Getting Started with Storage and Virtualization

NetApp virtualization solutions

NetApp
July 24, 2026

Table of Contents

Getting Started with Storage and Virtualization

Learn about NetApp Trident integration with Red Hat OpenShift

Storage is crucial in containerized applications, ensuring data is persistently available and efficiently managed across dynamic and often ephemeral environments. Modern storage solutions for containers are designed to scale seamlessly with applications, providing the flexibility to dynamically allocate storage resources based on application needs, supporting both growth and varying workloads. Since OpenShift Virtualization enables the VMs to run as containers, they use the same storage as other applications in the OpenShift Clusters.

NetApp ONTAP is a robust and highly scalable data management software that serves as the backbone for NetApp's enterprise-grade storage solutions. It provides a unified storage architecture that integrates seamlessly with various storage environments, including on-premises, cloud, and hybrid setups. ONTAP supports a variety of protocols such as NFS, CIFS/SMB, iSCSI, FC, and NVMe over TCP, allowing for seamless integration with different applications and workloads. It enables the consolidation of SAN, NAS, and object storage into a single platform, simplifying management and reducing costs. NetApp ONTAP is available on-premises as well as managed services in the cloud.

NetApp Trident

Trident is an open-source storage orchestrator developed by NetApp, designed to facilitate the use of ONTAP storage in containerized environments. It enables dynamic provisioning and management of persistent storage for containerized applications.

Key features of Trident

CSI Compliance
Ensuring compatibility with the latest Kubernetes storage features and standards.
Declarative Configuration
Allowing users to define storage requirements in YAML files, which are then automatically managed by Trident.
Drivers
Trident supports drivers for NFS, CIFS/SMB, iSCSI, FC and NVMe/TCP protocols supported by ONTAP.
Hybrid and multi cloud support
Trident supports drivers for ONTAP storage on-premises and managed storage service in the hyperscalers namely FSx for NetApp ONTAP in AWS, Azure NetApp Files in Azure and Google Cloud NetApp Volumes in Google Cloud.
Advanced Data Management
Leveraging ONTAP's snapshot and cloning capabilities for efficient data protection and rapid provisioning of test and development environments.

For complete details on Trident, refer to NetApp Trident documentation here

Trident Protect

Trident Protect is an advanced feature of Trident that provides enhanced data protection and security for containerized environments. It is designed to ensure that critical data is safeguarded against potential threats and data loss scenarios. It is available free of cost for all Trident users and can be easily enabled in any

OpenShift cluster with Trident installed.

Key features of Trident Protect

Automated Backups
Trident Protect enables automated, policy-driven backups of container applications and VMs, ensuring that they are regularly backed up without manual intervention.

Snapshot Management
It leverages ONTAP's snapshot capabilities to create frequent, point-in-time copies container applications and VMs, providing a reliable mechanism for recovery.

Data Encryption
Ensures that data is encrypted both at rest and in transit, meeting stringent security requirements and protecting sensitive information.

Compliance and Auditing
Provides tools and features that help organizations meet compliance requirements and conduct regular audits of their data protection practices.

Disaster Recovery
Integrates with ONTAP's replication features to provide robust disaster recovery solutions, ensuring business continuity even in the face of catastrophic events.

For complete details on Trident Protect, refer to NetApp Trident Protect documentation here.

NetApp ONTAP hardware models and Protocols supported

NetApp ONTAP software runs on a variety of hardware models, each designed to cater to different performance and capacity requirements. Trident extends its robust capabilities to support various NetApp ONTAP systems, including All Flash FAS (AFF), All SAN Array (ASA), and ASA R2 systems. This support ensures that organizations can leverage the full potential of their high-performance storage solutions in containerized environments.

All Flash FAS (AFF) Systems
AFF systems deliver exceptional performance and low latency, making them ideal for high-demand applications.

All SAN Array (ASA) Systems
ASA systems are designed for dedicated SAN environments, offering robust performance and reliability.

ASA R2 Systems
ASA R2 systems provide enhanced features and capabilities for SAN environments.

