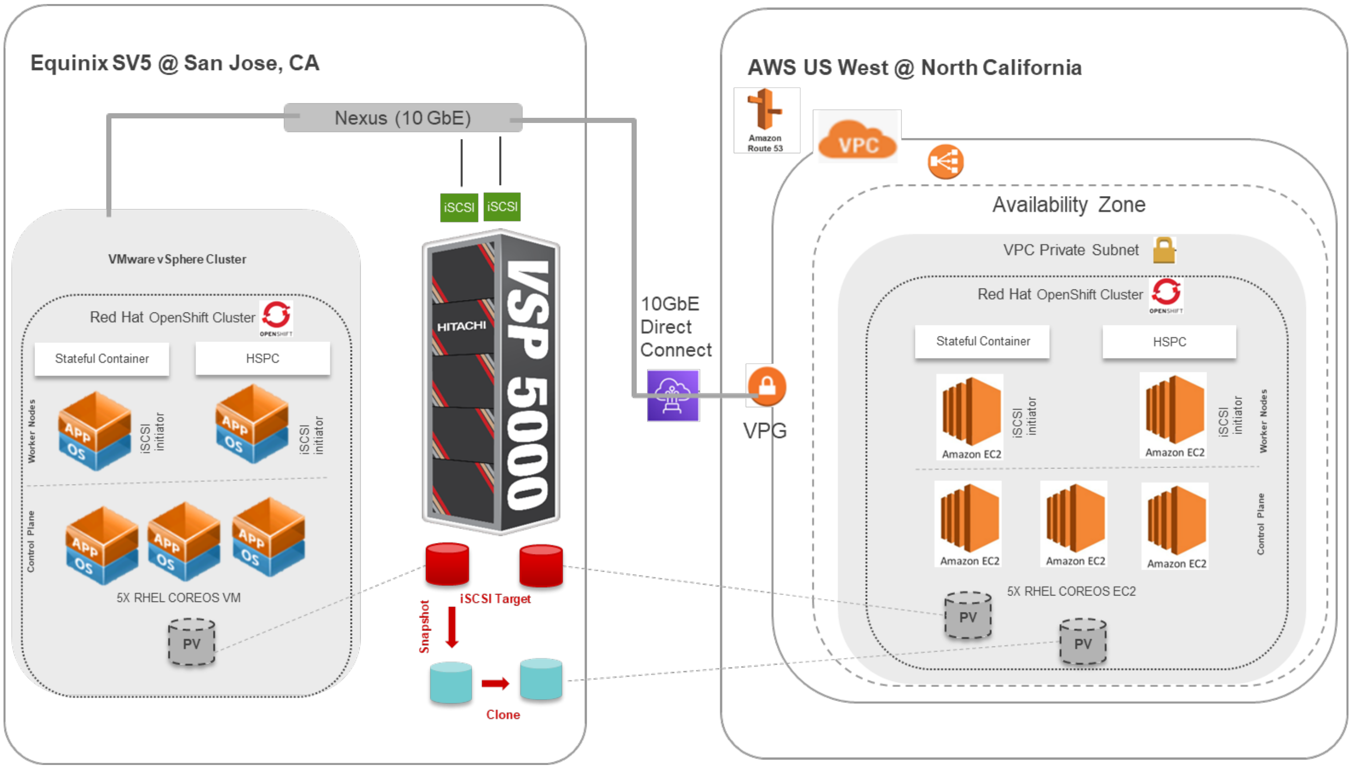


Figure 2: *Test Environment*

Figure 3 shows the detail view of backup and restore using Kasten K10 Multi-Cluster.

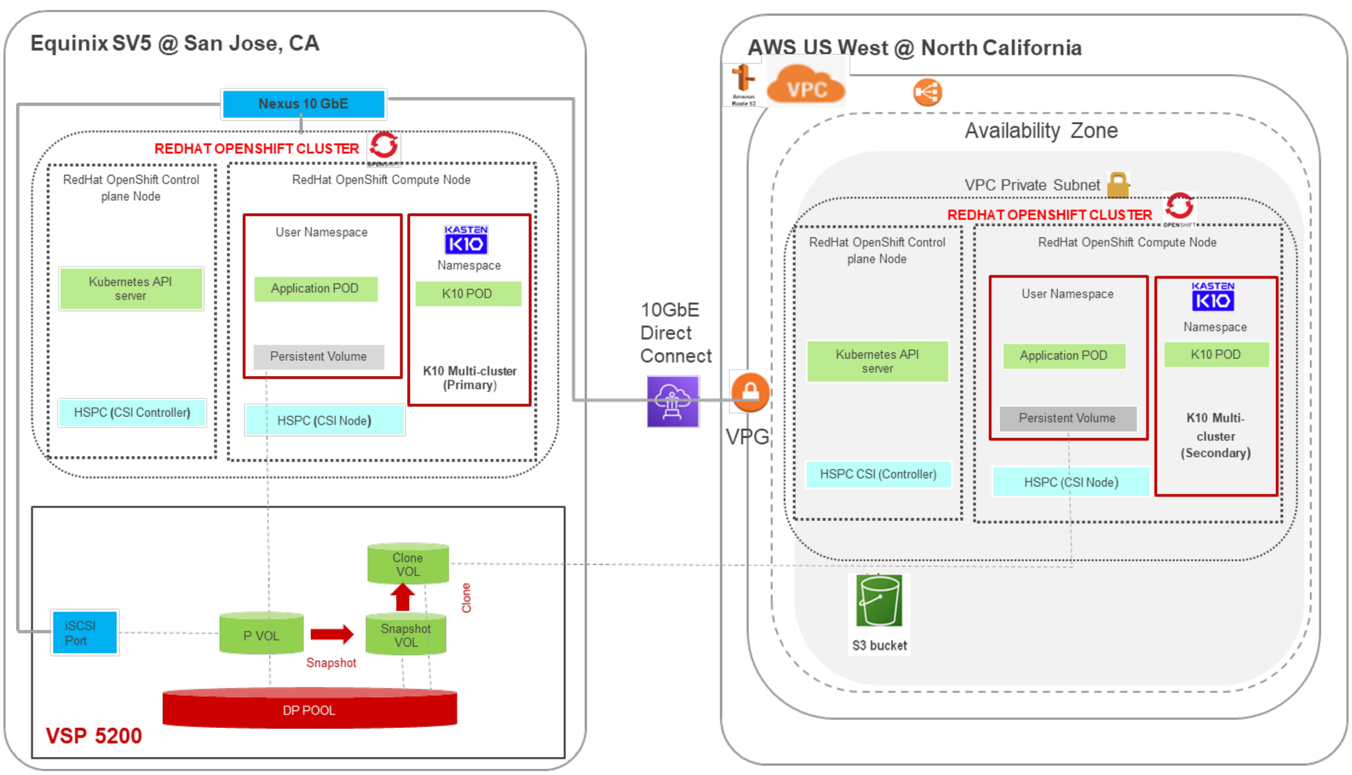


Figure 3: *Test Environment for Kasten K10 Multi-Cluster*

Hardware and Software

Table 1 provides the hardware specifications for the equipment used in this validation.

	Item	Description	Version	Function
Equinix Near-Cloud Data Center	Hitachi VSP 5200	1 TB cache (2) 20-core MPUs (4) RAID6 6D+2P parity groups (1) 10 GbE iSCSI port	SVOS RF 9.8.6 90-09-01-00/01	Storage system used to store application data.
Equinix Near-Cloud Data Center	Hitachi Advanced Server DS220	(2) 18-core Intel Xeon Gold 6140 @ 2.3 GHz 128 GB cache (1) Intel Ethernet Network Adapter XXV710	BMC 4.70.06 BIOS S5BH3B22.H00	4-node vSphere cluster used to deploy 5-nodes near-cloud Red Hat OpenShift cluster.
Equinix Near-Cloud Data Center	Cisco Nexus C93180YC-FX	Cisco Nexus C93180YC-FX 10 GbE Switch	NXOS 9.3(4)	Network switch at the data center that serviced the Direct Connect to AWS.
AWS	AWS EC2	(4) Intel Xeon Platinum 8000 series processor, 16 GB RAM	Instance type: t3.xlarge AMI Name: rhcos-411.86.202212072103-0-x86_64 AMI ID: ami-0298a5395cfd69001	5-nodes Red Hat OpenShift cluster in the cloud.
AWS	AWS S3	(1) Standard S3 bucket	N/A	Kasten K10 Multi-Cluster Global Location profile.

Table 1*: Hardware Components*

Table 2 provides the software specifications used in this validation.

Item	Version	Function
VMware vSphere	7.0 U2 (17867351)	Hypervisor operating system
VMware vCenter Server Appliance	7.0 U3 (18700403)	Management interface for vSphere cluster
Red Hat OpenShift Cluster	4.11.25	Red Hat OpenShift Cluster deployed in near-cloud and AWS.
Red Hat Enterprise Linux CoreOS (RHCOS)	4.11	RHCOS is the Operating System for Control Plane and Worker nodes.
Hitachi Storage Plug-in for Containers	3.11	HSPC plugin integrates Kubernetes or OpenShift with Hitachi storage systems using Container Storage Interface.
Kasten K10 Multi-Cluster	5.5.8	Kasten K10 Multi-Cluster Manager is a data management platform from Veeam which provides backup operation, disaster recovery, and application mobility for OpenShift applications across multiple clusters.
MySQL	5.7.41	A stateful database application used to validate data consistency on both Equinix and AWS.

Table 2*: Software Components*

Test Scenarios

Table 3 lists the test scenarios performed in the validation.

#	Description	Success Criteria
1	Prepare the environment: 1. Deploy two Red Hat OpenShift Clusters: One in a VMware environment in near-cloud and another in AWS. 2. Define storage, network, and iSCSI connection. 3. Use one Dynamic Provisioning (DP) pool to provision persistent volume for stateful application in near-cloud and AWS. 4. Deploy HSPC in both clusters. 5. Deploy Kasten K10 and K10 Multi-Cluster in both clusters. 6. Discover AWS cluster from Kasten K10 Multi-Cluster Manager deployed in near-cloud.	Environment is set up as per specifications.
2	Deploy a stateful application in the RHOCP clusters. This test case is performed in near-cloud as well as in AWS. The persistent volume is provisioned to both the Red Hat OpenShift Container Platform clusters from the same Hitachi VSP 5200 storage system located in near-cloud. 1. Define the storage class for the VSP 5200 storage system with the required settings. 2. Deploy MySQL database as a stateful application on Red Hat OpenShift Container Platform with persistent volume claim. 3. Create a new table and ingest new records.	Persistent volume from the VSP storage system is provisioned in Red Hat OpenShift Container Platform cluster in near-cloud as well as AWS successfully. Stateful application is deployed successfully.
3	Migrate a stateful application across OpenShift clusters using Kasten K10 Multi-Cluster: 1. Ingest data into MySQL application in near-cloud. 2. Create an S3 bucket in AWS. 3. Create a global location profile using this bucket. 4. Create a global snapshot policy. 5. Create a global distribution for snapshot policy and add both clusters. 6. Run the snapshot policy for the MySQL application to take the backup. 7. Create a global import policy for restore. 8. Create a global distribution for import policy and add both clusters. 9. Run the policy to restore the application in the target cluster. 10. Verify that the MySQL application is being restored and the ingested data is visible to the target MySQL environment.	Backup taken in near-cloud Red Hat OpenShift Container Platform cluster can be restored in Red Hat OpenShift Container Platform cluster in AWS using Kasten K10 Multi-Cluster.
4	Migrate a stateful application across OpenShift Clusters using HSPC (this test case is performed manually instead of Kasten K10): 1. Ingest data into MySQL application in near-cloud. 2. Create a Kubernetes volume snapshot. 3. Create PV and PVC of the snapshot volume. 4. Create a clone PVC using the PVC created in step 3 as the source PVC. 5. Use the clone as a volume source to deploy MySQL stateful application in the Red Hat OpenShift Container Platform cluster on AWS. 6. Verify that the ingested data is visible to the target MySQL environment.	Snapshot created in near-cloud Red Hat OpenShift Container Platform cluster can be manually restored in the Red Hat OpenShift Container Platform cluster in AWS.
5	Recover from ransomware attack: This test case is performed manually instead of Kasten K10. The Data Retention Utility feature is set on the snapshot volume to protect the backup from any write operations and define the data retention term for the protected volumes. 1. Ingest data into MySQL application in near-cloud. 2. Create a Kubernetes volume snapshot. 3. Set DRU attribute in the snapshot volume using Command Control Interface (CCI). 4. Assume that application is affected by ransomware in near-cloud and must restore the data from the snapshot taken in step 2. 5. Create a PVC using the snapshot volume created in step 2. 6. Create a Kubernetes volume snapshot (snap-on-snap) of the PVC created in step 5. This creates a cascaded snapshot volume. 7. Create PVC of the cascaded snapshot (snap-on-snap) volume. 8. Create a clone PVC using the PVC created in step 7 as the source PVC.	Revert to clean stateful MySQL application from snapshot data with DRU.

#	Description	Success Criteria
	9. Use the clone PVC as a volume source to deploy MySQL stateful application in the Red Hat OpenShift Container Platform cluster in AWS. 10. Verify that the ingested data is visible to the target MySQL environment.	

Table 3: *Test Scenarios*

Guidelines and Recommendations

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

- While installing a Red Hat OpenShift cluster in a private environment (for example, in an existing Amazon Virtual Private Cloud with a specific AWS Identify and Access Management user), use "*CredentialMode*" to set as "Manual" in the install-config.yaml file. The default mode is "Mint", which assumes that you have administrative privileges.
- While running the OpenShift installation, install-config.yaml file is used by the installer. You must keep a backup of this file. If the installation fails and must be re-run, copy the OpenShift installer and install-config.yaml to a new directory and then run from there. You must not re-use the same directory, or else X.509 certificate error occurs.
- Prepare a separate node outside the cluster for cluster deployment and install OpenShift CLI (oc) command to interact with OpenShift Container Platform for administration.
- While migrating an application using Kasten K10 across clusters, a location profile is mandatory. Without the location profile, import policy would not generate, and restoration is not possible to other clusters. However, to restore an application in the same cluster, a location profile is not required.
- While building a POD with persistent volume, HSPC automatically performs a series of tasks such as provisioning the volume, creating an iSCSI target (or FC host group), attaching the volume to it, discovering the volume on the target node, and then attaching the volume as a block device or creating a file system on it.
- In Kubernetes environment, a "VolumeSnapshot" object cannot be attached to a POD because it is not a persistent volume. To access the snapshot data, create a clone volume and then attach the clone volume to a POD.
- Retention time cannot be reduced while DRU setting is active on a volume.

Validation Results

This section shows the 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.

The test environment consists of two multi-node Red Hat OpenShift Clusters deployed using IPI method in a near-cloud VMware environment and AWS. You must configure the following components for validation of test cases:

- Configure physical LAN and iSCSI connections for OpenShift clusters.
- Provision DP pool to be used for persistent volume from VSP 5200 storage system.
- Deploy two Red Hat OpenShift Clusters: One in near-cloud VMware environment and another in AWS.
- Install HSPC.
- Deploy Kasten K10 Multi-Cluster.

Deploy Red Hat OpenShift Cluster in Near-Cloud

In this configuration, the cluster is installed using the IPI method in a VMware environment.

Prerequisites

Note that the following prerequisites are outside the scope of this document, so we do not describe them in detail. For more information, see: https://docs.openshift.com/container-platform/4.11/installing/installing_vsphere/installing-vsphere-installer-provisioned.html#installing-vsphere-installer-provisione.

- The OpenShift Container Platform installer requires access to port 443 on the vCenter and ESXi hosts. Verify that port 443 is accessible.
- In OpenShift Container Platform 4.11, internet access is required to install the cluster using IPI method.
- Use DHCP for the network and ensure that the DHCP server is configured to provide persistent IP addresses to the cluster machines. All nodes must be in the same VLAN.
- The installation in vSphere requires two static IP addresses:
  - The API IP address is used to access the cluster API.
  - The Ingress IP address is used for cluster ingress traffic. You must create DNS records for these two static IP addresses in the appropriate DNS server.
- Use a separate Red Hat Enterprise Linux virtual machine to trigger the OpenShift deployment. This node is also used as the Kubernetes admin node.
- Install OpenShift CLI (oc) on the admin node to interact with OpenShift Container Platform from a command-line interface.

Add vCenter Root CA Certificates

The installation program requires access to vCenter API; therefore, you must add vCenter trusted root CA certificates in the admin node system trust before installing the OpenShift Container Platform cluster.

```
# wget https://vcsa.juno.com/certs/download.zip
# unzip download.zip
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
cp: overwrite '/etc/pki/ca-trust/source/anchors/f67dd544.0'? y
cp: overwrite '/etc/pki/ca-trust/source/anchors/f67dd544.r0'? y
# update-ca-trust extract
```

Generate a Key Pair for Cluster Node SSH Access

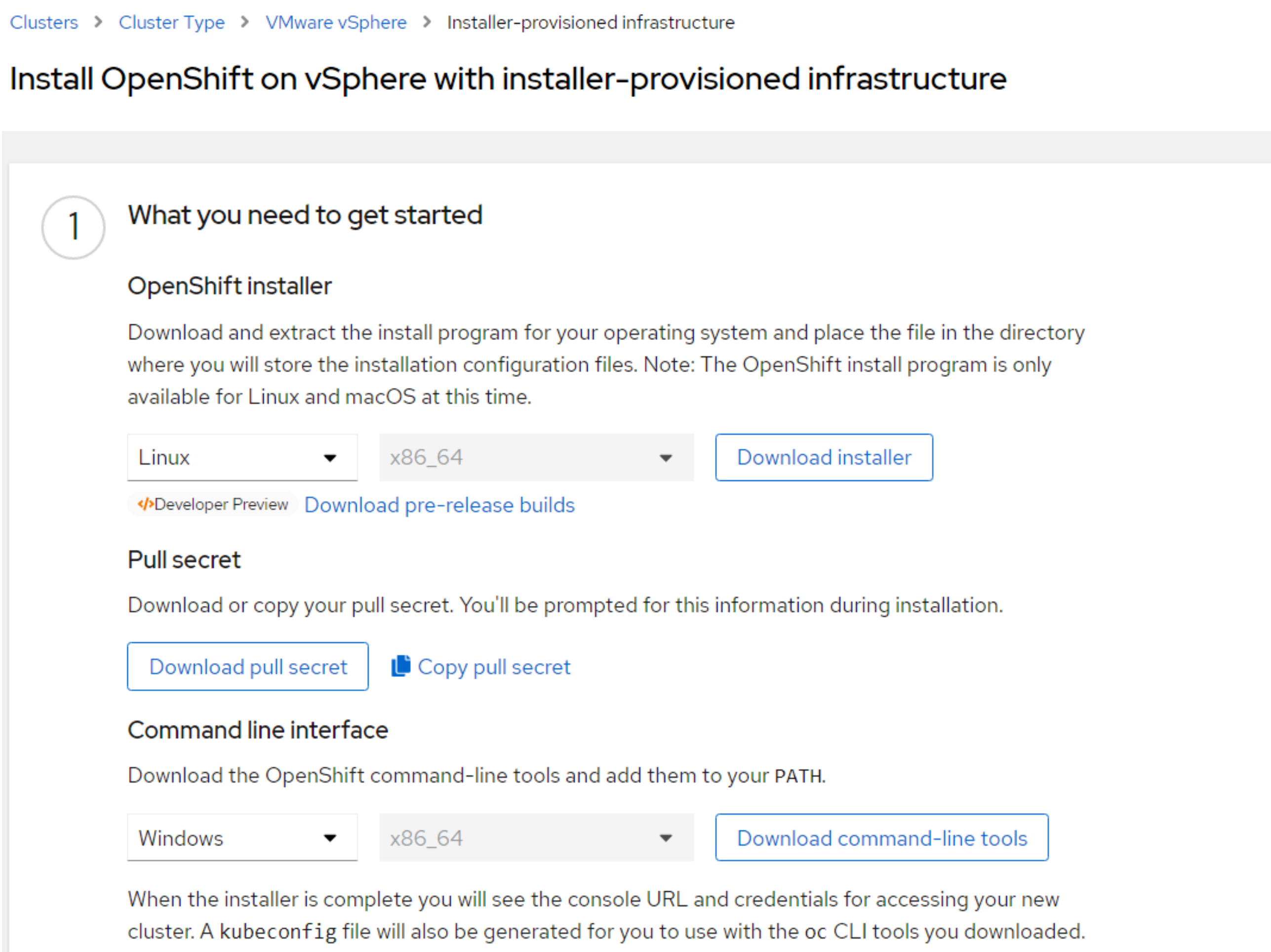
1. To generate a key pair, run the following command:
   ```
   # ssh-keygen -t ed25519 -N '' -f ~/.ssh/id_ed25519
   # eval "$(ssh-agent -s)"
   Agent pid 1199721
   ```

2. To view the public SSH key, run the following command:
   ```
   $ cat ~/.ssh/id_ed25519.pub
   ```

Obtain the Installation Program

You can download the latest OpenShift Installer from the Red Hat OpenShift Cluster Manager site. To download older versions such as v4.11.25, see: https://mirror.openshift.com/pub/openshift-v4/clients/ocp/4.11.25/.

1. Open the Infrastructure Provider page on the Red Hat OpenShift Cluster Manager site.
2. Navigate to download OpenShift Installer.
3. Pull secret and OpenShift command line interface.