AFX
NetApp AFX is a purpose-built, disaggregated, all-flash storage architecture designed for enterprise AI workloads. It offers high-performance NAS, allowing independent scaling of compute and capacity. NetApp AFX storage systems differ from other ONTAP-based systems (ASA, AFF, and FAS) in the implementation of their storage layer. AFX uses a distributed file system that allows for high performance and scalability, while other ONTAP-based systems use a traditional storage architecture.

1. Protocols supported for AFF: NFS, CIFS/SMB, iSCSI, FC, NVMe/TCP
2. Protocols supported for ASA: iSCSI, FC, NVMe/TCP

3. Protocols supported for ASA R2: iSCSI, FC, NVMe/TCP
4. Protocols supported for AFX: Starting with Trident 25.10, only NFS protocol is supported; SMB protocol is not supported.

Trident drivers for ONTAP storage

Trident includes the following CSI drivers, each capable of provisioning a volume in the backend storage

For NAS protocols on supported hardware models

1. ontap-nas: One pvc maps to 1 pv which is a dedicated flexvolume
2. ontap-nas-economy: Each PV provisioned is a qtree, with a configurable number of qtrees per FlexVolume (default is 200, configurable between 50 and 300).
   One pvc maps to 1 pv which is a qtree on a shared flexvolume

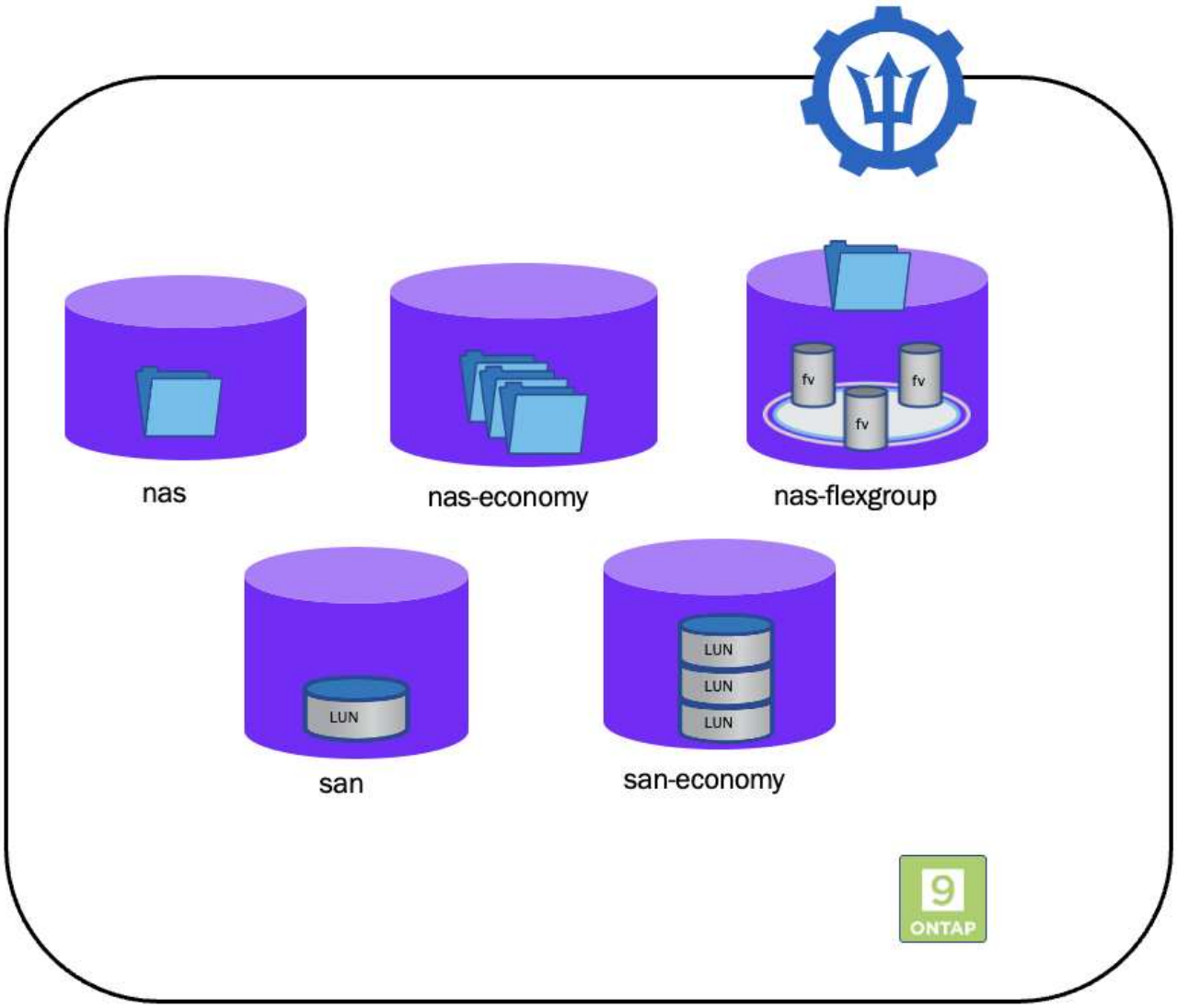
> You can place all the disks of the VMs as qtrees in a single volume. However, please be aware that Snapshots, Clones and Replication features are not supported by Trident. When those features are managed by the Storage administrator, they are not PV granular.

1. ontap-nas-flexgroup: Each PV provisioned is a full ONTAP FlexGroup, and all aggregates assigned to a SVM are used.

For SAN protocols on supported hardware models

1. ontap-san: One pvc maps to 1 pv which is a LUN on dedicated flexvolume
2. ontap-san-economy: One pvc maps to 1 pv which is a LUN on shared flexvolume. You can have multiple LUNs in the shared flexvolume.(default is 100, configurable between 50 and 200 LUNs/PVs per flexvol volume)

> You can place all the disks of the VMs as LUNs in a single volume. However, this driver, while it supports Snapshots and Clones, it does not support Replication. When Replication is managed by Storage administrator, it is not PV granular.



For the complete list of capabilities exposed through Trident drivers and those that must be applied by the storage Administrator, see ONTAP backend drivers section in the Trident documentation.

Install Trident on a Red Hat OpenShift cluster and create storage objects

Install Trident with the Red Hat Certified Trident Operator and create storage objects for ONTAP and Amazon FSx for NetApp ONTAP to enable dynamic volume provisioning for containers and VMs. Prepare worker nodes for block access when required.

Before you begin

- Complete the procedures on this page before you install OpenShift Virtualization. OpenShift Virtualization requires a default Trident-backed StorageClass and VolumeSnapshotClass to create golden images for VM templates.
- If you already installed OpenShift Virtualization before configuring Trident, delete any golden images created with a different storage class. After you set Trident as the default, OpenShift Virtualization recreates the golden images using Trident storage.

```
oc delete dv,VolumeSnapshot -n openshift-virtualization-os-images
--selector=cdi.kubevirt.io/dataImportCron
```