Create an Install Config File

To install the OpenShift Cluster, prepare the install config file as follows:

```
# ./openshift-install create install-config --log-level=debug
DEBUG OpenShift Installer 4.11.25
DEBUG Built from commit b1b244444835f9a3fd2c5e6717db9ba6d18607be
? Platform vsphere
? vCenter vcsa.juno.com
? Username administrator@vsphere.local
? Password [? for help] ***********
INFO Connecting to vCenter vcsa.juno.com
? Datacenter SV10
INFO Defaulting to only available cluster: DR
? Default Datastore vsp-5200-lun-fef0
? Network VM Network
? Virtual IP Address for API 172.23.31.180
? Virtual IP Address for Ingress 172.23.31.181
```

```
DEBUG   Generating Base Domain...
? Base Domain juno.com
? Cluster Name ocpcluster
DEBUG   Generating Pull Secret...
? Pull Secret [? for help]
**********************************************************************************************
**********************************************************************************************
DEBUG Generating Install Config...
INFO Install-Config created in: .
```
(Command output is truncated)

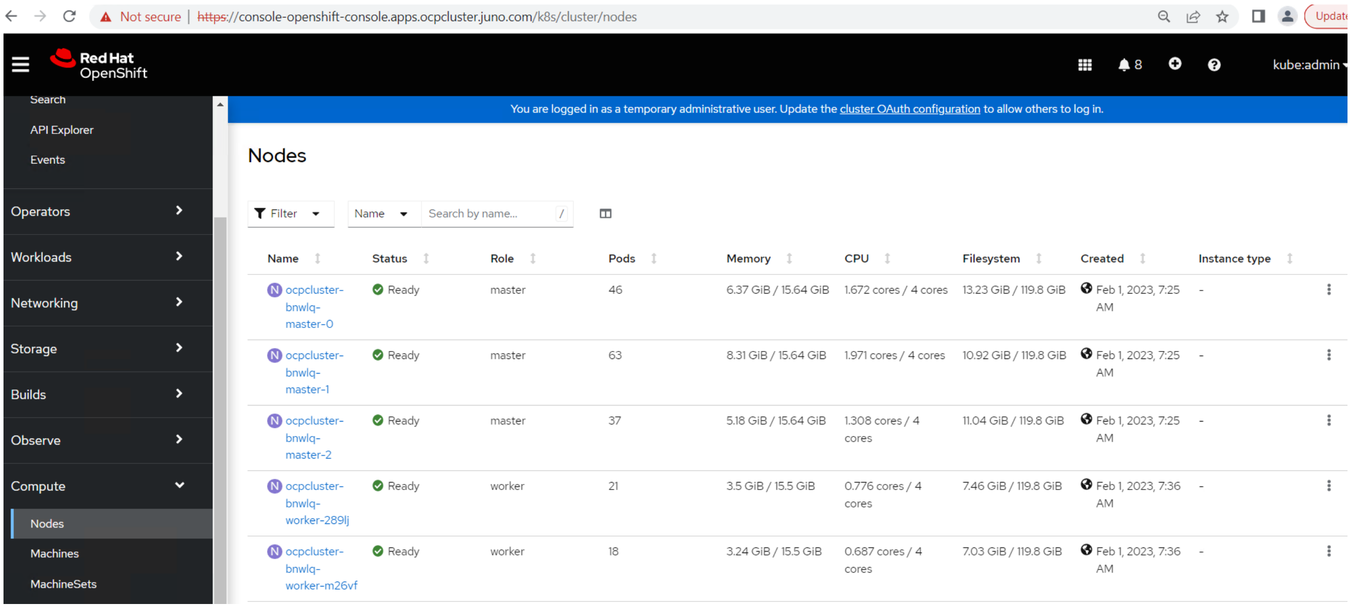
```
# cat install-config.yaml
apiVersion: v1
baseDomain: juno.com
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 2
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform: {}
  replicas: 3
metadata:
  creationTimestamp: null
  name: ocpcluster
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  vsphere:
    apiVIP: 172.23.31.180
    cluster: DR
    datacenter: SV10
    defaultDatastore: vsp-5200-lun-fef0
    ingressVIP: 172.23.31.181
    network: VM Network
    password: password1
    username: administrator@vsphere.local
    vCenter: vcsa.juno.com
publish: External
pullSecret:
'{"auths":{"cloud.openshift.com":{"auth":"b3BlbnNoaWZ0LXJlbGVhc2UtZGV2K2hkc19pbGFiX2tvbDFkZXV
jbHJsbmV3c2N5bmgwMnl0Y3lsMWVhbTpKOTNZSUpWVkFWMVRRNDBCNDRMSjdNREFSTU81VUNZNjRSV0JGTDZaWVlWRDRK
==","email":"abc1.xz@hds.com"}}}'
sshKey: |
  ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIDgfOKgebOX/IGyLR9N3NCVzPmPEdhN0XOt2/ScIloNm
root@linuxnfscl2
```
(Command output is truncated)

Create a Cluster

Navigate to the directory that contains the installation program and run the following **openshift-install** command:

```
# ./openshift-install create cluster –log-level=debug
```

After installation, access the console from: https://console-openshift-console.apps.ocpcluster.juno.com.



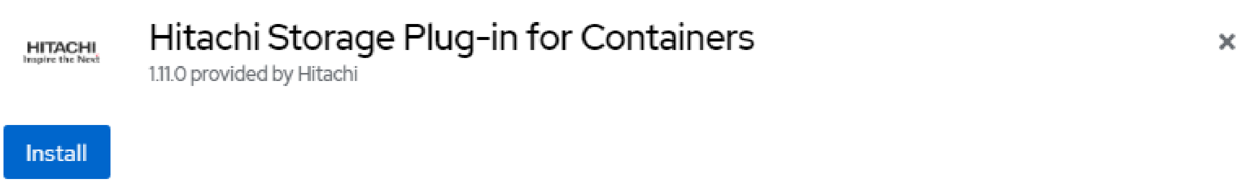
Install Hitachi Storage Plug-in for Containers

After successfully deploying the OpenShift cluster, install the HSPC software.

Deploy HSPC using OperatorHub

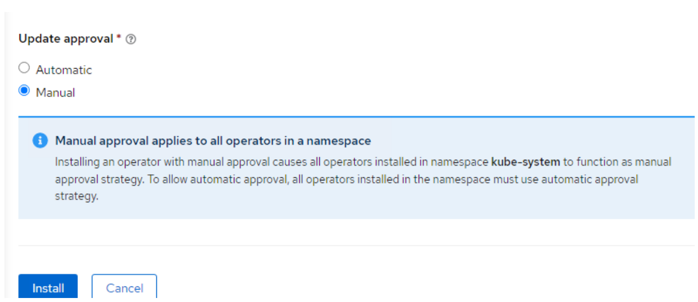
To deploy the HSPC plugin in OpenShift using OperatorHub, complete the following steps:

1. Log in to Red Hat OpenShift console and select **Operators** and click on **OperatorHub**.
2. Search **"Hitachi Storage Plug-in for Containers"** in **All Items** option and click on **"Hitachi Storage Plug-in for Containers"** displayed in the search result.
3. In Hitachi Storage Plug-in for Containers page, click on **Install**.

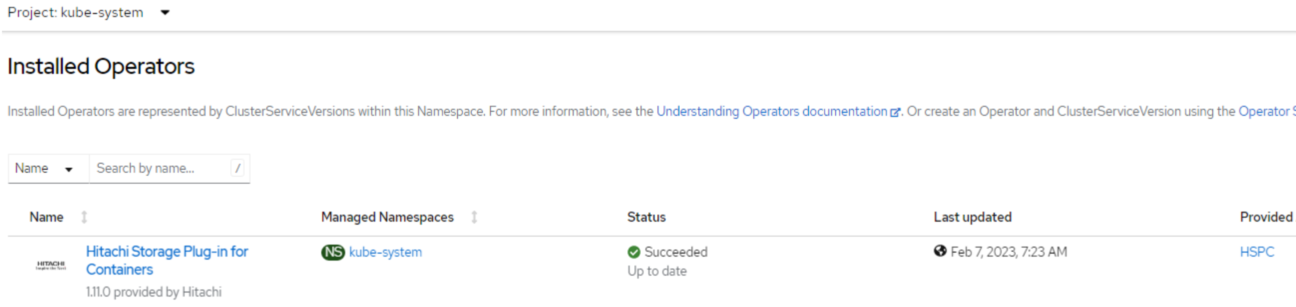


4. In the Install Operator window, enter the following information and click **Install**.

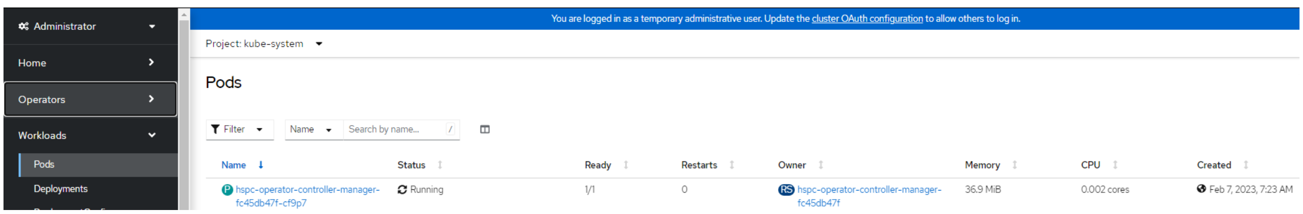
   - Installation mode: Select **A specific namespace on the cluster**.
   - Installed Namespace: Namespace where you want to install HSPC. Select the **kube-system** namespace.
   - Update approval: Select **Manual**.



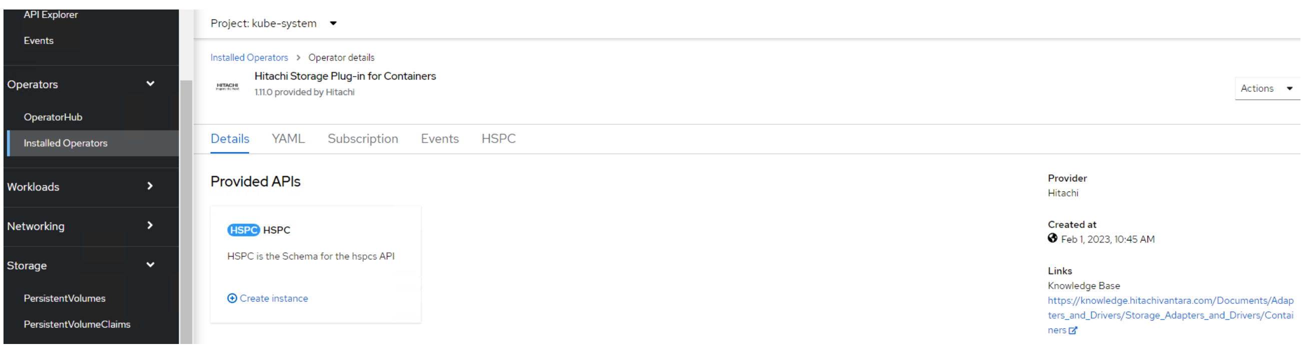
5. Wait until the installed operator status is ready to use. From the Red Hat OpenShift console, navigate to Operators and click **Installed Operators**. The following screenshot shows the status of the operator after a successful installation.



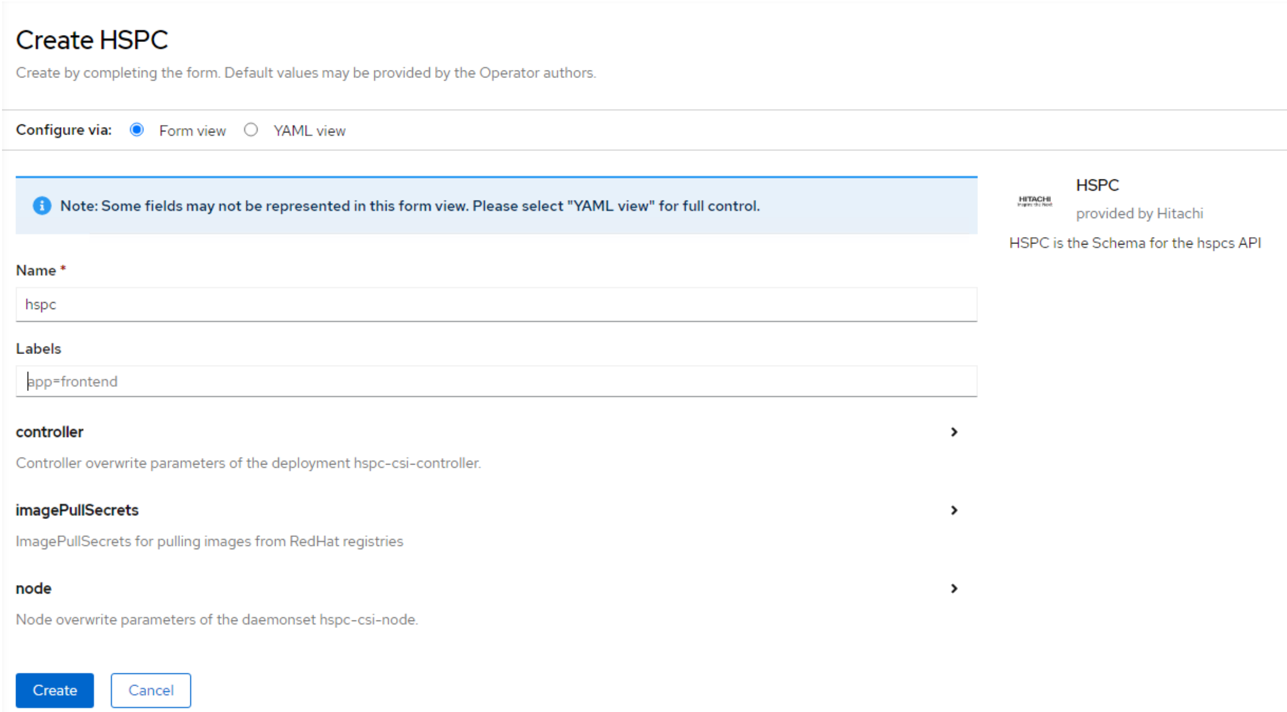
6. From the console, navigate to Workloads, click **Pods**, and ensure that the status of the operator pod is running.



7. Create an HSPC Instance. From the console, navigate to Operators and click **Installed Operators**. Open the Operator details window and click **Create instance**.



8. From the Create HSPC window, enter any name and click **Create**.



9. Verify that the Ready status of HSPC is **true**.

```
# oc get hspc -n kube-system
NAME    READY   AGE
hspc    true    3m42s
```

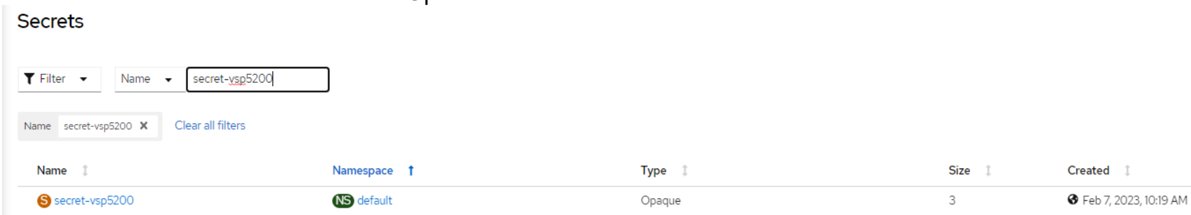
Create Storage Class and Volume SnapshotClass

After installing the HSPC plugin, create Storage Class to provision persistent volume from the VSP 5200 storage system. A Volume SnapshotClass is required to take point in time snapshot. Complete the following steps:

1. Create a secret for HSPC.
   a. From the Red Hat OpenShift console, navigate to Workloads, click **Secret**, and then click **Create** to open a **YAML** window.
   b. Enter the storage URL, username, and password in base64 format and click **Create** to generate secret. The following shows a sample secret YAML:

```
apiVersion: v1
kind: Secret
metadata:
  name: secret-vsp5200
  namespace: default
type: Opaque
data:
  url: aHR0cHM6Ly8xNzIuMjMuMzAuMTA=
  user: a3ViZXJuZXRlcw==
  password: a3ViZXJuZXRlcw==
```

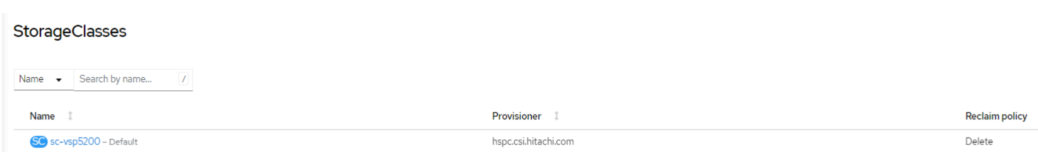
Status of the secret from the Red Hat OpenShift console:



2. Create a storage class for the VSP 5200 storage system.
    a. From the Red Hat OpenShift console, navigate to Storage, click **StorageClasses**, and then click **create StorageClass**.
    b. Enter the storage information (Pool ID, Port Number, and so on) and click **Create**. The following shows a sample storage class YAML:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-vsp5200
  annotations:
    kubernetes.io/description: Hitachi Storage Plug-in for Containers
provisioner: hspc.csi.hitachi.com
reclaimPolicy: Delete
volumeBindingMode: Immediate
allowVolumeExpansion: true
parameters:
  serialNumber: "40028"
  poolID: "0"
  portID : CL1-C
  connectionType: iscsi
  csi.storage.k8s.io/fstype: ext4
  csi.storage.k8s.io/node-publish-secret-name: "secret-vsp5200"
  csi.storage.k8s.io/node-publish-secret-namespace: "default"
  csi.storage.k8s.io/provisioner-secret-name: "secret-vsp5200"
  csi.storage.k8s.io/provisioner-secret-namespace: "default"
  csi.storage.k8s.io/controller-publish-secret-name: "secret-vsp5200"
  csi.storage.k8s.io/controller-publish-secret-namespace: "default"
  csi.storage.k8s.io/node-stage-secret-name: "secret-vsp5200"
  csi.storage.k8s.io/node-stage-secret-namespace: "default"
  csi.storage.k8s.io/controller-expand-secret-name: "secret-vsp5200"
  csi.storage.k8s.io/controller-expand-secret-namespace: "default"
```

Status of the storage class:

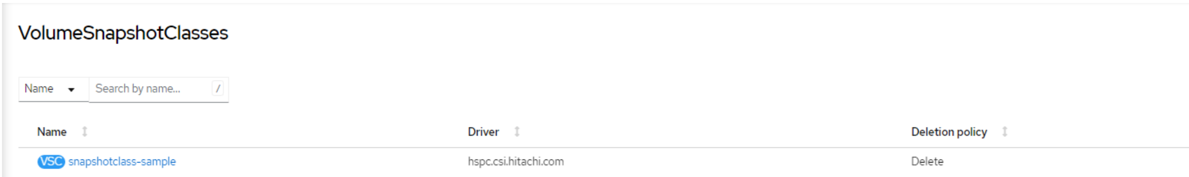


3. Create a volume snapshot class for the VSP 5200 storage system.
    a. From the Red Hat OpenShift console, navigate to Storage, click **VolumeSnapshotClass**, and then click **create VolumeSnapshotClass**.
    b. Populate "VolumeSnapshotClass" YAML with the required information and click **Create**. The following shows a sample YAML:

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: snapshotclass-sample
driver: hspc.csi.hitachi.com
deletionPolicy: Delete
```

```
parameters:
 poolID: "0"
 csi.storage.k8s.io/snapshotter-secret-name: "secret-vsp5200"
 csi.storage.k8s.io/snapshotter-secret-namespace: "default"
```

Status of the **VolumeSnapshotclass**:



Install Kasten K10 in Red Hat OpenShift Cluster

This section describes the process of deploying Kasten K10 in the near-cloud OpenShift cluster. Kasten K10 is integrated with HSPC plugin to provision backup and restore target. When the installation is complete, additional steps are performed to enable Multi-Cluster on Kasten K10.

1. Install the Helm package manager.
   a. To deploy Kasten K10, your client machine in the OpenShift Container Platform must have access to the helm command. To download the script for installing Helm, run the curl command:

```
[root@linuxnfscl2 sw3]# curl -fsSL -o get_helm.sh https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3
[root@linuxnfscl2 sw3]#
[root@linuxnfscl2 sw3]# chmod 700 get_helm.sh
```

   b. Run the script.

```
[root@linuxnfscl2 sw3]# ./get_helm.sh
Downloading https://get.helm.sh/helm-v3.11.1-linux-amd64.tar.gz
Verifying checksum... Done.
Preparing to install helm into /usr/local/bin
helm installed into /usr/local/bin/helm
[root@linuxnfscl2 sw3]#
```

2. Configure Helm chart repositories: Add a Helm chart repository to obtain the Kasten K10 chart.

```
[root@linuxnfscl2 sw3]# helm repo add kasten https://charts.kasten.io/
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /sw5/auth/kubeconfig
"kasten" has been added to your repositories
[root@linuxnfscl2 sw3]#
```

3. Add an annotation to “VolumeSnapshotClass”.
   a. Add the Kasten K10 annotation as follows:

```
[root@linuxnfscl2 sw3]# oc annotate volumesnapshotclass snapshotclass-sample  k10.kasten.io/is-snapshot-class=true
volumesnapshotclass.snapshot.storage.k8s.io/snapshotclass-sample annotated
[root@linuxnfscl2 sw3]#
```

   b. Verify the status after adding the Kasten K10 annotation.

```
[root@linuxnfscl2 sw3]# oc describe volumesnapshotclass snapshotclass-sample
Name:             snapshotclass-sample
Namespace:
Labels:           <none>
Annotations:      k10.kasten.io/is-snapshot-class: true
API Version:      snapshot.storage.k8s.io/v1
Deletion Policy:  Delete
Driver:           hspc.csi.hitachi.com
Kind:             VolumeSnapshotClass
```

   (Output is truncated)

4. Create a namespace (kasten-io) for installing Kasten K10.
5. Set the storage class sc-vsp5200 as default for installing Kasten K10.

Status of the Storage Class before:

```
[root@linuxnfscl2 sw3]# oc get storageclass
NAME             PROVISIONER                   RECLAIMPOLICY   VOLUMEBINDINGMODE      ALLOWVOLUMEEXPANSION   AGE
sc-vsp5200       hspc.csi.hitachi.com          Delete          Immediate              true                   5d20h
thin (default)   kubernetes.io/vsphere-volume  Delete          Immediate              false                  11d
thin-csi         csi.vsphere.vmware.com        Delete          WaitForFirstConsumer   true                   11d
[root@linuxnfscl2 sw3]#
```

To set the storage class sc-vsp5200 as default, run the following oc patch command:

```
[root@linuxnfscl2 sw3]# oc patch storageclass sc-vsp5200 -p '{"metadata": {"annotations": {"storageclass.kubernetes.io/is-default-class": "true"}}}'
storageclass.storage.k8s.io/sc-vsp5200 patched
[root@linuxnfscl2 sw3]#
[root@linuxnfscl2 sw3]# oc get storageclass
NAME                   PROVISIONER                    RECLAIMPOLICY   VOLUMEBINDINGMODE      ALLOWVOLUMEEXPANSION   AGE
sc-vsp5200 (default)   hspc.csi.hitachi.com           Delete          Immediate              true                   5d20h
thin (default)         kubernetes.io/vsphere-volume   Delete          Immediate              false                  11d
thin-csi               csi.vsphere.vmware.com         Delete          WaitForFirstConsumer   true                   11d
[root@linuxnfscl2 sw3]#
```

6. Before installing Kasten K10, run a Pre-Flight Checks script.
    a. To verify whether the Kubernetes settings meet the Kasten K10 requirements, run Pre-Flight Checks before installing Kasten K10 in Red Hat OpenShift container environment. Pre-Flight checks verify the following items:
        - Whether available “StorageClass” is cataloged.
        - Whether a CSI provisioner exists and basic verification is conducted.
    b. For Pre-Flight checks, run the following command:

```
# curl https://docs.kasten.io/tools/k10_primer.sh | bash
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  7967  100  7967    0     0  31615      0 --:--:-- --:--:-- --:--:-- 31615
Namespace option not provided, using default namespace
Checking for tools
 --> Found kubectl
 --> Found helm
 --> Found jq
```

(Output is truncated)

```
Validating Provisioners:
hspc.csi.hitachi.com:
  Is a CSI Provisioner  -  OK
  Missing/Failed to Fetch CSIDriver Object
  Storage Classes:
    sc-vsp5200
      Valid Storage Class  -  OK
  Volume Snapshot Classes:
    k10-clone-snapshotclass-sample
    snapshotclass-sample
      Has k10.kasten.io/is-snapshot-class annotation set to true  -  OK
      Has deletionPolicy 'Delete'  -  OK
```

(Output is truncated)

```
serviceaccount "k10-primer" deleted
clusterrolebinding.rbac.authorization.k8s.io "k10-primer" deleted
job.batch "k10primer" deleted
```

    c. To verify the snapshot behavior, run the following command by specifying storage class with “-s” option. Verify that the command does not return any error.

```
# curl -s https://docs.kasten.io/tools/k10_primer.sh  | bash /dev/stdin -c "storage
csi-checker -s sc-vsp5200 --runAsUser=1000"
Namespace option not provided, using default namespace
Checking for tools
 --> Found kubectl
 --> Found helm
 --> Found jq
```

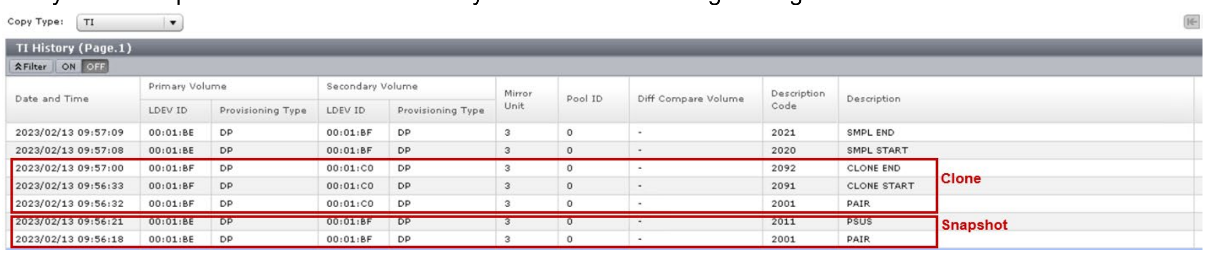
(Output is truncated)

```
Running K10Primer Job in cluster with command-
     ./k10tools primer storage csi-checker -s sc-vsp5200 --runAsUser=1000
serviceaccount/k10-primer created
```

(Output is truncated)

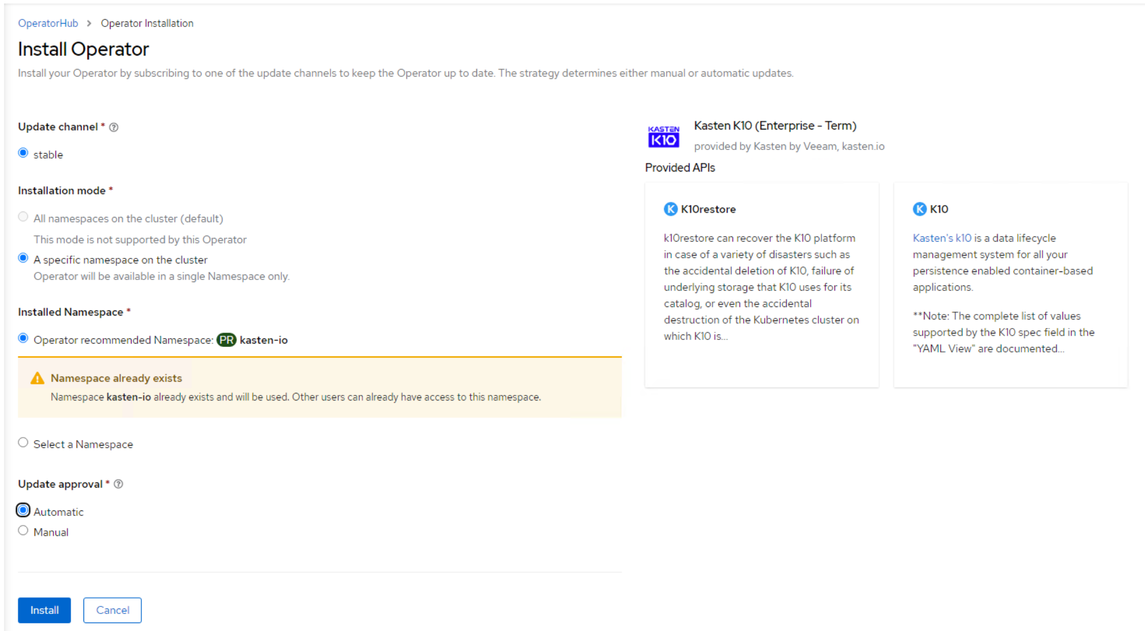
```
Creating application
  -> Created pod (kubestr-csi-original-podj7glq) and pvc (kubestr-csi-original-
pvcvnk5t)
Taking a snapshot
  -> Created snapshot (kubestr-snapshot-20230504121654)
Restoring application
  -> Restored pod (kubestr-csi-cloned-podhc245) and pvc (kubestr-csi-cloned-
pvcfpcv8)
Cleaning up resources
CSI Snapshot Walkthrough:
  Using annotated VolumeSnapshotClass (snapshotclass-sample)
  Successfully tested snapshot restore functionality.  -  OK
serviceaccount "k10-primer" deleted
clusterrolebinding.rbac.authorization.k8s.io "k10-primer" deleted
job.batch "k10primer" deleted
```

d. Verify that a snapshot and clone are correctly created on the Storage Navigator.

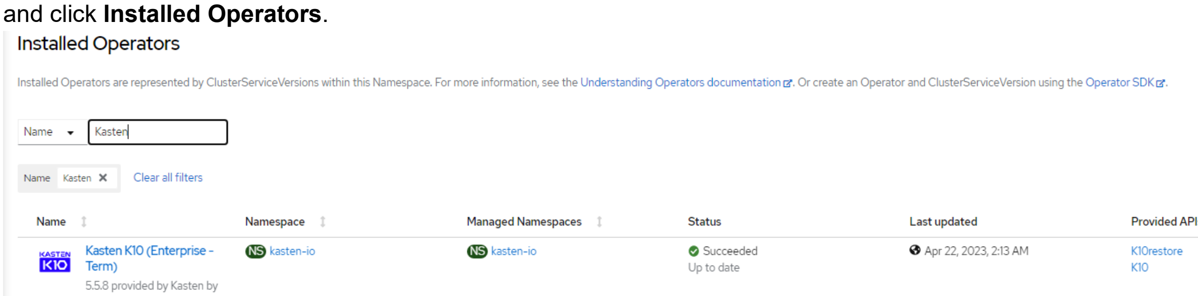


Date and Time	Primary Volume		Secondary Volume		Mirror Unit	Pool ID	Diff Compare Volume	Description Code	Description
	LDEV ID	Provisioning Type	LDEV ID	Provisioning Type					
2023/02/13 09:57:09	00:01:BE	DP	00:01:BF	DP	3	0	-	2021	SMPL END
2023/02/13 09:57:08	00:01:BE	DP	00:01:BF	DP	3	0	-	2020	SMPL START
2023/02/13 09:57:00	00:01:BF	DP	00:01:C0	DP	3	0	-	2092	CLONE END
2023/02/13 09:56:33	00:01:BF	DP	00:01:C0	DP	3	0	-	2091	CLONE START
2023/02/13 09:56:32	00:01:BF	DP	00:01:C0	DP	3	0	-	2001	PAIR
2023/02/13 09:56:21	00:01:BE	DP	00:01:BF	DP	3	0	-	2011	PSUS
2023/02/13 09:56:18	00:01:BE	DP	00:01:BF	DP	3	0	-	2001	PAIR

7. Install Kasten K10. To deploy Kasten K10 in OpenShift using OperatorHub, complete the following steps:
   a. Log in to Red Hat OpenShift console, select Operators, and click **OperatorHub**.
   b. Under All Items, search for "Kasten" and click **Kasten K10 (Enterprise - Term)**.
   c. In the Kasten K10 (Enterprise - Term) page, click **Install**, which opens the Install Operator window.
   d. In the Install Operator window, enter the following information and click **Install**.
      - Installation mode: Select **A specific namespace on the cluster**.
      - Installed Namespace: Namespace where the Kasten K10 must be installed. Select **kasten-io**.



e. Wait until the installed operator status is ready to use. In the Red Hat OpenShift console, navigate to Operators and click **Installed Operators**.

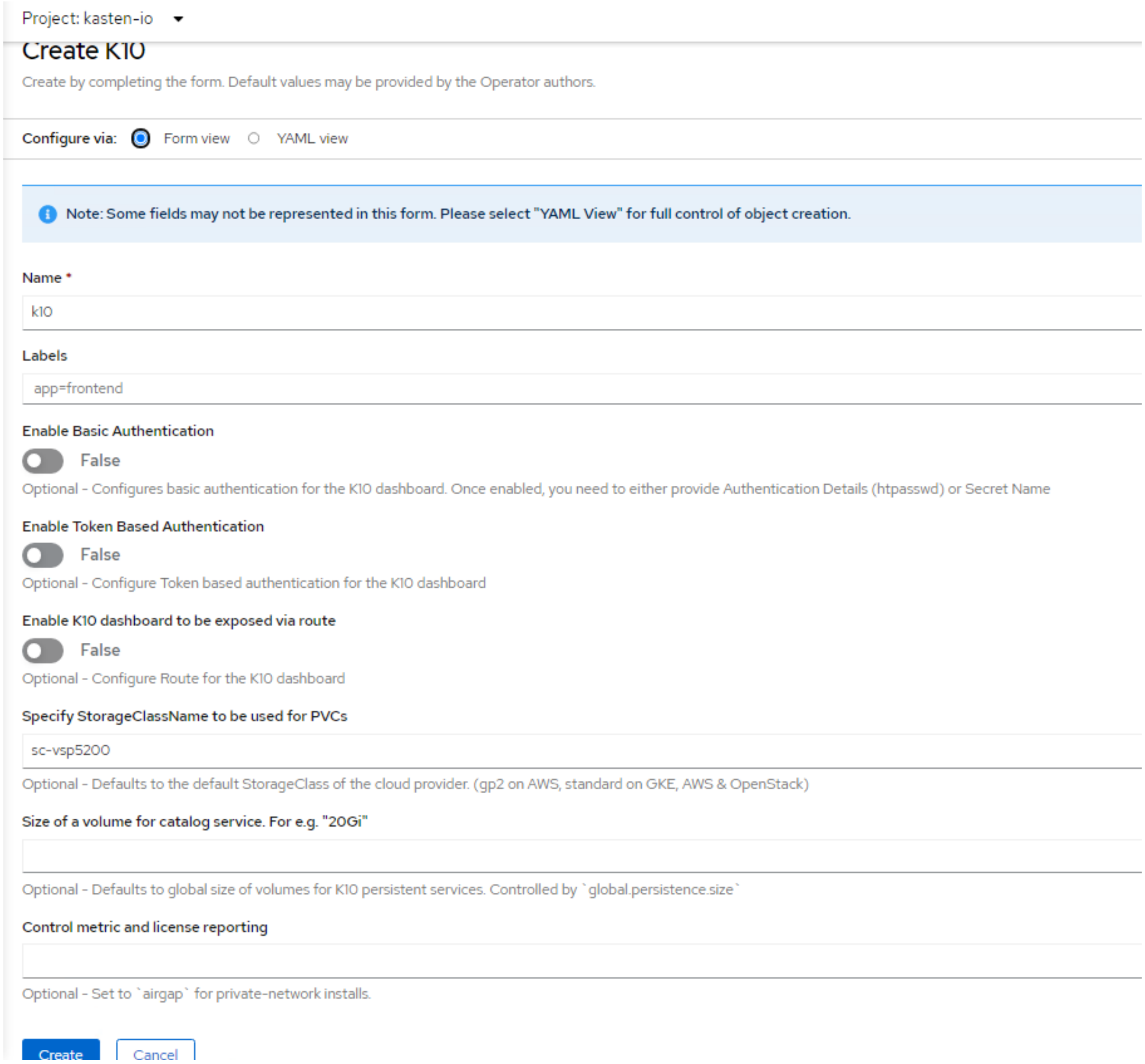


Status of Kasten K10 Operator from CLI:

```
[root@linuxnfscl2 sw3]# kubectl get pods --namespace kasten-io
NAME                                                      READY   STATUS    RESTARTS   AGE
k10-kasten-operator-term-rhmp-controller-manager-6bbf4d7d6k6rn9   2/2     Running   0          13m
[root@linuxnfscl2 sw3]#
```

f. From the Red Hat OpenShift console, select **Operators** and click **Installed Operators**.
g. From the Installed Operators menu, click **Kasten K10 (Enterprise - Term)**.
h. From the Kasten K10 (Enterprise - Term) page, click **Create Instance** which opens a Create K10 window.

i. Enter a Name, specify the Storage class name, and click **Create**.



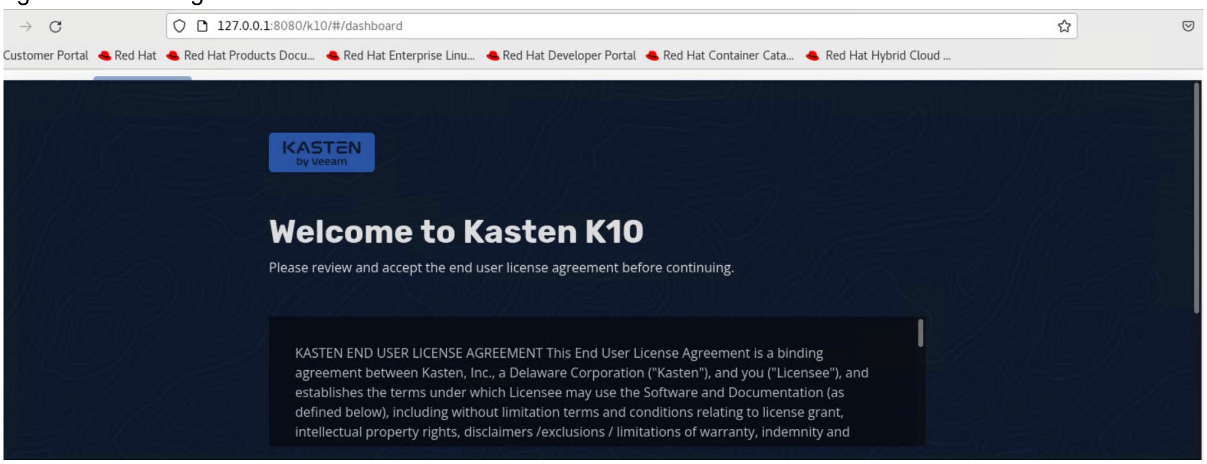
j. Verify whether the relevant Kasten K10 components are running.

```
[root@linuxnfscl2 sw3]# kubectl get pods --namespace kasten-io
NAME                                                      READY   STATUS    RESTARTS   AGE
aggregatedapis-svc-f4bfc797-5wpgs                         1/1     Running   0          108s
auth-svc-84479b86c-rm5hx                                  1/1     Running   0          109s
catalog-svc-5c7595c74b-zqd2n                              2/2     Running   0          108s
controllermanager-svc-68ff5957-4dd9v                      1/1     Running   0          109s
crypto-svc-5bd747dcd9-9fshj                               4/4     Running   0          108s
dashboardbff-svc-57595d7bcc-mhxwq                         2/2     Running   0          109s
executor-svc-6cfbf457cf-7ltn2                             2/2     Running   0          108s
executor-svc-6cfbf457cf-f6cfv                             2/2     Running   0          108s
executor-svc-6cfbf457cf-krjlc                             2/2     Running   0          108s
frontend-svc-55b4c48dfd-mnp5p                             1/1     Running   0          107s
gateway-9867fb979-qm5j2                                   1/1     Running   0          109s
jobs-svc-788c49585-vqqh4                                  1/1     Running   0          108s
k10-grafana-6b85fb7-ndqp9                                 0/1     Running   0          43s
k10-kasten-operator-term-rhmp-controller-manager-6bbf4d7d6k6rn9   2/2     Running   0          35m
kanister-svc-7759c5cc48-gzqx9                             1/1     Running   0          108s
logging-svc-7f87bc97db-9qz8l                              1/1     Running   0          108s
metering-svc-785459bcf7-2vmmz                             1/1     Running   0          108s
prometheus-server-6db457b46c-2lxx6                        2/2     Running   0          109s
state-svc-59f4c7845c-24k67                                2/2     Running   0          108s
[root@linuxnfscl2 sw3]#
```

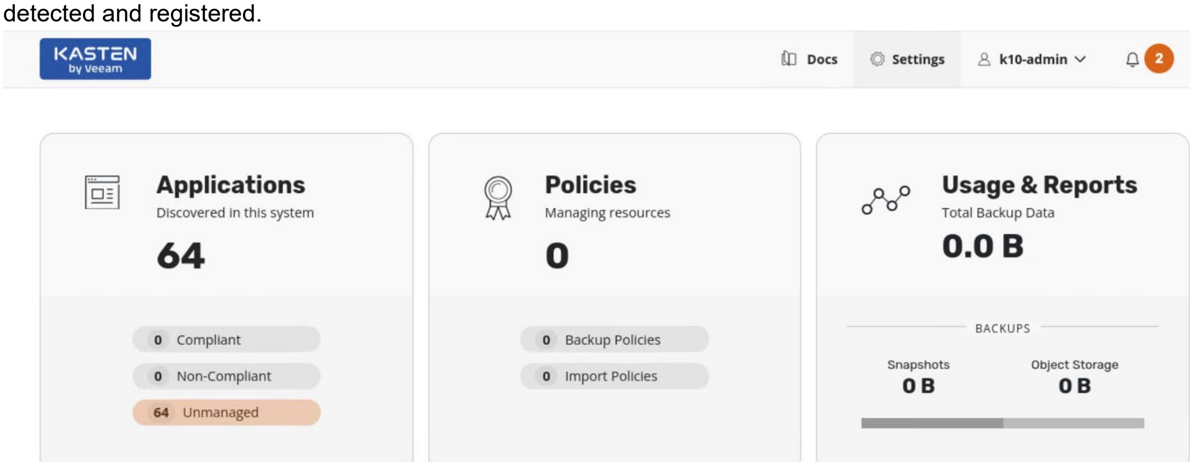
k. Run the following command and then access the Kasten K10 dashboard:

```
[root@linuxnfscl2 sw3]# kubectl --namespace kasten-io port-forward service/gateway 8080:8000
Forwarding from 127.0.0.1:8080 -> 8000
Forwarding from [::1]:8080 -> 8000
Handling connection for 8080
```

l. Access the Kasten K10 dashboard (http://127.0.0.1:8080/k10/#/) from the browser. Accept the end-user license agreement and log in.



The Applications option on the Kasten K10 dashboard contains a list of applications that were automatically detected and registered.



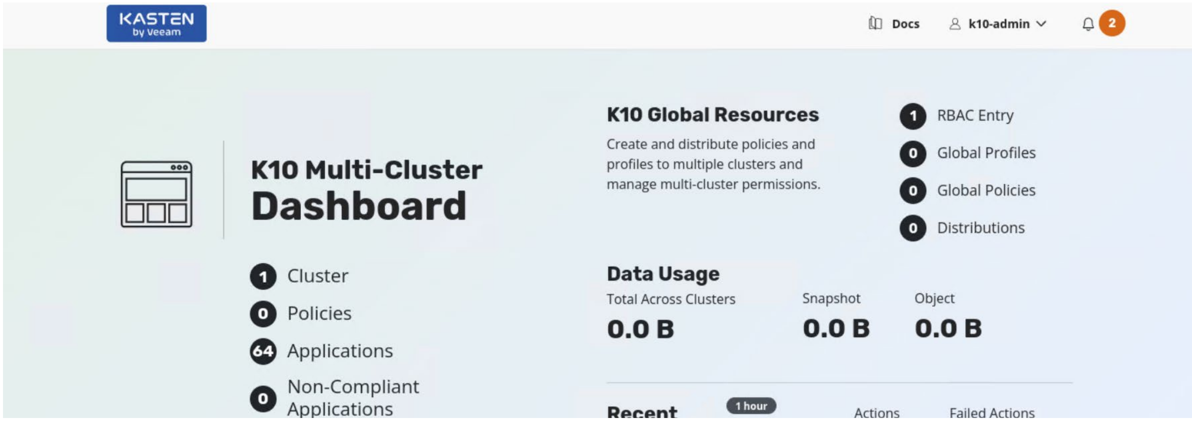
8. Install Kasten K10 Multi-Cluster Manager in OpenShift cluster at near-cloud.

a. Download the Kasten Multi-Cluster tool from: https://github.com/kastenhq/external-tools/releases.

b. Set the OpenShift cluster at near-cloud as Primary.

```
[root@linuxnfscl2 sw7]# ./k10multicluster setup-primary --name ocpcluster
Bootstrapping Primary Cluster...
Getting Primary Cluster Config...
Verifying cluster parameters: ocpcluster
Setting up primary multicluster configuration: ocpcluster
Setting up Primary Cluster Complete!
[root@linuxnfscl2 sw7]#
```

The K10 dashboard changed to K10 Multi-Cluster dashboard. Access the Kasten K10 Multi-Cluster dashboard using the same URL mentioned in step 7.



Deploy Red Hat OpenShift Cluster in AWS

The following links describe the OpenShift installation procedure with IPI method in AWS.

For prerequisites, see https://docs.openshift.com/container-platform/4.11/installing/installing_aws/preparing-to-install-on-aws.html.

If you do not have an AWS administrator account, you can deploy as an IAM user. See section 5.2.3 in: https://access.redhat.com/documentation/en-us/openshift_container_platform/4.11/html/installing/installing-on-aws.

To obtain the installation media and pull secret, see Deploy Red Hat OpenShift Cluster in Near-Cloud: Obtain the Installation Program.