Step 1: Install Trident

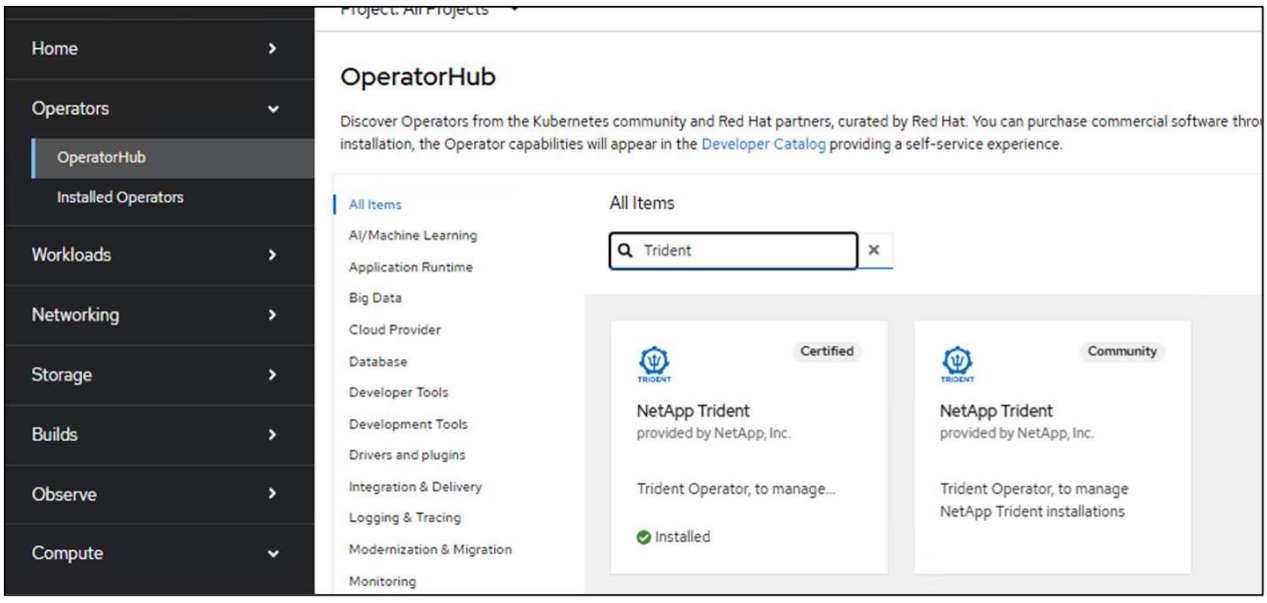
The Red Hat Certified Trident Operator is supported by NetApp for OpenShift on-premises, in public clouds, and in managed services such as ROSA. Starting with Trident 25.02, the operator can also prepare worker nodes