1. Prepare an “install-config.yaml” file for installing OpenShift Cluster as follows:

```
credentialsMode: Manual
apiVersion: v1
baseDomain: hvcloudconnect.com
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform:
    aws:
      rootVolume:
        size: 200
        type: gp2
      type: t3.xlarge
      zones:
      - us-west-1a
  replicas: 2
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    aws:
      zones:
      - us-west-1a
      rootVolume:
        size: 200
        type: gp2
      type: t3.xlarge
  replicas: 3
metadata:
  creationTimestamp: null
  name: awscluster
networking:
```

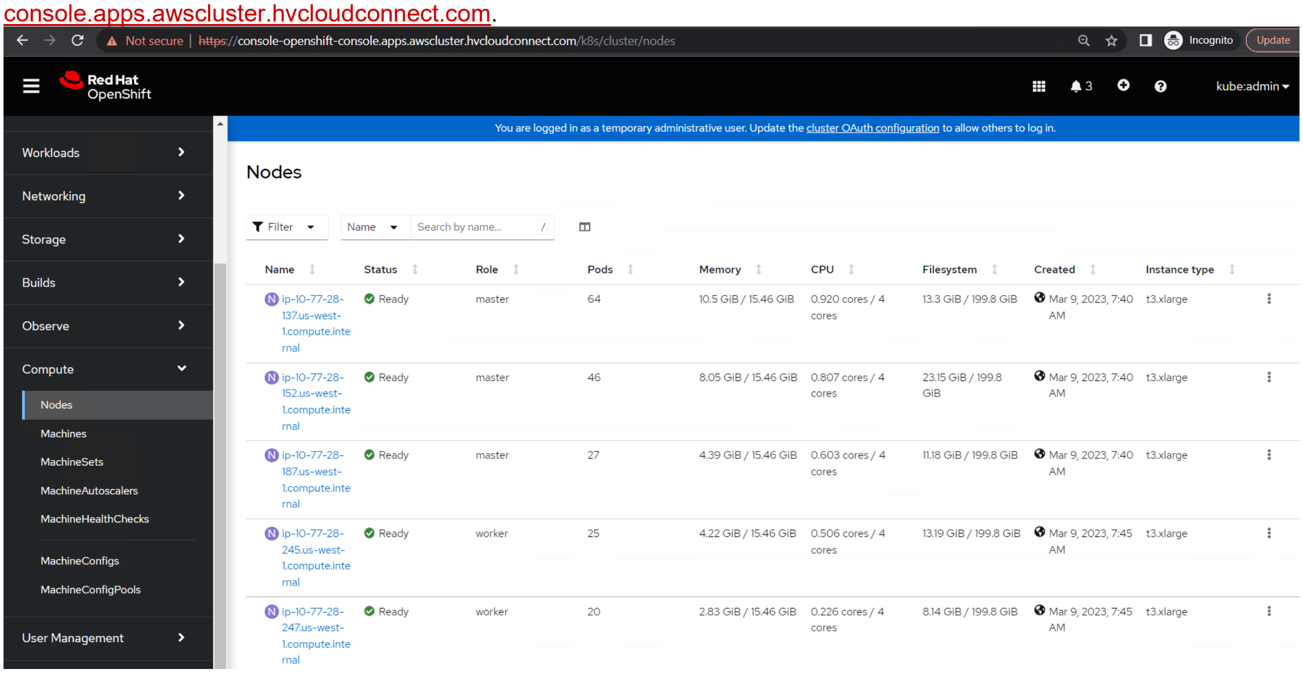


```
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.77.28.128/25
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-west-1
    subnets:
    - subnet-074541383711fd230

publish: Internal
pullSecret:
'{"auths":{"cloud.openshift.com":{"auth":"b3BlbnNoaWZ0LXJlbGVhc2UtZGV2K2hkc19pbGFiX2tvbDFk
ZXVfrgf==","email":"abc1.xz@hds.com"}}}'
sshKey: |
  ssh-rsa +uXAPvCfwTuiWu2+/GgGMBTUGwKLjcGYwdngSZW8e3C0Y5i/v root@ip-10-77-24-140.us-west-
1.compute.internal
```

2. Create the cluster. Change to the directory that contains the installation program and run the following openshift-install command:

```
#./openshift-install create cluster –log-level=debug
```

3. When the installation is complete, obtain the console URL from: https://console-openshift-console.apps.awscluster.hvcloudconnect.com.



The automated installation creates two load balancers in AWS. One Classic Load Balancer for Ingress traffic and one Network Load Balancer for API traffic.

Name	DNS name	State	VPC ID	Availability Zones	Type
a1609fb6b6c184ab8b6bb92155ca1192	internal-a1609fb6b6c184ab8b6bb92155ca1192-627184225.us-west-1.elb.amazonaws.com	-	vpc-07fed8ae295819772	us-west-1a (usw1-az3)	classic
awscluster-vbvwz-int	awscluster-vbvwz-int-4505eb3a44635dc4.elb.us-west-1.amazonaws.com	Active	vpc-07fed8ae295819772	us-west-1a (usw1-az3)	network

A Route53 private hosted zone entry is automatically created with the baseDomain and cluster name set in “install-config.yaml” file.

	Hosted zone name	Type	Create...	Record ...	Description	Hosted zone ID
○	awscluster.hvcloudconnect.com	Private	Route 53	5	Managed by Terraform	Z030164026DWTCVV189BM

In the host, zone 3 records are added and API requests are redirected to the Network Load Balancer.

☐	Record name	Type	Routin...	Di...	Alias	Value/Route traffic to
☐	api-int.awscluster.hvcloudconnect.com	A	Simple	-	Yes	awscluster-vbvwz-int-4505eb3a44635dc4.elb.us-west-1.amazonaws.com.
☐	api.awscluster.hvcloudconnect.com	A	Simple	-	Yes	awscluster-vbvwz-int-4505eb3a44635dc4.elb.us-west-1.amazonaws.com.

All other requests are forwarded to the Classic Load Balancer.

☐	Record name	Type	Routin...	Differe...	Alias	Value/Route traffic to
☐	*.apps.awscluster.hvcloudconnect.com	A	Simple	-	Yes	internal-a1609fb6b6c184ab8b6bb92155ca1192-627184225.us-west-

4. Verify the nodes, cluster version, and OpenShift URL as follows:

```
[root@ip-10-77-28-159 sw_ocp18]# oc get nodes
NAME                                        STATUS   ROLES    AGE   VERSION
ip-10-77-28-161.us-west-1.compute.internal  Ready    master   69m   v1.24.6+5658434
ip-10-77-28-167.us-west-1.compute.internal  Ready    worker   64m   v1.24.6+5658434
ip-10-77-28-206.us-west-1.compute.internal  Ready    master   68m   v1.24.6+5658434
ip-10-77-28-223.us-west-1.compute.internal  Ready    master   68m   v1.24.6+5658434
ip-10-77-28-231.us-west-1.compute.internal  Ready    worker   60m   v1.24.6+5658434
[root@ip-10-77-28-159 sw_ocp18]#
```

```
[root@ip-10-77-28-159 sw_ocp18]# oc get clusterversion
NAME      VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
version   4.11.25   True        False         75m     Cluster version is 4.11.25
[root@ip-10-77-28-159 sw_ocp18]#
```

```
[root@ip-10-77-28-159 sw_ocp18]# oc cluster-info
Kubernetes control plane is running at https://api.awscluster.hvcloudconnect.com:6443

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
[root@ip-10-77-28-159 sw_ocp18]#
```

Install HSPC in Red Hat OpenShift Cluster in AWS

To install HSPC in Red Hat OpenShift Cluster in AWS, see Install Hitachi Storage Plug-in for Containers.

Install Kasten K10 in Red Hat OpenShift Cluster in AWS

To install Kasten K10 in Red Hat OpenShift Cluster in AWS, see Install Kasten K10 in Red Hat OpenShift Cluster.

Access Kasten K10 Dashboard

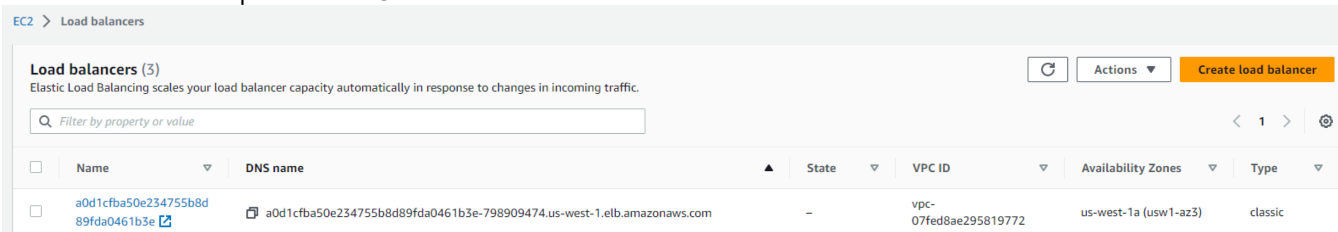
Kasten K10 dashboard URL is similar to that of near-cloud (http://127.0.0.1:8080/k10/#/), and is not externally published by default. To publish the K10 dashboard externally, run the following command:

```
#   helm upgrade --force k10 kasten/k10 --namespace=kasten-io \
>       --reuse-values \
>       --set externalGateway.create=true \
>       --set auth.tokenAuth.enabled=true
```

The command creates a service of type “LoadBalancer”.

```
[root@ip-10-77-28-159 sw_ocp18]# oc get svc gateway-ext --namespace kasten-io -o wide
NAME          TYPE           CLUSTER-IP    EXTERNAL-IP                                                               PORT(S)        AGE   SELECTOR
gateway-ext   LoadBalancer   172.30.77.1   a0d1cfba50e234755b8d89fda0461b3e-798909474.us-west-1.elb.amazonaws.com   80:32479/TCP   36d   service=gateway
[root@ip-10-77-28-159 sw_ocp18]#
```

LoadBalancer service provisions a Classic Load Balancer in AWS.



Install Kasten K10 Multi-Cluster Manager in Red Hat OpenShift Cluster in AWS

Download and untar the Kasten K10 Multi-Cluster tool from: https://github.com/kastenhq/external-tools/releases.

Context Configuration

The default name “kasten-io/api-awscluster-hvcloudconnect-com:6443/system:admin” may cause confusion to know which cluster we are managing, that is why changed the name to make it clearer.

1. List the available Kubernetes context.

```
# oc config get-contexts

CURRENT   NAME                                                            CLUSTER
AUTHINFO                                                    NAMESPACE
          admin                                                           awscluster
admin
*         kasten-io/api-awscluster-hvcloudconnect-com:6443/system:admin   api-awscluster-
hvcloudconnect-com:6443   system:admin/api-awscluster-hvcloudconnect-com:6443   kasten-io
```

2. Rename the kasten-io context as development.

```
# oc config rename-context kasten-io/api-awscluster-hvcloudconnect-com:6443/system:admin
development
Context "kasten-io/api-awscluster-hvcloudconnect-com:6443/system:admin" renamed to
"development".
```

After renaming:

```
# oc config get-contexts
CURRENT   NAME          CLUSTER                                 AUTHINFO
NAMESPACE
          admin         awscluster                              admin
*         development   api-awscluster-hvcloudconnect-com:6443  system:admin/api-
awscluster-hvcloudconnect-com:6443   kasten-io
```

3. Use the Kasten K10 Multi-Cluster tool to generate a modified kubeconfig file of Red Hat OpenShift Cluster in AWS (development). This is required during discovery of this cluster as secondary cluster from the near-cloud Kasten K10 Multi-Cluster instance.

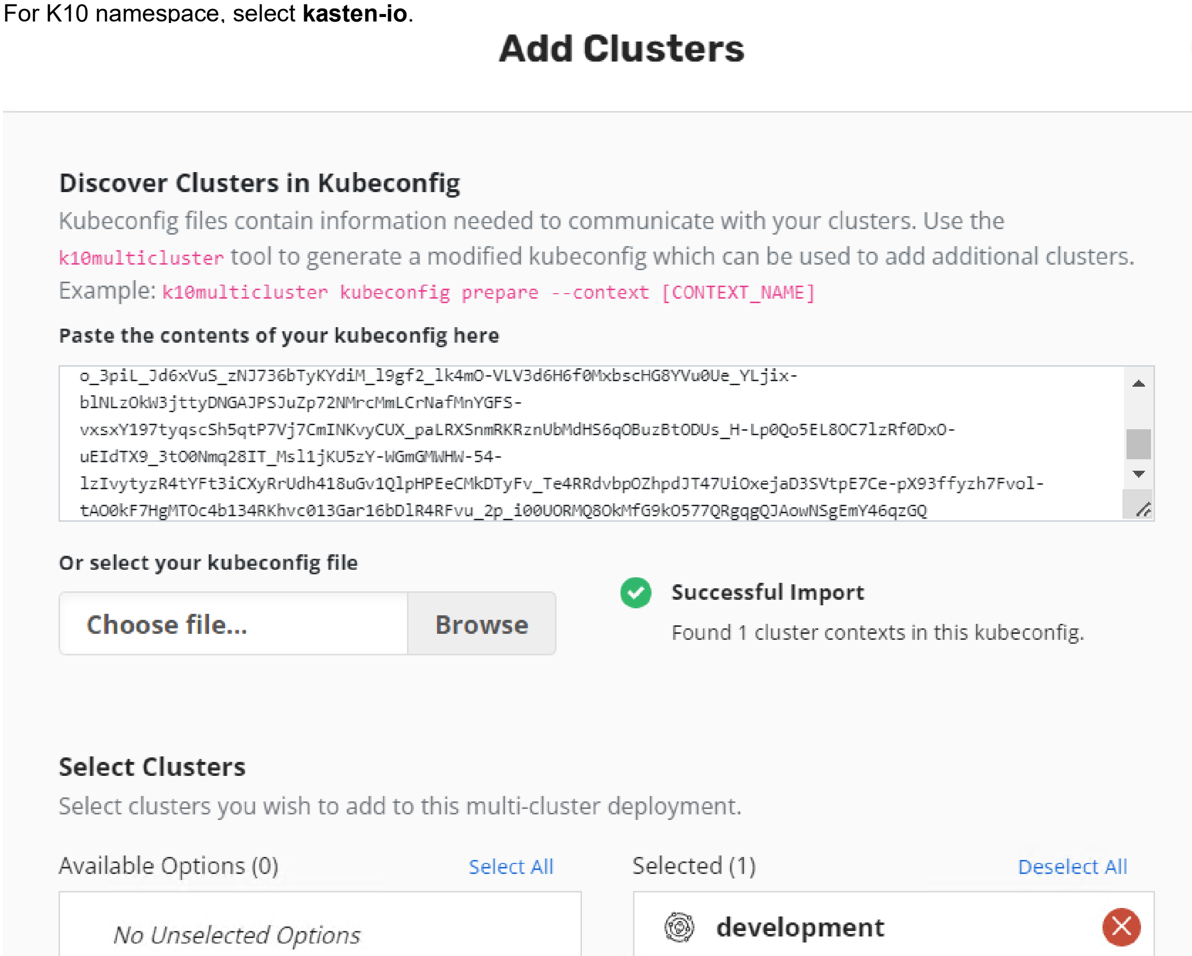
```
# k10multicluster kubeconfig prepare --context development
Preparing context development…
```

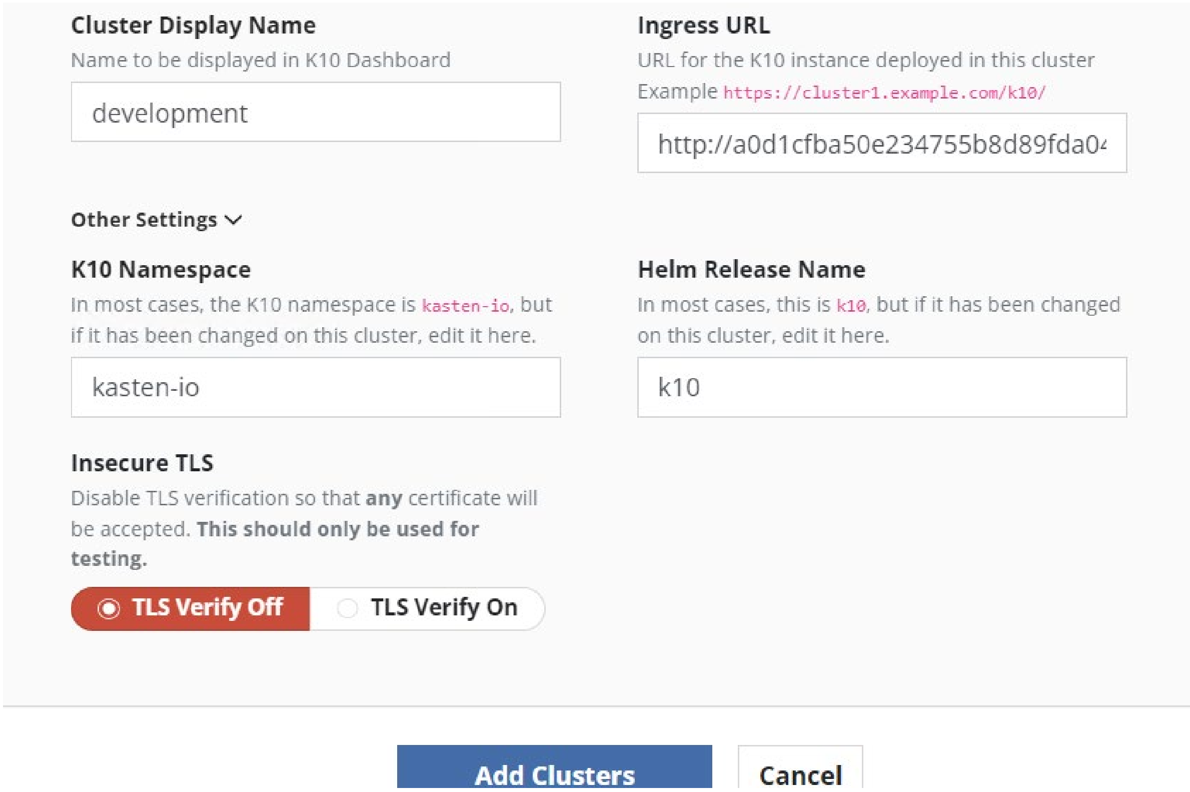
(Output is truncated)

Discover Development Cluster from Kasten K10 Multi-Cluster in Near-Cloud

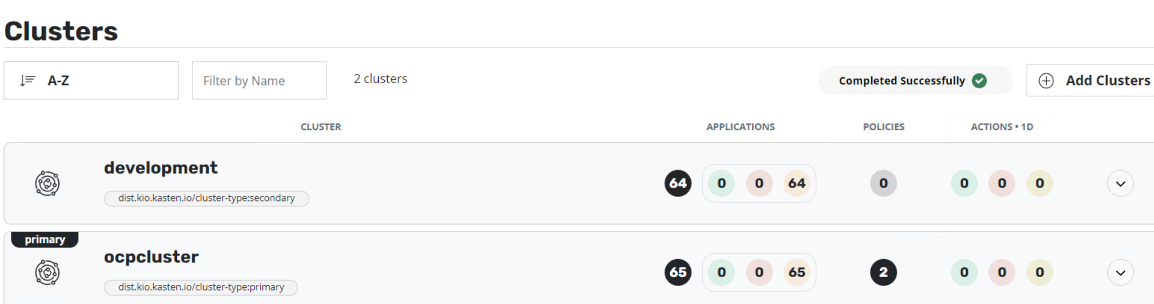
1. Add the secondary cluster (development) from near-cloud Kasten K10 UI by uploading kubeconfig file.
2. Navigate to the Kasten K10 Multi-Cluster dashboard and click **Add Clusters**.
3. In the **Add Clusters** window, enter the following information and click **Add Cluster**.
    a. Paste the content of kubeconfig of the development cluster generated using the k10multicluster tool.
    b. In the select cluster dialog box, select the development cluster from the list of available clusters.
    c. Provide Ingress URL of the K10 instance in AWS. For Ingress URL, see Access Kasten K10 Dashboard.

d. For K10 namespace, select **kasten-io**.





Status of the newly added cluster “development” in Kasten K10 dashboard:



Test 2: Deploy a Stateful Application in Red Hat OpenShift Cluster

This test case describes the process of deploying MySQL Stateful application in Red Hat OpenShift cluster in near-cloud as well as AWS using persistent volume from the VSP 5200 storage system located in near-cloud. HSPC plugin enables the application to use a persistent volume from the VSP 5200 storage system.

Deploy in Near-Cloud

1. Deploy a Stateful MySQL application using the mysqlsts.yaml manifest file.

   a. Create a namspace for the MySQL application.
   ```
   # oc create namespace productionmysql
   ```

   b. Create a mysqlsts.yaml manifest file for MySQL service and POD. For creating storage class sc-vsp5200, see Install Hitachi Storage Plug-in for Containers: Create Storage Class and Volume SnapshotClass.

```
apiVersion: v1
kind: Service
metadata:
    namespace: productionmysql
    name: mysqlapp
    labels:
       app: mysqlapp
spec:
    ports:
    - port: 3306
      name: mysqlapp
    clusterIP: None
    selector:
       app: mysqlapp
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
    namespace: productionmysql
    name: mysqlapp
spec:
  selector:
    matchLabels:
      app: mysqlapp
  serviceName: "mysqlapp"
  podManagementPolicy: Parallel
  replicas: 1
  template:
    metadata:
      labels:
        app: mysqlapp
    spec:
      terminationGracePeriodSeconds: 30
      containers:
      - name: mysqlapp
        image: mysql:5.7
        args:
          - "--ignore-db-dir=lost+found"
        env:
        - name: MYSQL_ROOT_PASSWORD
          value: pass123
        - name: MYSQL_DATABASE
          value: wordpress
        - name: MYSQL_USER
          value: admin
        - name: MYSQL_PASSWORD
          value: secret
        ports:
```

```
        - containerPort: 3306
          name: mysql
        volumeMounts:
        - name: mysql-vol
          mountPath: /var/lib/mysql
  volumeClaimTemplates:
  - metadata:
      name: mysql-vol
    spec:
      storageClassName: sc-vsp5200
      accessModes: [ "ReadWriteOnce" ]
      resources:
        requests:
          storage: 30Gi
```

c. To create MySQL service and POD using the YAML file, run the following oc command:

```
# oc create -f mysqlsts.yaml
```

d. Check the status of the MySQL service as follows:

```
# oc get svc -n productionmysql
NAME       TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
mysqlapp   ClusterIP   None         <none>        3306/TCP   25s
```

e. Verify whether the pod is created and the status is running.

```
# oc get pod -n productionmysql
NAME         READY   STATUS    RESTARTS   AGE
mysqlapp-0   1/1     Running   0          38s
```

f. Verify whether pvc is created from the VSP 5200 storage system as per the manifest file. Using storage class dynamically provisions a persistent volume in the VSP 5200 storage system.

```
# oc get pvc -n productionmysql
NAME                   STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
mysql-vol-mysqlapp-0   Bound    pvc-a3864ef6-23cf-4e39-889f-9829779762eb   30Gi
RWO            sc-vsp5200     45s

# oc get pv |grep productionmysql
pvc-a3864ef6-23cf-4e39-889f-9829779762eb   30Gi       RWO            Delete
Bound    productionmysql/mysql-vol-mysqlapp-0                      sc-vsp5200
```

2. Access the MySQL application.
   a. Log in to pod MySQL and verify whether the 30 GB persistent volume is created and mounted in /var/lib/mysql.

```
[root@linuxnfscl2 sw_k10mc]# oc -n productionmysql rsh mysqlapp-0
sh-4.2$ df -h
Filesystem      Size  Used Avail Use% Mounted on
overlay         120G   19G  102G  16% /
tmpfs            64M     0   64M   0% /dev
tmpfs           7.8G     0  7.8G   0% /sys/fs/cgroup
shm              64M     0   64M   0% /dev/shm
tmpfs           7.8G   47M  7.8G   1% /etc/passwd
/dev/sda4       120G   19G  102G  16% /etc/hosts
/dev/sdf         30G  255M   28G   1% /var/lib/mysql
```

   b. Log in to MySQL database using “***mysql -u root -p***”.
   c. Verify whether the “wordpress” database is created.

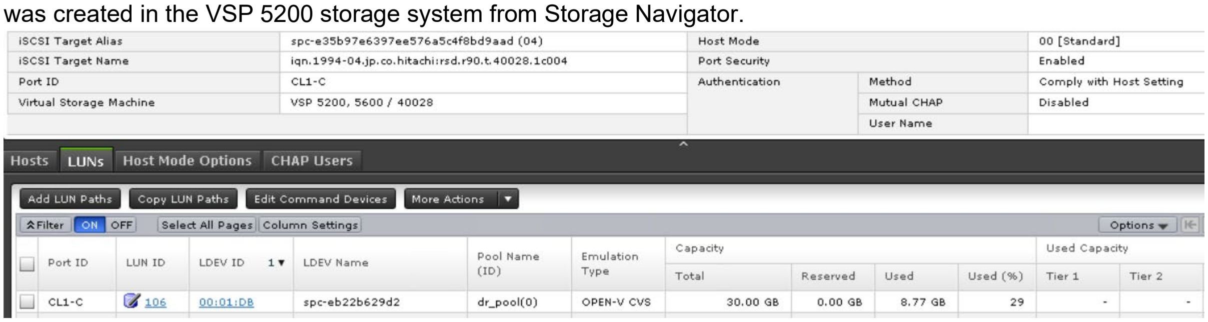
d. Select the “wordpress” database.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| wordpress          |
+--------------------+
5 rows in set (0.00 sec)

mysql> use wordpress;
Database changed
```

e. Create a table “employee” and ingest some records to the table.

```
mysql> SELECT * FROM employees;
+--------+------------+------------+-----------+--------+--------------+
| emp_no | birth_date | first_name | last_name | gender | joining_date |
+--------+------------+------------+-----------+--------+--------------+
|  10001 | 1988-12-03 | Ajay       | Kumar     | M      | 2018-07-18   |
|  10002 | 1989-12-03 | Amit       | Kumar     | M      | 2018-09-18   |
|  10003 | 1985-12-06 | Robert     | Callahan  | M      | 2018-07-18   |
|  10004 | 1985-12-06 | Anne       | Buchanan  | F      | 2018-07-18   |
|  10005 | 1989-12-19 | Ravi       | Reddy     | M      | 2018-07-18   |
|  10006 | 1990-12-06 | Carlos     | Fuller    | M      | 2018-07-18   |
|  10007 | 1980-12-06 | Satish     | J         | M      | 2018-07-18   |
|  10008 | 1989-08-19 | Raj        | Singh     | M      | 2018-07-18   |
|  10009 | 1990-06-06 | Andrew     | Muller    | M      | 2018-07-18   |
|  10010 | 1980-11-06 | Rabin      | RD        | M      | 2018-07-18   |
+--------+------------+------------+-----------+--------+--------------+
10 rows in set (0.01 sec)

mysql>
```

f. HSPC automatically creates an iSCSI target on port 1-C of the storage system. Verify whether the 30 GB volume was created in the VSP 5200 storage system from Storage Navigator.



Deploy in AWS

1. Deploy a stateful MySQL application in “devclustermysql” namespace with a 200 GB persistent volume.
2. Create a manifest file and deploy the application. See Test 2: Deploy a Stateful Application in Red Hat OpenShift Cluster: Deploy in Near-Cloud.
3. Access the stateful MySQL application.
   a. Verify whether the pod is running.

```
# oc get pod -n devclustermysql
NAME                READY    STATUS     RESTARTS    AGE
```

```
mysqlappdev-0   1/1     Running   0          47s
```

b. Log in to pod MySQL and verify that the 200 GB persistent volume is mounted in “/var/lib/mysql”.

```
[root@ip-10-77-28-159 sw_ocp18]# oc -n devclustermysql rsh mysqlappdev-0
sh-4.2$ df -h
Filesystem      Size  Used Avail Use% Mounted on
overlay         200G   31G  169G  16% /
tmpfs            64M     0   64M   0% /dev
tmpfs           7.8G     0  7.8G   0% /sys/fs/cgroup
shm              64M     0   64M   0% /dev/shm
tmpfs           7.8G   52M  7.7G   1% /etc/passwd
/dev/nvme0n1p4  200G   31G  169G  16% /etc/hosts
/dev/sdi        197G  271M  187G   1% /var/lib/mysql
```

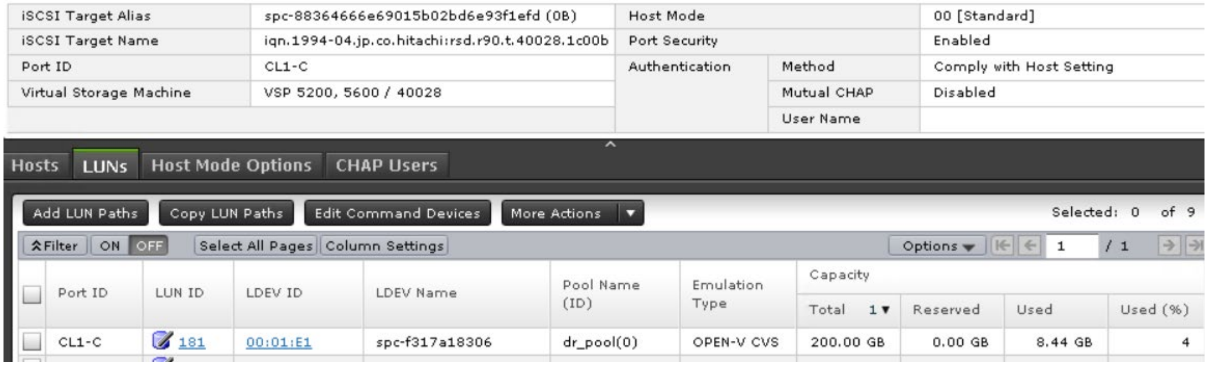
c. Log in to MySQL database using “***mysql -u root -p***”.
d. Verify that “persistantdb” database is created as mentioned in the manifest file.
e. Select the “persistantdb” database.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| persistantdb       |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql> use persistantdb;
Database changed
```

f. Create a table “student” and ingest some new records to the table.

```
mysql> SELECT * FROM student;
+-----------------+------------+------------+-----------+--------+---------------+
| registration_no | birth_date | first_name | last_name | gender | admision_date |
+-----------------+------------+------------+-----------+--------+---------------+
|           70001 | 1988-12-03 | Ramesh     | Kumar     | M      | 2018-07-18    |
|           70002 | 1989-12-03 | Akshya     | Kumar     | M      | 2018-09-18    |
|           70003 | 1985-12-06 | Robert     | Callahan  | M      | 2018-07-18    |
|           70004 | 1985-12-06 | Anne       | Buchanan  | F      | 2018-07-18    |
|           70005 | 1989-12-19 | David      | Hussain   | M      | 2018-07-18    |
|           70006 | 1990-12-06 | Ananda     | Muller    | M      | 2018-07-18    |
|           70007 | 1980-12-06 |            | J         | M      | 2018-07-18    |
|           70008 | 1989-08-19 | RN         | Prasad    | M      | 2018-07-18    |
|           70009 | 1990-06-06 | Ad         | Bolt      | M      | 2018-07-18    |
|           70010 | 1980-11-06 | Rabin      | RD        | M      | 2018-07-18    |
|           70011 | 1985-10-02 | Raj        | MOHAN     | M      | 2019-07-18    |
|           70012 | 1985-10-02 | Priayam    | Ajad      | M      | 2019-07-18    |
+-----------------+------------+------------+-----------+--------+---------------+
12 rows in set (0.00 sec)

mysql>
```

g. HSPC automatically creates an iSCSI target on port 1-C of the storage system. Verify whether the 200 GB volume was created in the VSP 5200 storage system from Storage Navigator.



Test 3: Migrate Stateful Applications Across OpenShift Clusters Using Kasten K10 Multi-Cluster

This test case describes the process of migrating a stateful application by performing backup and restoration operations between two OpenShift clusters using Kasten K10 Multi-Cluster Global policy and HSPC. To illustrate this, we captured the snapshot of a stateful MySQL application running on Red Hat OpenShift cluster in near-cloud and then restored it on a secondary cluster in AWS. The entire process is performed from Kasten K10 Multi-Cluster UI. The VSP 5200 storage system serves the persistent volume required for stateful MySQL applications in both clusters.

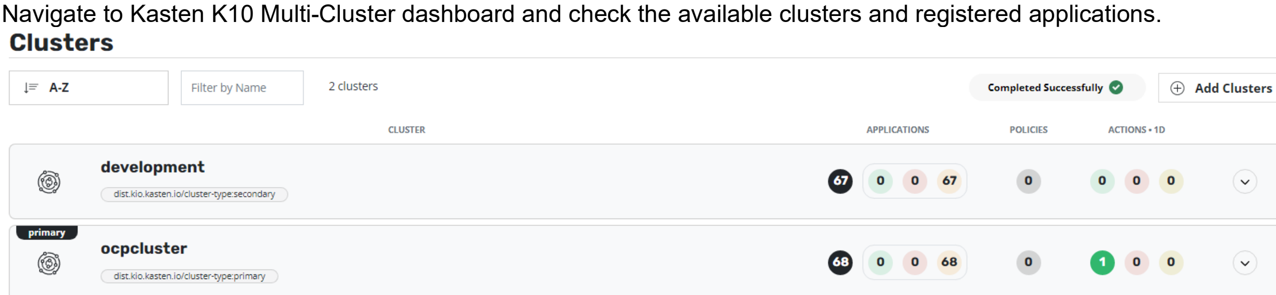
Snapshot Operation

Before performing snapshot operation of an application, create the following:

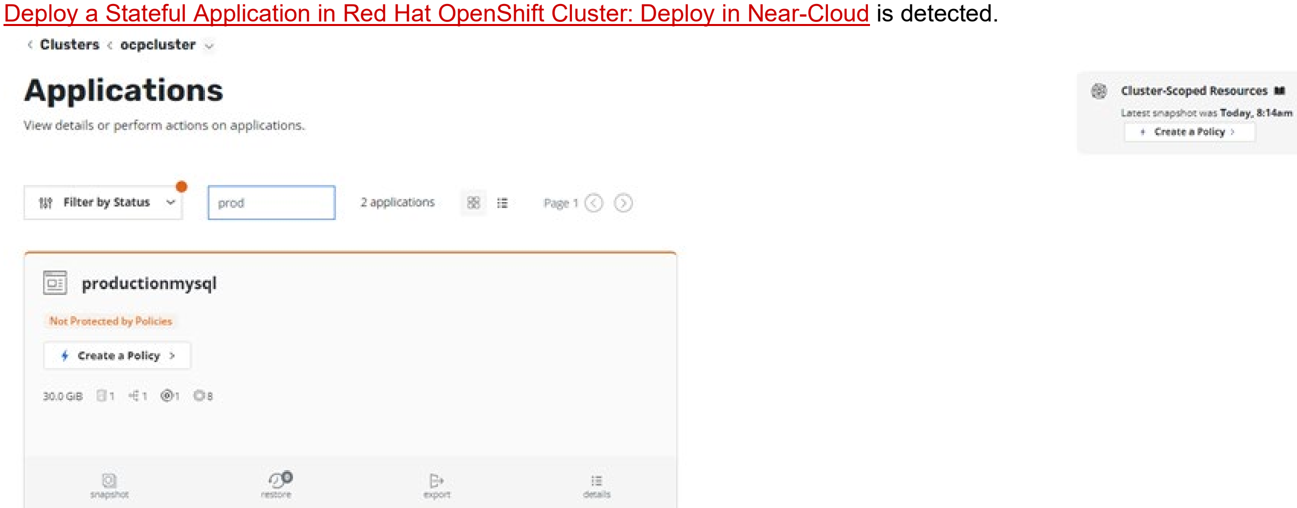
- Global Location Profile: Profiles define credentials and locations required to move the data in and out of the cluster. In this scenario, an Amazon S3 bucket is used.
- Global Policy: Policies are used to automate your data management workflows. To achieve this, they combine actions you want to take (such as snapshot), a frequency or schedule for how often you want to take that action, and a label-based selection criteria for the resources you want to manage.
- Distribution: Distributions define which K10 resources belong to which clusters.

To perform snapshot operation of an application using Kasten K10, complete the following steps:

1. Verify that the application created in near-cloud as shown in Test 2: Deploy a Stateful Application in Red Hat OpenShift Cluster: Deploy in Near-Cloud is registered in the Kasten K10 UI.
2. Navigate to Kasten K10 Multi-Cluster dashboard and check the available clusters and registered applications.



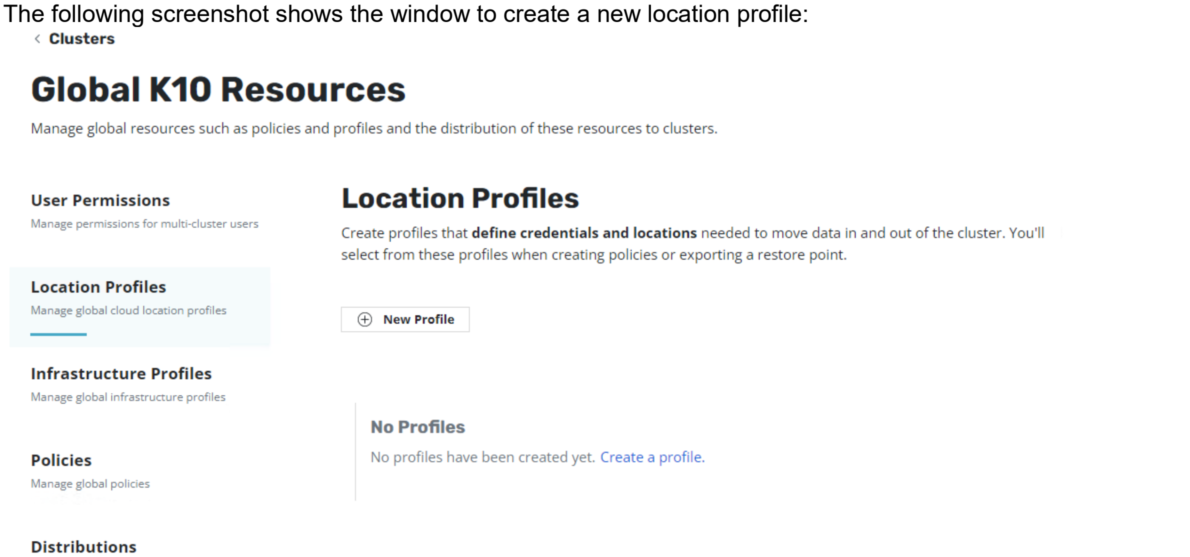
3. Navigate to Clusters, select **ocpcluster** (primary), and view the registered applications. Kasten K10 registers detected namespaces as an application. The following screenshot shows that the “productionmysql" namespace created in Test 2: Deploy a Stateful Application in Red Hat OpenShift Cluster: Deploy in Near-Cloud is detected.



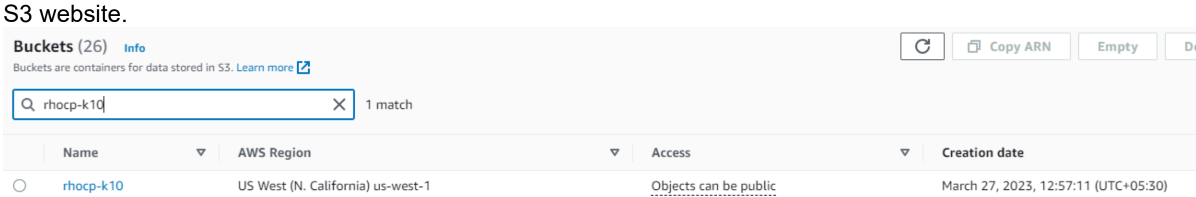
4. Create a Global Cloud Location Profile.

a. In the K10 Global Resources section of the K10 Multi-Cluster dashboard, click **Global Profile**.

The following screenshot shows the window to create a new location profile:

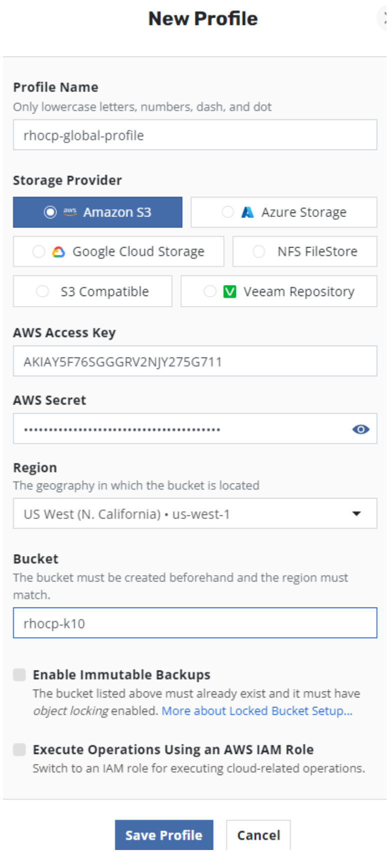


In this scenario, an Amazon AWS S3 bucket named "**rhocp-k10**" is created and designated as the destination for Global Location Profile. For instructions to create an S3 bucket, see the User Guide section on the Amazon AWS S3 website.

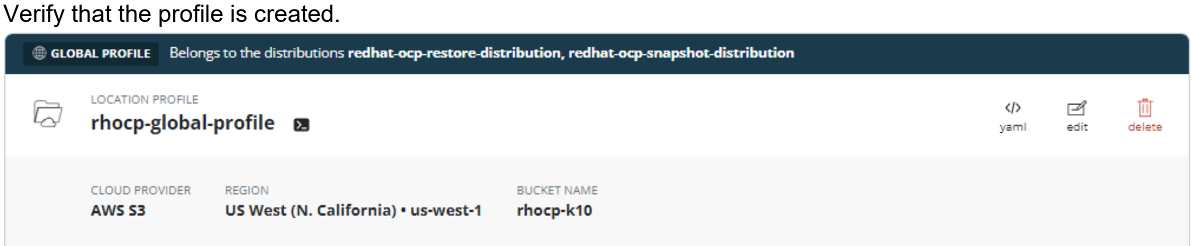


b. In Global K10 Resources window, click **New Profile**.

c. Enter the required information (such as Profile Name, Storage Provider, AWS region, Bucket Name, AWS Access Key, Secret Key, and so on) and then click **Save Profile**.



d. Verify that the profile is created.

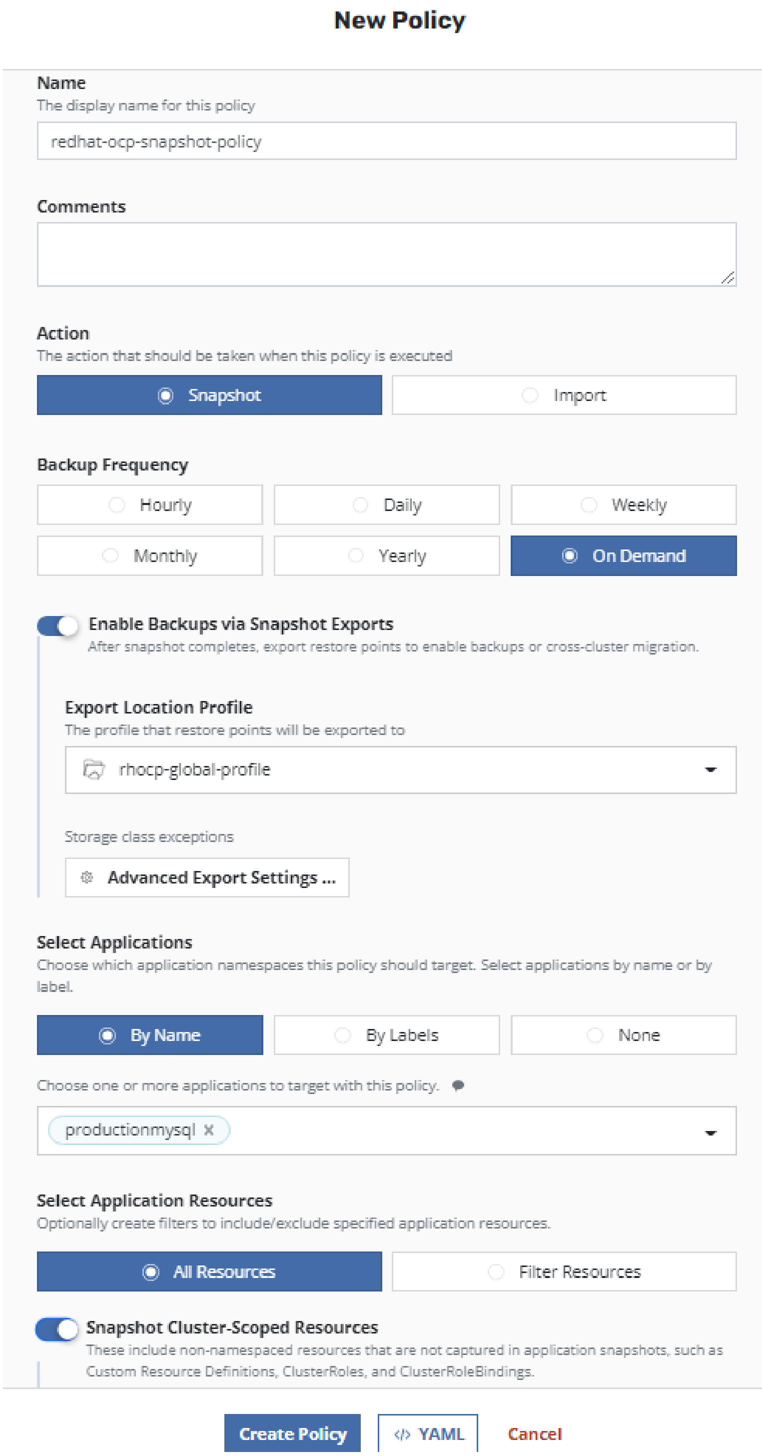


5. Create a Global Snapshot Policy.

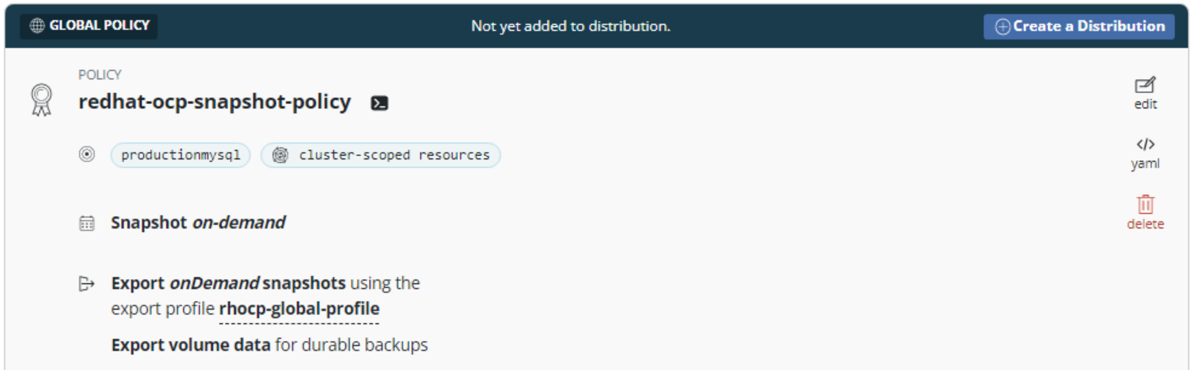
a. From the K10 Global Resources page, click **Global Polices** and then click **New Policy**.

b. Enter the snapshot related information (such as Policy Name, Backup Frequency, target application, application resources, and so on).

c. Select **Enable Backup via Snapshot Exports**, select the location profile that you created, and click **Create Policy**. This is required to generate an import policy while restoring the application.