for iSCSI when you use Amazon FSx for NetApp ONTAP and plan to run OpenShift Virtualization VM workloads.

For other installation options, see the Trident documentation.

Steps

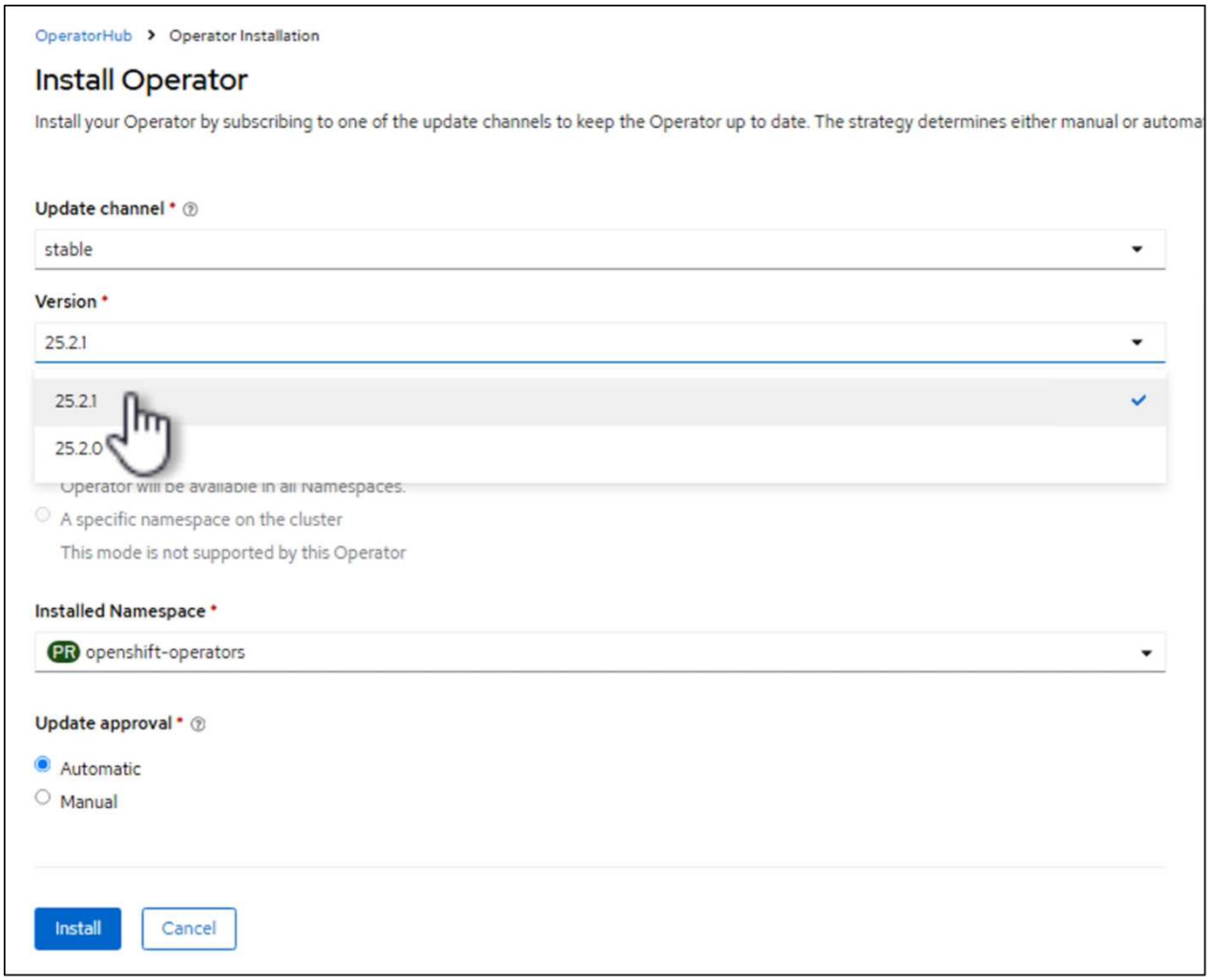
1. In **OperatorHub**, select **Certified NetApp Trident**.