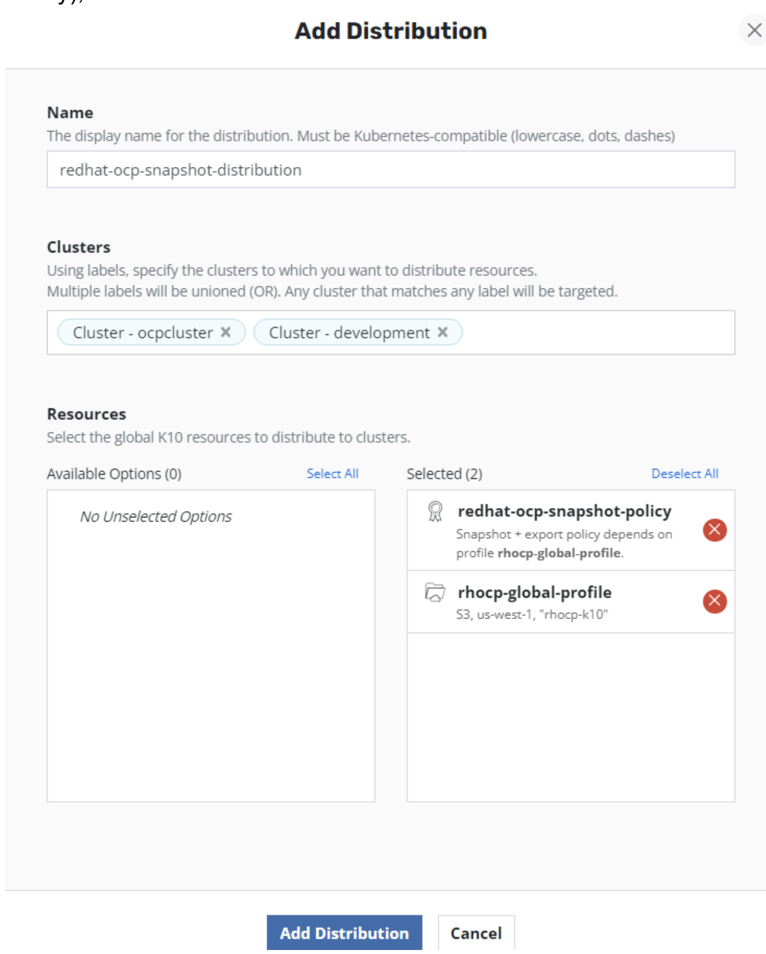
d. Verify that the policy is created.



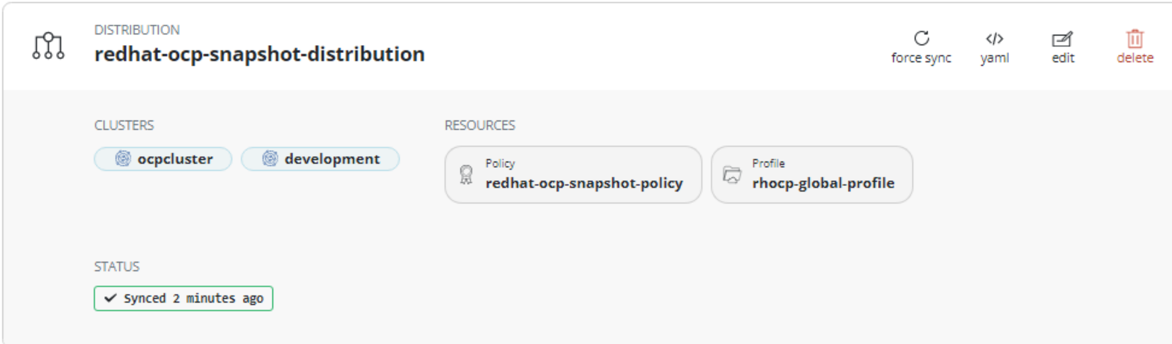
6. Create a distribution.

   a. From the K10 Global Resources page, click **Distributions** and then click **New Distribution**.

   b. In the Add Distribution window, enter the required information (such as Distribution Name, specify both near-cloud and AWS clusters), specify the two resources you created (Global Location Profile and Global Snapshot Policy), and then click **Add Distribution**.



c. Verify that the distribution is added.

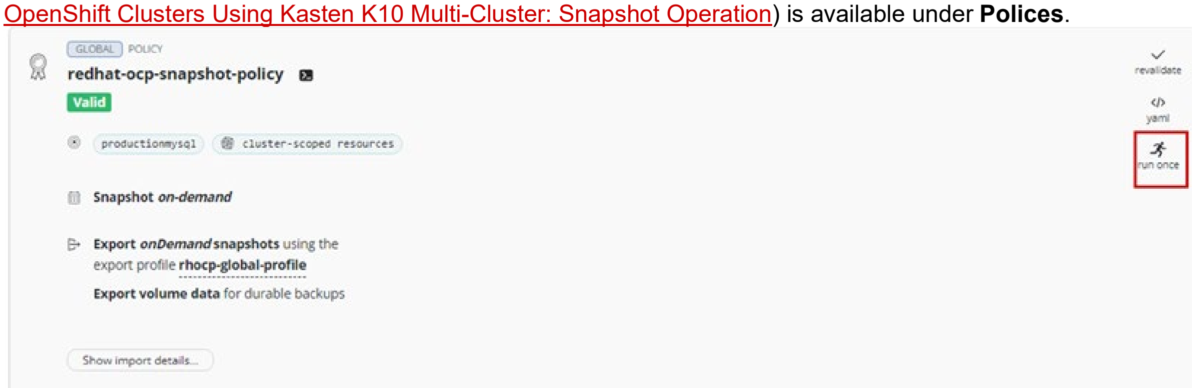


d. Navigate to the Kasten K10 dashboard and verify that the global policy is distributed to both clusters.

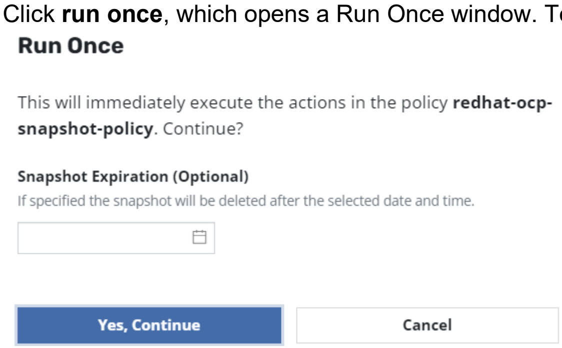
7. Collect a snapshot of the registered application using the Global Snapshot Policy.

a. From the K10 Multi-Cluster dashboard, click Cluster "**ocpcluster**" and then click **Policies**.

b. Verify that the Global snapshot on-demand policy (created in Test 3: Migrate Stateful Applications Across OpenShift Clusters Using Kasten K10 Multi-Cluster: Snapshot Operation) is available under **Polices**.

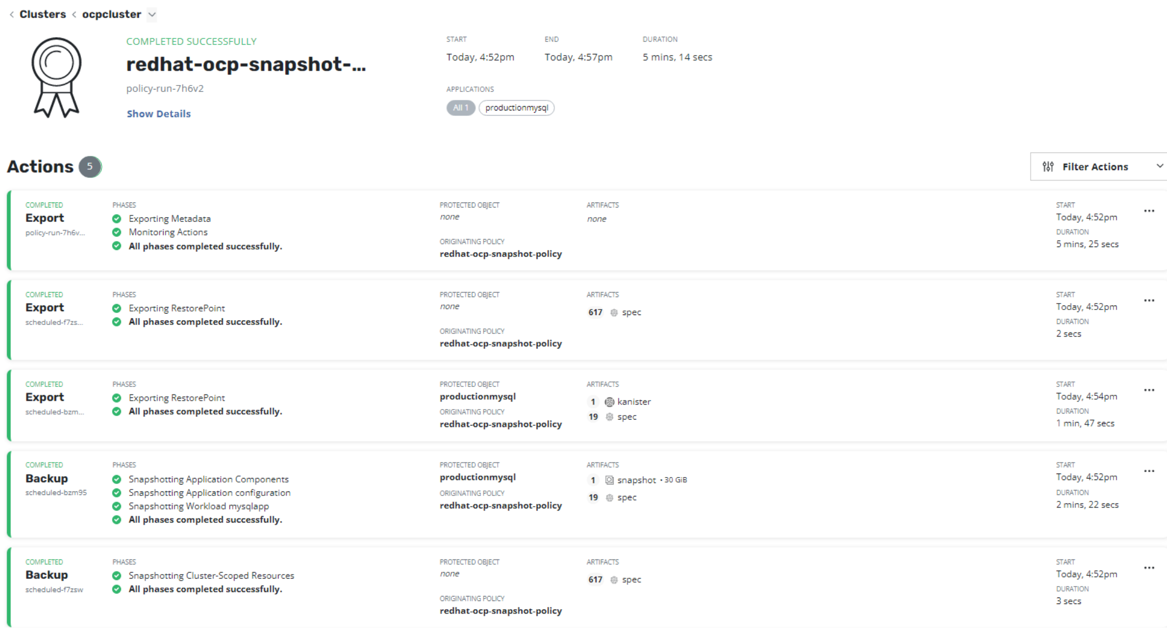


c. Click **run once**, which opens a Run Once window. To start the snapshot, click **Yes, Continue**.



d. Open the Kasten K10 Multi-Cluster dashboard and check the status of the policy in the [**Actions**] window. To check the phase in progress, click the related action.

e. Verify that the phase has changed to Completed Successfully. Click the relevant action to confirm that no error is present.



f. Integrating Kasten K10 with HSPC creates a Thin Image snapshot and splits the pair. In Storage Navigator, confirm the pair's status.

Date and Time	Primary Volume		Secondary Volume		Mirror Unit	Pool ID	Diff Compare Volume	Description Code	Description
	LDEV ID	Provisioning Type	LDEV ID	Provisioning Type					
2023/03/27 16:56:10	00:01:DB	DP	00:01:DC	DP	3	0	-	2011	PSUS
2023/03/27 16:56:07	00:01:DB	DP	00:01:DC	DP	3	0	-	2001	PAIR

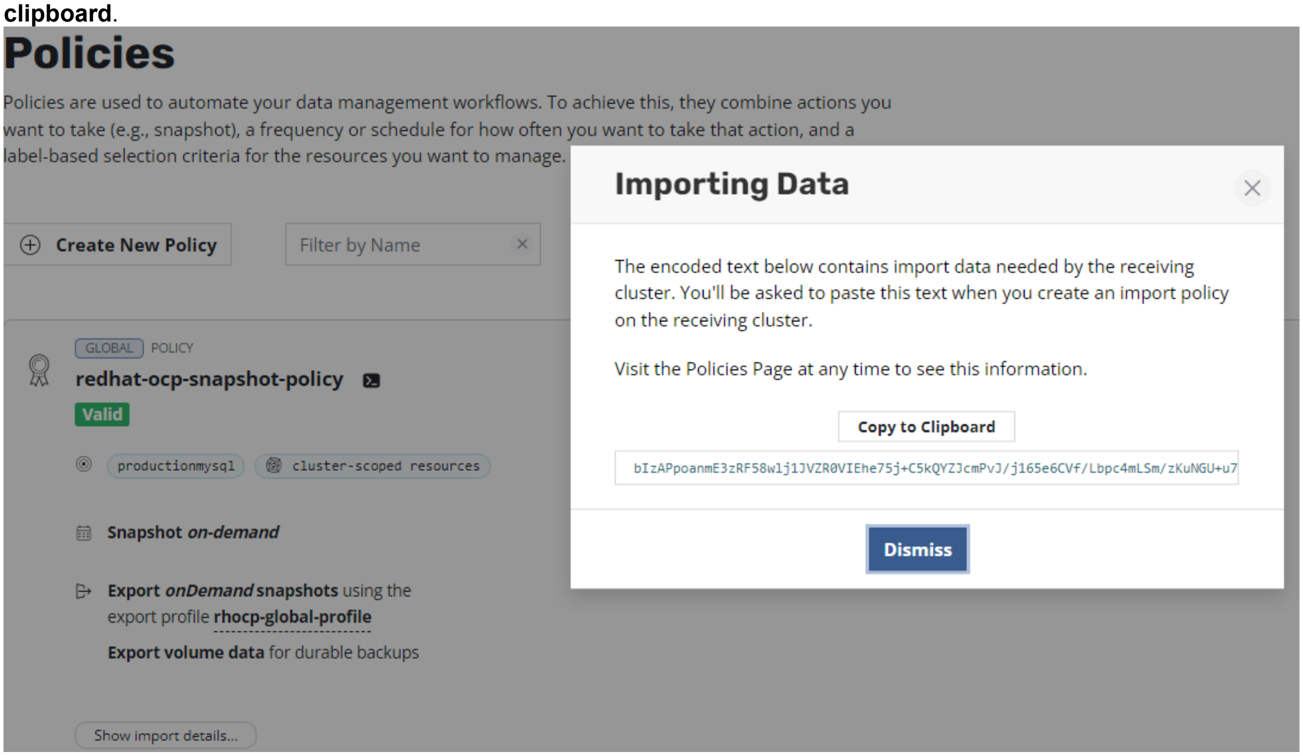
Restore Operation

You can restore an application from snapshot across clusters from Kasten K10 Multi-Cluster. Restoring operation consists of the following high-level steps:

- Copying the Import data.
- Creating a Restore policy.
- Apply the Restore Policy.

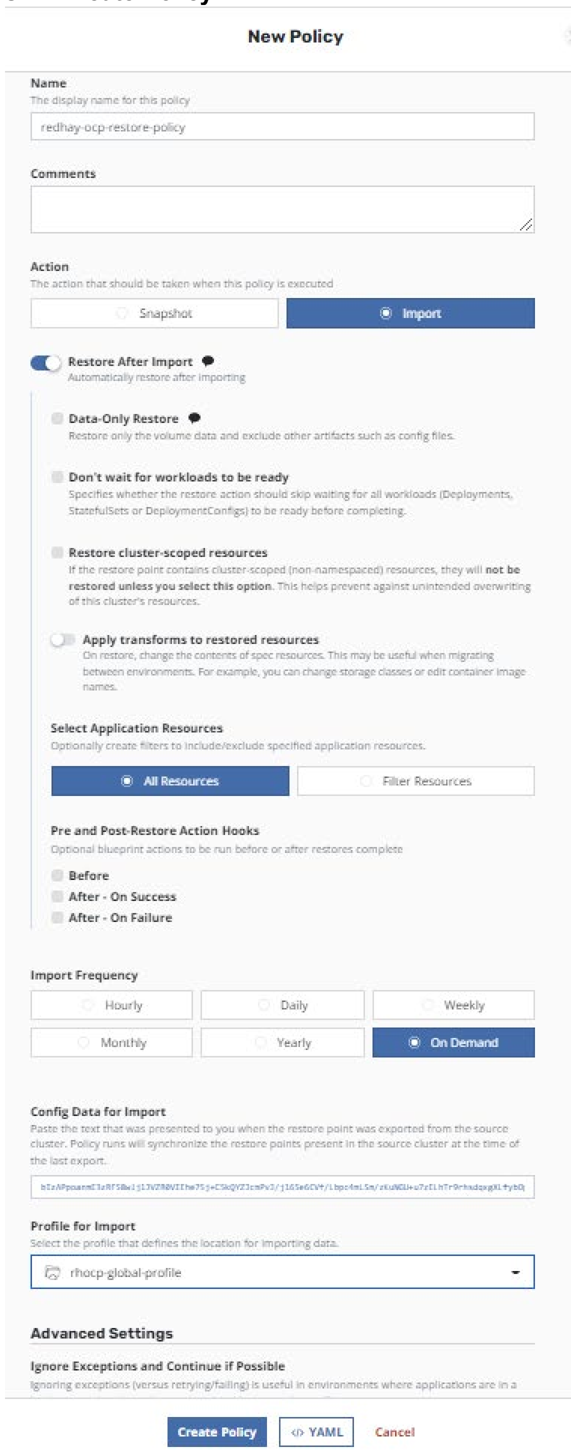
1. To copy the Import data from the K10 Multi-Cluster dashboard, click the production Cluster "**ocpcluster**" and then click **Policies**.

2. In the Polices window, select the Global Snapshot Policy created earlier, click **Show Import data**, and then click **Copy to clipboard**.

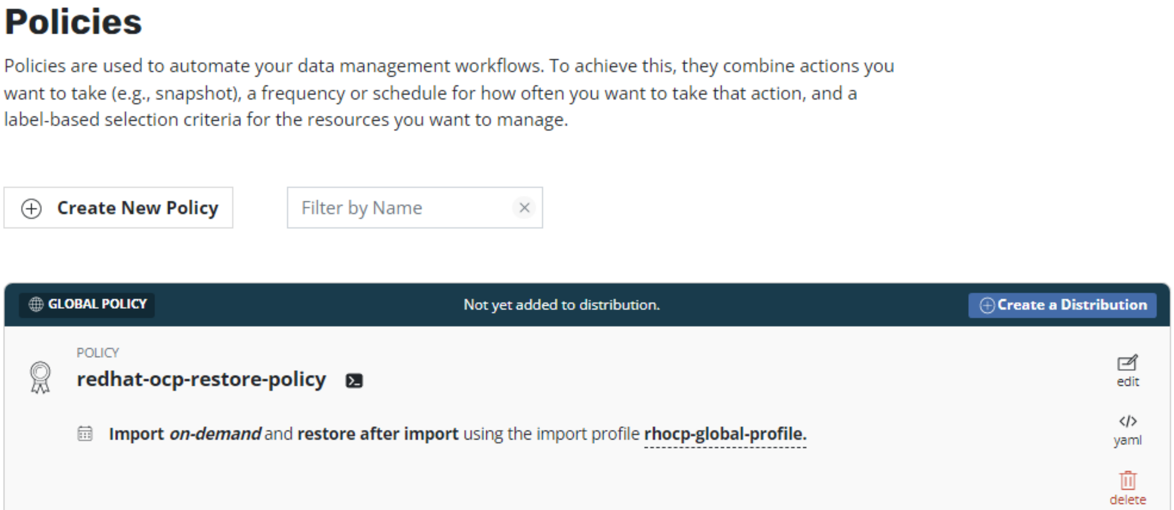


3. Create a restore policy.
    a. From the K10 Global Resources page, click **Polices** and then click **New Policy**.
    b. In the New Policy window, enter a Policy Name, and select Import Frequency as **On Demand**.
    c. In Config Data for Import section, paste the import policy copied in step 2.
    d. Select **Restore after Import** and select the Global Location in Profile for Import.

e. Click **Create Policy**.



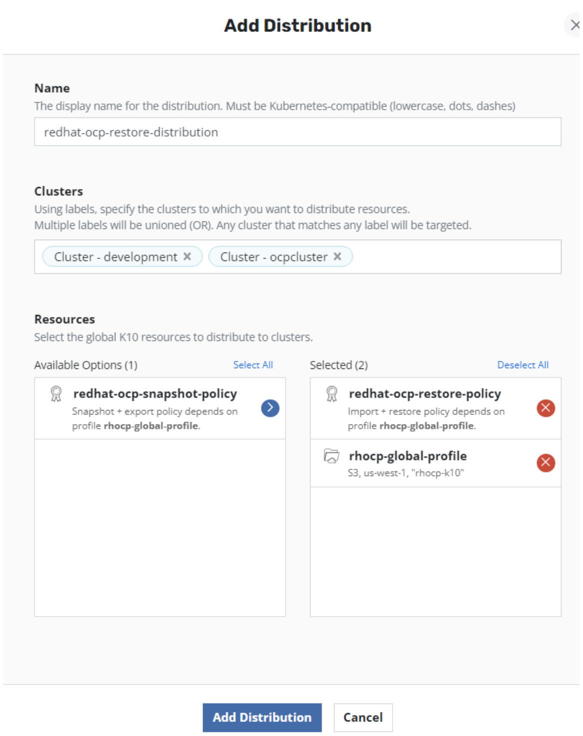
f. Verify that the policy is created.



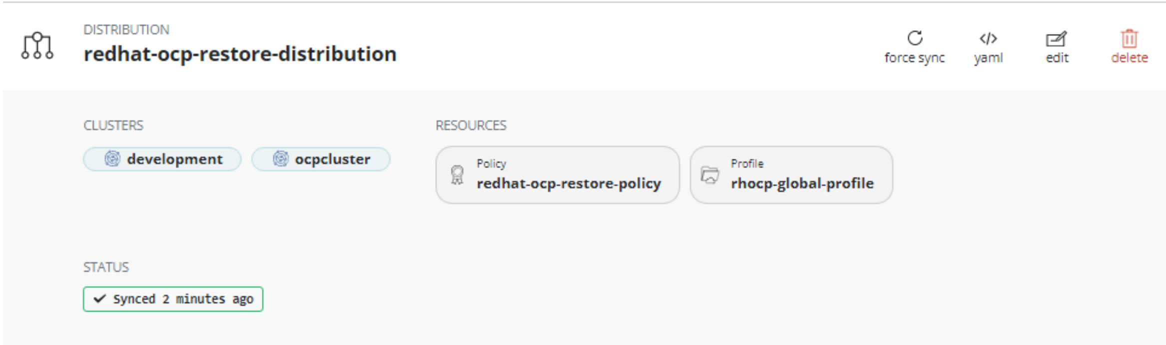
4. Create a Distribution.

   a. From the K10 Global Resources page, click **Distributions** and then click **New Distribution**.

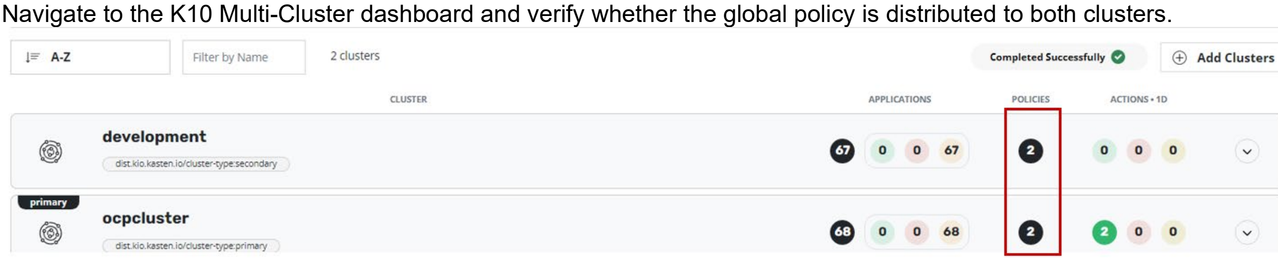
   b. In the Add Distribution window, enter the name, specify both clusters, select the restore policy and location profile in Resources, and then click **Add Distribution**.



c. Verify that the distribution is added.



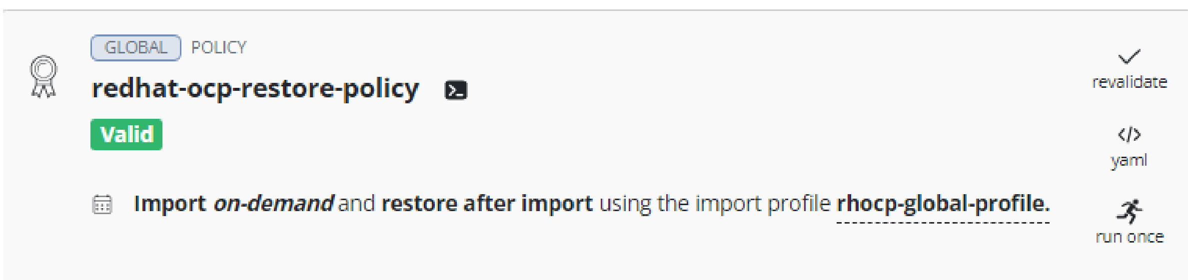
5. Navigate to the K10 Multi-Cluster dashboard and verify whether the global policy is distributed to both clusters.



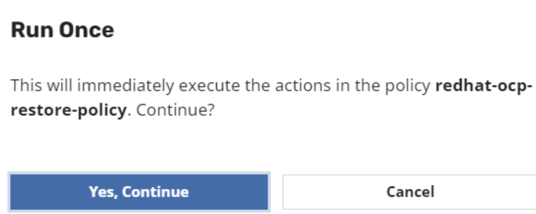
6. Run the restore operation.

a. From the K10 Multi-Cluster dashboard, click **development** (the secondary cluster) and then click **Policies**.

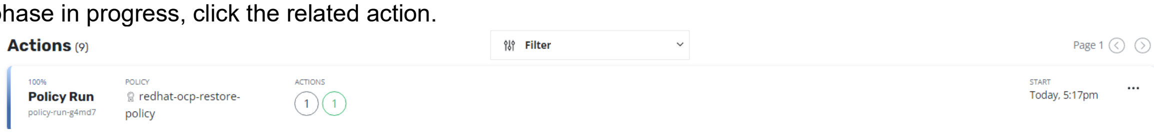
b. Verify whether the Global restore on-demand policy created in Test 3: Migrate Stateful Applications Across OpenShift Clusters Using Kasten K10 Multi-Cluster: Restore Operation section is available under Polices.



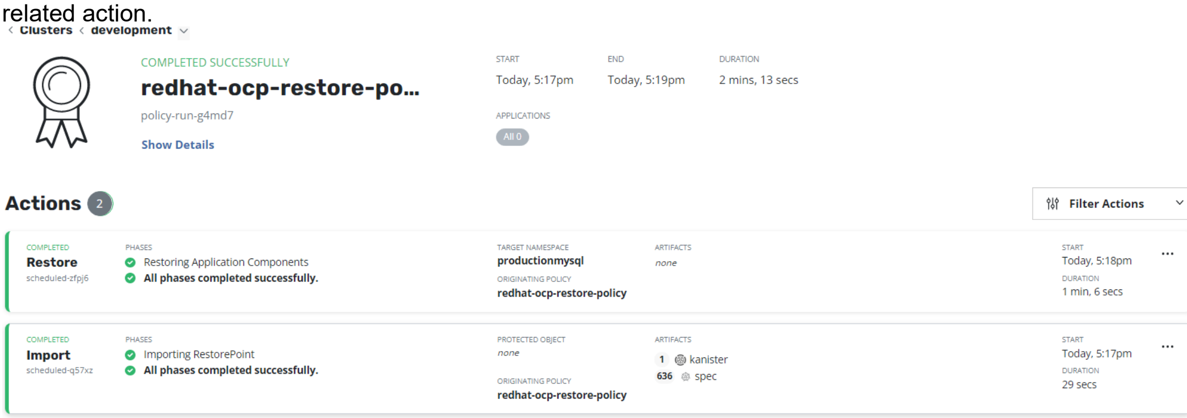
c. Click **run once**, which opens the Run Once window. To start the restore, click **Yes, Continue**.



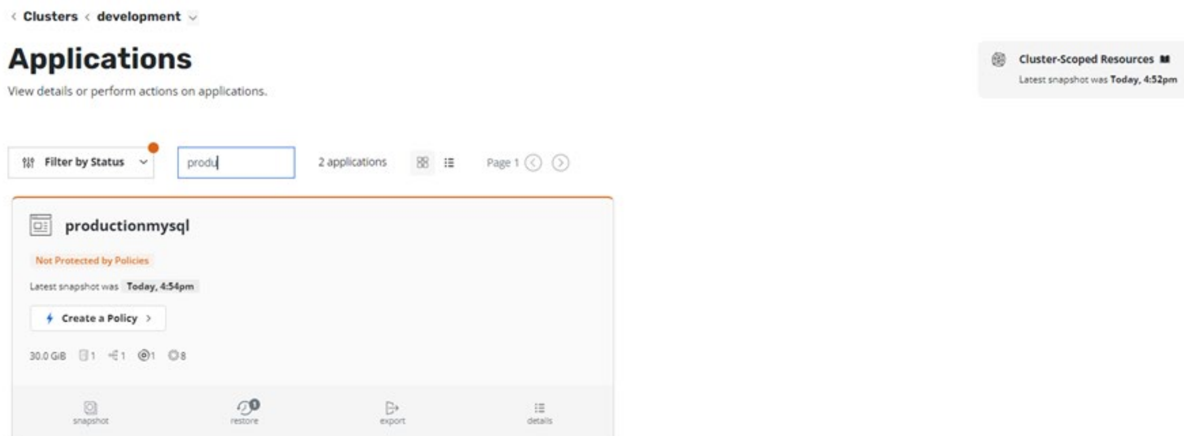
d. Open the K10 Multi-Cluster dashboard and check the status of the policy in the [Actions] window. To check the phase in progress, click the related action.



e. Verify that the phase has changed to Completed Successfully. To confirm that no error is present, click the related action.



f. From the K10 Multi-Cluster dashboard, navigate to cluster Development and verify that application "**productionmysql**" is restored.



7. Verify that the data is available.

a. Log in to the pod mysqlapp-0 in productionmysql namespace in the development cluster and verify whether the 30 GB persistent volume is mounted.

```
[root@ip-10-77-28-159 sw_k10]# oc get pod -n productionmysql
NAME          READY    STATUS     RESTARTS    AGE
mysqlapp-0    1/1      Running    0           10m
```

```
[root@ip-10-77-28-159 sw_k10]# oc -n productionmysql rsh mysqlapp-0
sh-4.2$ df -h
Filesystem      Size  Used Avail Use% Mounted on
overlay         200G   22G  178G  11% /
tmpfs            64M     0   64M   0% /dev
tmpfs           7.8G     0  7.8G   0% /sys/fs/cgroup
shm              64M     0   64M   0% /dev/shm
tmpfs           7.8G   49M  7.7G   1% /etc/passwd
/dev/nvme0n1p4  200G   22G  178G  11% /etc/hosts
/dev/sdc         30G  255M   28G   1% /var/lib/mysql
```

b. Log in to MySQL and verify whether database wordpress and employee table is available.

```
mysql> use wordpress;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql>
mysql> show tables;
+----------------------+
| Tables_in_wordpress |
+----------------------+
| employees            |
+----------------------+
1 row in set (0.00 sec)
```

c. Verify whether the ingested data in Primary cluster "ocpcluster" (as shown in Test 2: Deploy a Stateful Application in Red Hat OpenShift Cluster: Deploy in Near-Cloud) is available in "development" cluster in AWS.

```
mysql> SELECT * FROM employees;
+--------+------------+------------+-----------+--------+--------------+
| emp_no | birth_date | first_name | last_name | gender | joining_date |
+--------+------------+------------+-----------+--------+--------------+
|  10001 | 1988-12-03 | Ajay       | Kumar     | M      | 2018-07-18   |
|  10002 | 1989-12-03 | Amit       | Kumar     | M      | 2018-09-18   |
|  10003 | 1985-12-06 | Robert     | Callahan  | M      | 2018-07-18   |
|  10004 | 1985-12-06 | Anne       | Buchanan  | F      | 2018-07-18   |
|  10005 | 1989-12-19 | Ravi       | Reddy     | M      | 2018-07-18   |
|  10006 | 1990-12-06 | Carlos     | Fuller    | M      | 2018-07-18   |
|  10007 | 1980-12-06 | Satish     | J         | M      | 2018-07-18   |
|  10008 | 1989-08-19 | Raj        | Singh     | M      | 2018-07-18   |
|  10009 | 1990-06-06 | Andrew     | Muller    | M      | 2018-07-18   |
|  10010 | 1980-11-06 | Rabin      | RD        | M      | 2018-07-18   |
+--------+------------+------------+-----------+--------+--------------+
10 rows in set (0.00 sec)
```

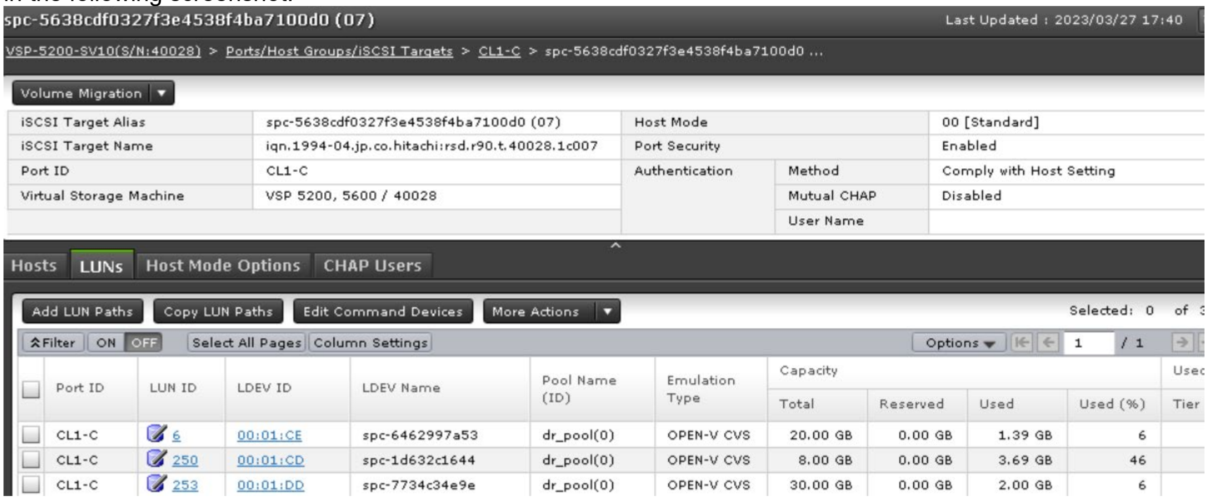
During restoration a clone volume "00:01:DD" was created from the snapshot volume "00:01:DC". The following screenshot shows the creation of the clone volume:

TI History (Page.1)

Filter ON OFF

Date and Time	Primary Volume		Secondary Volume		Mirror Unit	Pool ID	Diff Compare Volume	Description Code	Description
	LDEV ID	Provisioning Type	LDEV ID	Provisioning Type					
2023/03/27 16:57:51	00:01:DC	DP	00:01:DD	DP	3	0	-	2092	CLONE END
2023/03/27 16:56:56	00:01:DC	DP	00:01:DD	DP	3	0	-	2091	CLONE START
2023/03/27 16:56:56	00:01:DC	DP	00:01:DD	DP	3	0	-	2001	PAIR

Clone volume "00:01:DD" was assigned to the restored application in the development cluster in AWS, as shown in the following screenshot:



Test 4: Migrate a Stateful Application Across OpenShift Cluster Manually

Test 3 describes the applications migration process with Kasten K10 Multi-Cluster Manager. This test case, instead of using Kasten K10, describes the Kubernetes commands with the help of HSPC plugin that can be used for migrating a stateful application from OpenShift cluster in near-cloud to AWS. The VSP 5200 storage system provides the persistent volume required for stateful MySQL application in both clusters.

Snapshot Operation

Complete the following steps in OpenShift Cluster in near-cloud:

1. Create a new namespace “prodmysql” and deploy a Stateful MySQL application with a 250 GB persistent volume from the VSP 5200 storage system, as shown in Test 2: Deploy a Stateful Application in Red Hat OpenShift Cluster: Deploy in Near-Cloud.
2. Access the stateful MySQL application.
    a. Log in to pod MySQL and verify that the 250 GB persistent volume is mounted in “/var/lib/mysql”, as per the manifest file.

```
[root@linuxnfscl2 ~]# oc -n prodmysql rsh prodmysqlapp-0
sh-4.2$ df -h
Filesystem      Size  Used Avail Use% Mounted on
overlay         120G   20G  100G  17% /
tmpfs            64M     0   64M   0% /dev
tmpfs           7.8G     0  7.8G   0% /sys/fs/cgroup
shm              64M     0   64M   0% /dev/shm
tmpfs           7.8G   48M  7.8G   1% /etc/passwd
/dev/sda4       120G   20G  100G  17% /etc/hosts
/dev/sdj        246G  271M  234G   1% /var/lib/mysql
```

    b. Log in to MySQL database using “***mysql -u root -p***” and verify that database “prodmysqldb” is created as per the manifest file.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| prodmysqldb        |
| sys                |
+--------------------+
5 rows in set (0.01 sec)
```

c. Create a table “employees” and ingest some new records to the table.

```
mysql> SELECT * FROM employees;
+--------+------------+------------+-----------+--------+--------------+
| emp_no | birth_date | first_name | last_name | gender | joining_date |
+--------+------------+------------+-----------+--------+--------------+
|  50001 | 1988-12-03 | Ramesh     | Kumar     | M      | 2018-07-18   |
|  50002 | 1989-12-03 | Amit       | Kumar     | M      | 2018-09-18   |
|  50003 | 1985-12-06 | Robert     | Fernandez | M      | 2018-07-18   |
|  50004 | 1985-12-06 | Md         | Riaz      | M      | 2018-07-18   |
|  50005 | 1989-12-19 | Jagdish    | Reddy     | M      | 2018-07-18   |
|  50006 | 1990-12-06 | Carlos     | Fuller    | M      | 2018-07-18   |
|  50008 | 1989-08-19 | Raj        | Singh     | M      | 2018-07-18   |
|  50009 | 1990-06-06 | Andrew     | Muller    | M      | 2018-07-18   |
|  50010 | 1980-11-06 | Rabin      | RD        | M      | 2018-07-18   |
|  50011 | 1985-10-02 | Firoz      | Ali       | M      | 2019-07-18   |
|  50012 | 1985-10-02 | David      | H         | M      | 2019-07-18   |
+--------+------------+------------+-----------+--------+--------------+
11 rows in set (0.00 sec)
```

d. HSPC automatically creates an iSCSI target on port 1-C of the storage system. Verify whether a dynamically provisioned volume (00:01:EB) of 250 GB was created in the VSP 5200 from storage system.

3. Create a snapshot.

a. Create VolumeSnapshotClass. See Install Hitachi Storage Plug-in for Containers.

b. Create a manifest file to collect snapshot of the persistent volume created in step 2.

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: mysql-manual-snapshot
  namespace: prodmysql
spec:
  volumeSnapshotClassName: snapshotclass-sample
  source:
    persistentVolumeClaimName: prod-vol-prodmysqlapp-0
```

c. Run the oc command using manifest file to create the snapshot.

```
# oc create -f volumesnapshot-mysql-manual.yaml
```

d. Verify that the snapshot is created.

```
# oc get VolumeSnapshot -n prodmysql
NAME                    READYTOUSE    SOURCEPVC
SOURCESNAPSHOTCONTENT   RESTORESIZE    SNAPSHOTCLASS          SNAPSHOTCONTENT
CREATIONTIME    AGE
mysql-manual-snapshot   true          prod-vol-prodmysqlapp-0
250Gi         snapshotclass-sample   snapcontent-ba8ddcf5-38b9-4581-b3c3-
```

```
84bdc386ef07   14d            14d
```

e. From Storage Navigator, verify that the snapshot volume (00:01:EC) is created successfully.

Copy Type: TI

TI History (Page.1)

Filter ON OFF

Date and Time	Primary Volume		Secondary Volume		Mirror Unit	Pool ID	Diff Compare Volume	Description Code	Description
	LDEV ID	Provisioning Type	LDEV ID	Provisioning Type					
2023/04/17 06:17:47	00:01:EB	DP	00:01:EC	DP	3	0	-	2011	PSUS
2023/04/17 06:17:44	00:01:EB	DP	00:01:EC	DP	3	0	-	2001	PAIR

Restore Operation

Complete the following steps in the OpenShift cluster in AWS:

1. Identify the volume handle string for the snapshot secondary volume 00:01:EC. Volume handle string for this LDEV is "60060e80089c5c0000509c5c0000**01ec—spc-208715bccc**". In the string, LDEV ID is "01ec" and LDEV Name is "spc-208715bccc". LDEV name is automatically assigned by HSPC.
2. Create PV and PVC using volume (00:01:EC) with the pre-defined volume handle string.
   a. Create a namespace in the OpenShift cluster in AWS.
   ```
   # oc create namespace devmysql
   ```
   b. Create a manifest file for PV and PVC. Use the volume handle string for PV manifest. This way, storage class does not dynamically create a new volume; instead, it uses the existing volume to preserve the snapshot data.
   ```
   apiVersion: v1
   kind: PersistentVolume
   metadata:
     name: devmysqlpv
     namespace: devmysql
   spec:
     capacity:
       storage: 250Gi
     accessModes:
       - ReadWriteOnce
     persistentVolumeReclaimPolicy: Retain
     storageClassName: sc-vsp5200
     csi:
       driver: hspc.csi.hitachi.com
       volumeHandle: 60060e80089c5c0000509c5c000001EC--spc-208715bccc
     claimRef:
       name: devmysqlpvc
       namespace: devmysql
   ---
   apiVersion: v1
   kind: PersistentVolumeClaim
   metadata:
     name: devmysqlpvc
     namespace: devmysql
   spec:
     accessModes:
     - ReadWriteOnce
     resources:
           requests:
             storage: 250Gi
     volumeName: devmysqlpv
     storageClassName: sc-vsp5200
   ```
   c. To create PV and PVC using manifest, run the following oc command:
   ```
   # oc create -f devmysqlpvc.yaml
   ```

d. Verify that PV and PVC are created as per manifest.

```
[root@ip-10-77-28-159 sw_k10]# oc get pv |grep devmysqlpv
devmysqlpv                    250Gi      RWO            Retain           Bound     devmysql/devmysqlpvc                      sc-vsp5200
[root@ip-10-77-28-159 sw_k10]#
```

```
[root@ip-10-77-28-159 sw_k10]# oc get pvc -n devmysql
NAME          STATUS   VOLUME        CAPACITY   ACCESS MODES   STORAGECLASS   AGE
devmysqlpvc   Bound    devmysqlpv    250Gi      RWO            sc-vsp5200     10s
```

3. Create a manifest file to create a clone PVC using the “devmysqlpvc” PVC as data source.
   a. Create a manifest.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: devmysqlclone
  namespace: devmysql
spec:
  storageClassName: sc-vsp5200
  dataSource:
    name: devmysqlpvc
    kind: PersistentVolumeClaim
    apiGroup: ""
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 250Gi
```

   b. To create the clone PVC, run the following oc command using manifest:

```
# oc create -f devmysqlclone.yaml
```

   c. HSPC dynamically provisions a PV from the VSP 5200 storage system. Verify the PVC and PV.

```
# oc get pvc -n devmysql
NAME            STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
devmysqlclone   Bound    pvc-cd4a340d-538d-41cd-991d-963a3d9605fd   250Gi      RWO
sc-vsp5200     60s
# oc get pv |grep devmysql/devmysqlclone
pvc-cd4a340d-538d-41cd-991d-963a3d9605fd   250Gi      RWO            Delete
Bound    devmysql/devmysqlclone                    sc-vsp5200                4m20s
```

The dynamically created PV is the Thin Image clone volume. In the following screenshot, “00:01:ED” is the designated clone volume.

Copy Type: TI

TI History (Page.1)

Filter ON OFF

Date and Time	Primary Volume		Secondary Volume		Mirror Unit	Pool ID	Diff Compare Volume	Description Code	Description
	LDEV ID	Provisioning Type	LDEV ID	Provisioning Type					
2023/04/17 07:24:12	00:01:EC	DP	00:01:ED	DP	3	0	-	2092	CLONE END
2023/04/17 07:19:33	00:01:EC	DP	00:01:ED	DP	3	0	-	2091	CLONE START
2023/04/17 07:19:32	00:01:EC	DP	00:01:ED	DP	3	0	-	2001	PAIR

4. Restore the MySQL application in the AWS cluster. In the volume section, use the claim “devmysqlclone” created in step 3, which ensures that MySQL application uses the clone PVC for persistent data.
   a. Create a manifest.

```
apiVersion: v1
kind: Service
metadata:
    namespace: devmysql
    name: prodmysqlapp
```

```
    labels:
       app: prodmysqlapp
spec:
    ports:
    - port: 3306
      name: prodmysqlapp
    clusterIP: None
    selector:
       app: prodmysqlapp
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
    namespace: devmysql
    name: prodmysqlapp
spec:
  selector:
    matchLabels:
      app: prodmysqlapp
  serviceName: "prodmysqlapp"
  podManagementPolicy: Parallel
  replicas: 1
  template:
    metadata:
      labels:
        app: prodmysqlapp
    spec:
      terminationGracePeriodSeconds: 30
      containers:
      - name: prodmysqlapp
        image: mysql:5.7
        args:
          - "--ignore-db-dir=lost+found"
        env:
        - name: MYSQL_ROOT_PASSWORD
          value: pass123
        - name: MYSQL_DATABASE
          value: prodmysqldb
        - name: MYSQL_USER
          value: admin
        - name: MYSQL_PASSWORD
          value: secret
        ports:
        - containerPort: 3306
          name: mysql
        volumeMounts:
        - name: prod-vol
          mountPath: /var/lib/mysql
      volumes:
      - name: prod-vol
        persistentVolumeClaim:
          claimName: devmysqlclone
```

b. To create MySQL pod and service, run the following oc command using manifest:

```
# oc create -f prodmysqlapp-sts.yaml
```

c. Verify that MySQL pod and service are created as per manifest.

```
[root@ip-10-77-28-159 sw_k10]# oc get svc -n devmysql
NAME           TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
prodmysqlapp   ClusterIP   None         <none>        3306/TCP   16s
[root@ip-10-77-28-159 sw_k10]# oc get pod -n devmysql
NAME             READY   STATUS    RESTARTS   AGE
prodmysqlapp-0   1/1     Running   0          27s
```

d. Log in to the pod prodmysqlapp-0 and verify whether the 250 GB persistent volume is mounted on "/var/lib/mysql".

```
[root@ip-10-77-28-159 sw_k10]# oc -n devmysql rsh prodmysqlapp-0
sh-4.2$ df -h
Filesystem       Size  Used Avail Use% Mounted on
overlay          200G   30G  171G  15% /
tmpfs             64M     0   64M   0% /dev
tmpfs            7.8G     0  7.8G   0% /sys/fs/cgroup
shm               64M     0   64M   0% /dev/shm
tmpfs            7.8G   50M  7.7G   1% /etc/passwd
/dev/nvme0n1p4   200G   30G  171G  15% /etc/hosts
/dev/sdf         246G  271M  234G   1% /var/lib/mysql
```

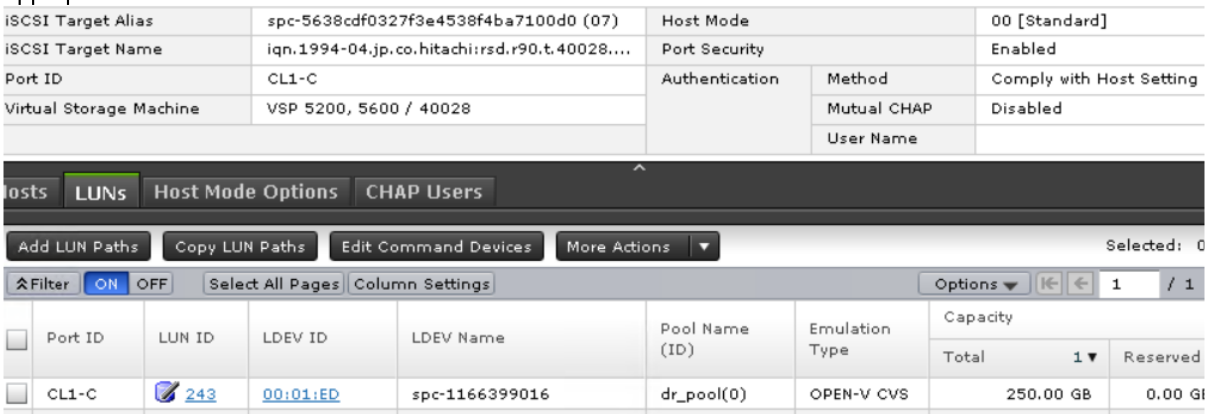
e. Log in to MySQL and verify whether the database "prodmysqldb" is available.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| prodmysqldb        |
| sys                |
+--------------------+
5 rows in set (0.01 sec)
```

f. Verify whether the ingested data from Primary cluster "ocpcluster" is available.

```
mysql> SELECT * FROM employees;
+--------+------------+------------+-----------+--------+--------------+
| emp_no | birth_date | first_name | last_name | gender | joining_date |
+--------+------------+------------+-----------+--------+--------------+
|  50001 | 1988-12-03 | Ramesh     | Kumar     | M      | 2018-07-18   |
|  50002 | 1989-12-03 | Amit       | Kumar     | M      | 2018-09-18   |
|  50003 | 1985-12-06 | Robert     | Fernandez | M      | 2018-07-18   |
|  50004 | 1985-12-06 | Md         | Riaz      | M      | 2018-07-18   |
|  50005 | 1989-12-19 | Jagdish    | Reddy     | M      | 2018-07-18   |
|  50006 | 1990-12-06 | Carlos     | Fuller    | M      | 2018-07-18   |
|  50008 | 1989-08-19 | Raj        | Singh     | M      | 2018-07-18   |
|  50009 | 1990-06-06 | Andrew     | Muller    | M      | 2018-07-18   |
|  50010 | 1980-11-06 | Rabin      | RD        | M      | 2018-07-18   |
|  50011 | 1985-10-02 | Firoz      | Ali       | M      | 2019-07-18   |
|  50012 | 1985-10-02 | David      | H         | M      | 2019-07-18   |
+--------+------------+------------+-----------+--------+--------------+
11 rows in set (0.00 sec)

mysql>
```

HSPC automatically creates an iSCSI target on port 1-C of the storage system and assigns the volume to the appropriate worker node.



Test 5: Recover from a Ransomware Attack

This test case demonstrates how a VSP snapshot combined with immutability feature from Data Retention Utility program product can be used to recover a stateful application affected by a ransomware attack. Assume that application is affected by a ransomware attack and we must restore clean data from the snapshot. This recovery process can be carried out either in the near-cloud or in AWS cluster.

Recovering from a ransomware attack consists of the following high-level steps:

- Creating PVC with the snapshot volume (where the DRU attribute Write Disabled is set).
- Creating a cascaded snapshot of this volume because write is disabled.
- Using the cascaded snapshot (snap-on-snap) to recover the application data in any cluster.
- Creating a clone PVC and using that PVC as data volume to restore the MySQL application because snapshot volumes must not be directly used in a POD.
- Verifying that the data ingested from near-cloud cluster is available.

Snapshot Operation

Complete the following steps in the near-cloud OpenShift cluster:

1. Create a new namespace "drusnapshot" and deploy a Stateful MySQL application with a persistent volume of 250 GB from a VSP 5200 storage system, as shown in Test 2: Deploy a Stateful Application in Red Hat OpenShift Cluster: Deploy in Near-Cloud.
2. Access the stateful MySQL application.
    a. Log in to the pod mysqldru-0 and verify whether the 250 GB persistent volume is mounted on "/var/lib/mysql".

```
[root@linuxnfscl2 ~]# oc -n drusnapshot rsh mysqldru-0
sh-4.2$ df -h
Filesystem      Size  Used Avail Use% Mounted on
overlay         120G   20G  101G  17% /
tmpfs            64M     0   64M   0% /dev
tmpfs           7.8G     0  7.8G   0% /sys/fs/cgroup
shm              64M     0   64M   0% /dev/shm
tmpfs           7.8G   48M  7.8G   1% /etc/passwd
/dev/sda4       120G   20G  101G  17% /etc/hosts
/dev/sdi        246G  271M  234G   1% /var/lib/mysql
tmpfs            15G   20K   15G   1% /run/secrets/kubernetes.io/serviceaccount
tmpfs           7.8G     0  7.8G   0% /proc/acpi
tmpfs           7.8G     0  7.8G   0% /proc/scsi
tmpfs           7.8G     0  7.8G   0% /sys/firmware
```

    b. Log in to MySQL and verify whether the database "drusnapshotdb" is created.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| drusnapshotdb      |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)
```

c. Create a table “student” and ingest some new records to the table.

```
mysql> SELECT * FROM student;
+-----------------+------------+------------+-----------+--------+----------------+
| registration_no | birth_date | first_name | last_name | gender | admission_date |
+-----------------+------------+------------+-----------+--------+----------------+
|           70001 | 1988-12-03 | Promad     | Kumar     | M      | 2018-07-18     |
|           70002 | 1989-12-03 | Ji         | Lehman    | M      | 2018-09-18     |
|           70003 | 1985-12-06 | Robert     | Heiman    | M      | 2018-07-18     |
|           70004 | 1985-12-06 | Amy        | Wildsmith | F      | 2018-07-18     |
|           70005 | 1989-12-19 | Nader      | Hussain   | M      | 2018-07-18     |
|           70006 | 1990-12-06 | Aleberto   | D         | M      | 2018-07-18     |
|           70007 | 1980-12-06 | Amit       | Jain      | M      | 2018-07-18     |
|           70008 | 1989-08-19 | Rakesh     | Singh     | M      | 2018-07-18     |
|           70009 | 1990-06-06 | Amanto     | Pator     | M      | 2018-07-18     |
|           70010 | 1980-11-06 | Kramsa     | Taro      | M      | 2018-07-18     |
|           70011 | 1985-10-02 | Abhay      | Mushary   | M      | 2019-07-18     |
|           70012 | 1985-10-02 | Priyanka   | Timungpi  | F      | 2019-07-18     |
+-----------------+------------+------------+-----------+--------+----------------+
12 rows in set (0.00 sec)
```

3. Create a snapshot.

a. Create a manifest to take volume snapshot of the MySQL application PVC.

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: mysqldrusnapshotnew1
  namespace: drusnapshot
spec:
  volumeSnapshotClassName: snapshotclass-sample
  source:
    persistentVolumeClaimName: dru-vol-mysqldru-0
```

b. To create the snapshot, run the following oc command using manifest:

```
[root@linuxnfscl2 ~]# oc create -f volumesnapshot-mysqldrunew1.yaml
volumesnapshot.snapshot.storage.k8s.io/mysqldrusnapshotnew1 created
[root@linuxnfscl2 ~]#
```

c. Verify that the snapshot is created as per manifest.

```
# oc get volumesnapshot -n drusnapshot
NAME                        READYTOUSE    SOURCEPVC             SOURCESNAPSHOTCONTENT
RESTORESIZE   SNAPSHOTCLASS             SNAPSHOTCONTENT
CREATIONTIME   AGE
mysqldrusnapshotnew1    true          dru-vol-mysqldru-0
250Gi         snapshotclass-sample   snapcontent-4e643b81-4260-4a49-b306-
52aa04f77960   <invalid>             64s
```

d. In Storage Navigator, verify that snapshot volume (00:01:E6) is created successfully.

Copy Type: TI

TI History (Page.1)

Filter ON OFF

Date and Time	Primary Volume		Secondary Volume		Mirror Unit	Pool ID	Diff Compare Volume	Description Code	Description
	LDEV ID	Provisioning Type	LDEV ID	Provisioning Type					
2023/04/13 07:28:40	00:01:E3	DP	00:01:E6	DP	5	0	-	2011	PSUS
2023/04/13 07:28:40	00:01:E3	DP	00:01:E6	DP	5	0	-	2001	PAIR

4. Set DRU attribute (write disable) to snapshot volume 486 (00:01:E6).

```
[root@linuxnfscl2 ~]# raidvchkdsp -g grp0 -fd -v gflag -I1
Group    PairVol  Device_File       Seq# LDEV# GI-C-R-W-S  PI-C-R-W-S  R-Time
grp0     pair0    Unknown          540028    486  E E E E E   E E E E E       0
[root@linuxnfscl2 ~]#
[root@linuxnfscl2 ~]# raidvchkset -g grp0 -vg wtd 5 -I1
[root@linuxnfscl2 ~]# raidvchkdsp -g grp0 -fd -v gflag -I1
Group    PairVol  Device_File       Seq# LDEV# GI-C-R-W-S  PI-C-R-W-S  R-Time
grp0     pair0    Unknown          540028    486  E E E D E   E E E D E       5
[root@linuxnfscl2 ~]#
```

Restore Operation

The section shows the restoration procedure when an application in near-cloud is affected by ransomware.

1. Create PV and PVC for the snapshot volume (00:01: E6).
   a. Identify the volume handle string for the snapshot volume (00:01:E6). The volume handle string for this LDEV is "60060e80089c5c0000509c5c000001e6--spc-439ad69acd". In the string, the LDEV ID is "01e6" and the LDEV name is "spc-439ad69acd". The LDEV name is automatically assigned by HSPC.

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: drusnapshotpv
  namespace: drusnapshot
spec:
  capacity:
    storage: 250Gi
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  storageClassName: sc-vsp5200
  csi:
    driver: hspc.csi.hitachi.com
    volumeHandle: 60060e80089c5c0000509c5c000001e6--spc-439ad69acd
  claimRef:
    name: drusnapshotpvc
    namespace: drusnapshot
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: drusnapshotpvc
  namespace: drusnapshot
spec:
  accessModes:
  - ReadWriteOnce
  resources:
        requests:
          storage: 250Gi
  volumeName: drusnapshotpv
  storageClassName: sc-vsp5200
```

   b. To create PV and PVC, run the following oc command using manifest:

```
# oc create -f drusnapshotpvpvc.yaml
```

   c. Verify that PV and PVC are created.

```
# oc get pvc -n drusnapshot
NAME                 STATUS   VOLUME                                        CAPACITY
ACCESS MODES   STORAGECLASS   AGE
```

```
drusnapshotpvc       Bound    drusnapshotpv                              250Gi
RWO            sc-vsp5200     17s
# oc get pv -n drusnapshot
NAME                                       CAPACITY   ACCESS MODES   RECLAIM POLICY
STATUS   CLAIM                                                      STORAGECLASS
REASON   AGE
drusnapshotpv                              250Gi      RWO             Retain
Bound    drusnapshot/drusnapshotpvc                                 sc-vsp5200
41s
```

2. Create a cascaded snapshot (snapshot of S-VOL "00:01:E6").

   a. To create a cascaded snapshot, use the following manifest:

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshot
metadata:
  name: drusnaponsnap
  namespace: drusnapshot
spec:
  volumeSnapshotClassName: snapshotclass-sample
  source:
    persistentVolumeClaimName: drusnapshotpvc
```

   b. To create a new snapshot, run the following oc command using manifest. A new S-VOL "00:01:E9" will be created.

```
# oc create -f drusnaponsnap.yaml
```

   c. Verify that the new volumesnapshot is created.

```
# oc get volumesnapshot -n drusnapshot
NAME                   READYTOUSE   SOURCEPVC           SOURCESNAPSHOTCONTENT
RESTORESIZE   SNAPSHOTCLASS          SNAPSHOTCONTENT
CREATIONTIME   AGE
drusnaponsnap          true         drusnapshotpvc
250Gi         snapshotclass-sample   snapcontent-43548d17-0c0a-4d37-b1ec-
146b0f2e7cff   <invalid>            9s
```

   d. In Storage Navigator, verify that the snapshot volume ("00:01:E9") is created successfully.

Copy Type: TI

TI History (Page.1)

Filter ON OFF

Date and Time	Primary Volume		Secondary Volume		Mirror Unit	Pool ID	Diff Compare Volume	Description Code	Description
	LDEV ID	Provisioning Type	LDEV ID	Provisioning Type					
2023/04/14 06:54:54	00:01:E6	DP	00:01:E9	DP	4	0	-	2011	PSUS
2023/04/14 06:54:54	00:01:E6	DP	00:01:E9	DP	4	0	-	2001	PAIR

The DRU attribute is no longer available on the new volume ("00:01:09").

```
[root@linuxnfscl2 etc]# raidvchkdsp -g grp1 -fd -v gflag -I3
Group   PairVol  Device_File      Seq# LDEV# GI-C-R-W-S  PI-C-R-W-S  R-Time
grp1    pair1    Unknown        540028   489  E E E E E   E E E E E       0
[root@linuxnfscl2 etc]#
```

3. To create a clone volume for restoring an application, use the snap-on-snap volume.

   a. Create a namespace "drusnaponsnap".

   b. Create a manifest to create PV and PVC from the snap-on-snap volume (00:01:E9). Mentioning the volume handle string prevents storage class from dynamically creating a new volume; instead, use an existing volume to preserve the snapshot data.

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: drusnaponsnappv
  namespace: drusnaponsnap
```

```
spec:
  capacity:
    storage: 250Gi
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  storageClassName: sc-vsp5200
  csi:
    driver: hspc.csi.hitachi.com
    volumeHandle: 60060e80089c5c0000509c5c000001e9—spc-2bdf56bb18
  claimRef:
    name: drusnaponsnappvc
     namespace: drusnaponsnap
    ---
 apiVersion: v1
 kind: PersistentVolumeClaim
 metadata:
  name: drusnaponsnappvc
  namespace: drusnaponsnap
 spec:
  accessModes:
  - ReadWriteOnce
  resources:
        requests:
          storage: 250Gi
  volumeName: drusnaponsnappv
  storageClassName: sc-vsp5200
```

c. To create PV and PVC, run the following oc command using manifest:

```
# oc create -f drusnaponsnappvc.yaml
```

d. Verify that PVC and PV are created.

```
[root@ip-10-77-28-159 sw_k10]# oc get pvc -n drusnaponsnap
NAME               STATUS   VOLUME            CAPACITY   ACCESS MODES   STORAGECLASS   AGE
drusnaponsnappvc   Bound    drusnaponsnappv   250Gi      RWO            sc-vsp5200     38s
[root@ip-10-77-28-159 sw_k10]#
```

```
[root@ip-10-77-28-159 sw_k10]# oc get pv -n drusnaponsnap
NAME              CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                              STORAGECLASS   REASON   AGE
devclusterpv      50Gi       RWO            Retain           Bound    devmysqlnew/devclusterpvc          sc-vsp5200              15d
drusnaponsnappv   250Gi      RWO            Retain           Bound    drusnaponsnap/drusnaponsnappvc     sc-vsp5200              51s
```

e. Create a clone PVC using the snapshot PVC.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: drusnaponsnapclone
  namespace: drusnaponsnap
spec:
  storageClassName: sc-vsp5200
  dataSource:
    name: drusnaponsnappvc
    kind: PersistentVolumeClaim
    apiGroup: ""
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 250Gi
```

f. To create the clone PVC, run the following oc command using manifest:

```
# oc create -f drusnaponsnapclone.yaml
```

g. Verify that the PVC is created.

```
# oc get pvc -n drusnaponsnap
NAME                 STATUS   VOLUME                                     CAPACITY
ACCESS MODES   STORAGECLASS   AGE
drusnaponsnapclone   Bound    pvc-b3981ce6-c788-44fd-a169-11febdff2782   250Gi
RWO            sc-vsp5200     18d
```

h. In Storage Navigator, verify that the clone volume (00:01:EA) is created successfully.

Copy Type: TI

TI History (Page.1)

Filter ON OFF

Date and Time	Primary Volume		Secondary Volume		Mirror Unit	Pool ID	Diff Compare Volume	Description Code	Description
	LDEV ID	Provisioning Type	LDEV ID	Provisioning Type					
2023/04/14 08:20:30	00:01:E9	DP	00:01:EA	DP	3	0	-	2092	CLONE END
2023/04/14 08:15:50	00:01:E9	DP	00:01:EA	DP	3	0	-	2091	CLONE START
2023/04/14 08:15:49	00:01:E9	DP	00:01:EA	DP	3	0	-	2001	PAIR

4. Restore the MySQL application using the clone PVC.

a. Create a manifest of stateful MySQL application using the clone PVC (“drusnaponsnapclone”).

```
apiVersion: v1
kind: Service
metadata:
    namespace: drusnaponsnap
    name: mysqldruapp
    labels:
       app: mysqldruapp
spec:
    ports:
    - port: 3306
      name: mysqldruapp
    clusterIP: None
    selector:
       app: mysqldruapp
---
apiVersion: apps/v1
kind: StatefulSet
metadata:
    namespace: drusnaponsnap
    name: mysqldruapp
spec:
  selector:
    matchLabels:
      app: mysqldruapp
  serviceName: "mysqldruapp"
  podManagementPolicy: Parallel
  replicas: 1
  template:
    metadata:
      labels:
        app: mysqldruapp
    spec:
      terminationGracePeriodSeconds: 30
      containers:
      - name: mysqldruapp
        image: mysql:5.7
        args:
          - "--ignore-db-dir=lost+found"
        env:
        - name: MYSQL_ROOT_PASSWORD
          value: pass123
        - name: MYSQL_DATABASE
          value: drusnapshotdb
        - name: MYSQL_USER
```

```
        value: admin
      - name: MYSQL_PASSWORD
        value: secret
      ports:
      - containerPort: 3306
        name: mysql
      volumeMounts:
      - name: dru-vol
        mountPath: /var/lib/mysql
    volumes:
    - name: dru-vol
      persistentVolumeClaim:
        claimName: drusnaponsnapclone
```

b. To create MySQL pod and service, run the following oc command using manifest:

```
# oc create -f mysqldruapp.yaml
```

c. Verify that pod and MySQL service are created as per manifest.

```
# oc get svc -n drusnaponsnap
NAME          TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
mysqldruapp   ClusterIP   None         <none>        3306/TCP   18d
# oc get pod -n drusnaponsnap
NAME            READY   STATUS    RESTARTS   AGE
mysqldruapp-0   1/1     Running   0          18d
```

d. Log in to the pod “mysqldruapp-0”and verify whether the 250 GB persistent volume is mounted on “/var/lib/mysql”.

```
[root@ip-10-77-28-159 sw_k10]# oc -n drusnaponsnap rsh mysqldruapp-0
sh-4.2$ df -h
Filesystem      Size  Used Avail Use% Mounted on
overlay         200G   31G  170G  16% /
tmpfs            64M     0   64M   0% /dev
tmpfs           7.8G     0  7.8G   0% /sys/fs/cgroup
shm              64M     0   64M   0% /dev/shm
tmpfs           7.8G   52M  7.7G   1% /etc/passwd
/dev/nvme0n1p4  200G   31G  170G  16% /etc/hosts
/dev/sdl        246G  271M  234G   1% /var/lib/mysql
tmpfs            15G   20K   15G   1% /run/secrets/kubernetes.io/serviceaccount
tmpfs           7.8G     0  7.8G   0% /proc/acpi
tmpfs           7.8G     0  7.8G   0% /proc/scsi
tmpfs           7.8G     0  7.8G   0% /sys/firmware
```

e. Log in to MySQL and verify whether the database “drusnapshotdb” is available.

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| drusnapshotdb      |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.01 sec)
```

f. Verify whether the ingested data before taking snapshot is available after restoring.

```
mysql> show tables;
+------------------------+
| Tables_in_drusnapshotdb |
+------------------------+
| student                |
+------------------------+
1 row in set (0.00 sec)

mysql> SELECT * FROM student;
+-----------------+------------+------------+-----------+--------+----------------+
| registration_no | birth_date | first_name | last_name | gender | admission_date |
+-----------------+------------+------------+-----------+--------+----------------+
|           70001 | 1988-12-03 | Promad     | Kumar     | M      | 2018-07-18     |
|           70002 | 1989-12-03 | Ji         | Lehman    | M      | 2018-09-18     |
|           70003 | 1985-12-06 | Robert     | Heiman    | M      | 2018-07-18     |
|           70004 | 1985-12-06 | Amy        | Wildsmith | F      | 2018-07-18     |
|           70005 | 1989-12-19 | Nader      | Hussain   | M      | 2018-07-18     |
|           70006 | 1990-12-06 | Aleberto   | D         | M      | 2018-07-18     |
|           70007 | 1980-12-06 | Amit       | Jain      | M      | 2018-07-18     |
|           70008 | 1989-08-19 | Rakesh     | Singh     | M      | 2018-07-18     |
|           70009 | 1990-06-06 | Amanto     | Pator     | M      | 2018-07-18     |
|           70010 | 1980-11-06 | Kramsa     | Taro      | M      | 2018-07-18     |
|           70011 | 1985-10-02 | Abhay      | Mushary   | M      | 2019-07-18     |
|           70012 | 1985-10-02 | Priyanka   | Timungpi  | F      | 2019-07-18     |
+-----------------+------------+------------+-----------+--------+----------------+
12 rows in set (0.00 sec)
```

g. Verify whether HSPC automatically created an iSCSI target on port 1-C and whether the clone volume "01:01:EA" is mounted.

iSCSI Target Alias	spc-88364666e69015b02bd6e93f1efd (0B)	Host Mode		00 [Standard]
iSCSI Target Name	iqn.1994-04.jp.co.hitachi:rsd.r90.t.40028....	Port Security		Enabled
Port ID	CL1-C	Authentication	Method	Comply with Host Setting
Virtual Storage Machine	VSP 5200, 5600 / 40028		Mutual CHAP	Disabled
			User Name	

osts | LUNs | Host Mode Options | CHAP Users

Add LUN Paths | Copy LUN Paths | Edit Command Devices | More Actions | Selected:

Filter ON OFF | Select All Pages | Column Settings | Options | 1 / 1

Port ID	LUN ID	LDEV ID	LDEV Name	Pool Name (ID)	Emulation Type	Capacity Total	Capacity Reserved
CL1-C	63	00:01:EA	spc-876fed1c72	dr_pool(0)	OPEN-V CVS	250.00 GB	0.00 G

h. Delete the snap-on-snap PV and PVC created in step 3b.

```
[root@ip-10-77-28-159 sw_k10]# oc delete pvc drusnaponsnappvc -n drusnaponsnap
persistentvolumeclaim "drusnaponsnappvc" deleted
[root@ip-10-77-28-159 sw_k10]#
```

```
[root@ip-10-77-28-159 sw_k10]# oc delete pv drusnaponsnappv
persistentvolume "drusnaponsnappv" deleted
```