   Show example



2. On the **Install** page, keep the latest version and select **Install**.

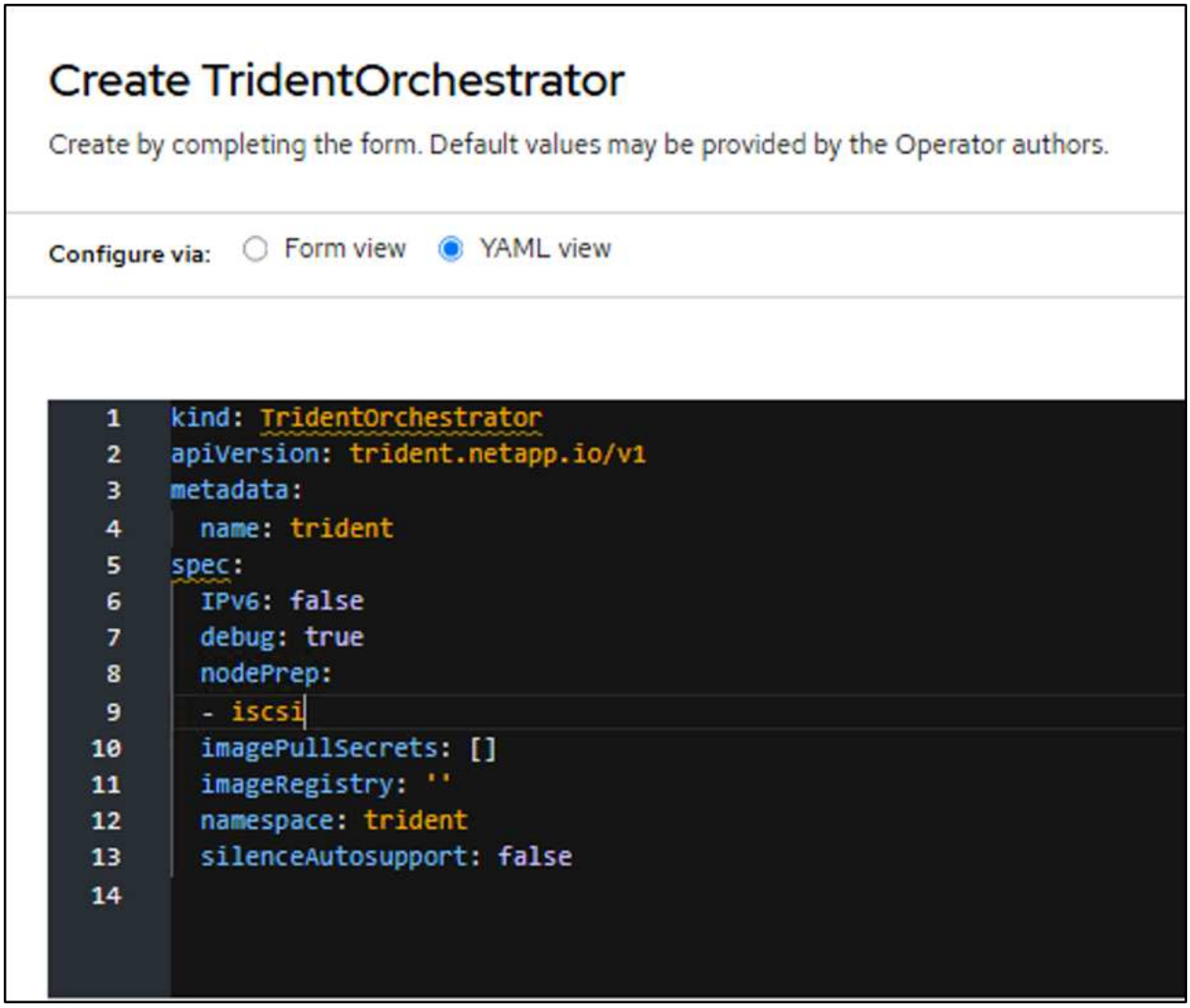
Show example



3. After the operator installs, select **View operator** and create an instance of the Trident Orchestrator.

   If you want to prepare worker nodes for iSCSI, switch to YAML view and add `iscsi` to `nodePrep`.

Show example



4. Confirm that all Trident pods are running in the cluster.

Show example

```
[root@localhost ~]# oc get pods -n trident
NAME                                READY   STATUS    RESTARTS   AGE
trident-controller-84cb9bff89-lkx6k   6/6     Running   0          16h
trident-node-linux-d88b9            2/2     Running   0          16h
trident-node-linux-ld4b8            2/2     Running   0          16h
trident-node-linux-mj5r8            2/2     Running   0          16h
trident-node-linux-mkmmp            2/2     Running   0          16h
trident-node-linux-qhgr7            2/2     Running   0          16h
trident-node-linux-vt9tp            2/2     Running   0          16h
[root@localhost ~]#
```

5. If you enabled iSCSI node preparation, log in to worker nodes and verify that `iscsid` and `multipathd` are active and that `multipath.conf` has entries.

Show example

```
sh-5.1# systemctl status iscsid
● iscsid.service - Open-iSCSI
     Loaded: loaded (/usr/lib/systemd/system/iscsid.service; enabled; preset: disabled)
     Active: active (running) since Fri 2025-04-25 00:23:49 UTC; 3 days ago
TriggeredBy: ● iscsid.socket
       Docs: man:iscsid(8)
             man:iscsiuio(8)
             man:iscsiadm(8)
   Main PID: 74787 (iscsid)
     Status: "Ready to process requests"
      Tasks: 1 (limit: 410912)
     Memory: 1.8M
        CPU: 6ms
     CGroup: /system.slice/iscsid.service
             └─74787 /usr/sbin/iscsid -f

Apr 25 00:23:49 ocp11-worker1 systemd[1]: Starting Open-iSCSI...
Apr 25 00:23:49 ocp11-worker1 systemd[1]: Started Open-iSCSI.
sh-5.1#
```

Show example

```
sh-5.1# systemctl status multipathd
● multipathd.service - Device-Mapper Multipath Device Controller
     Loaded: loaded (/usr/lib/systemd/system/multipathd.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-04-25 00:23:50 UTC; 3 days ago
TriggeredBy: ● multipathd.socket
    Process: 74905 ExecStartPre=/sbin/modprobe -a scsi_dh_alua scsi_dh_emc scsi_dh_rdac dm-multipath (code=exited, status=0/SUCCESS)
    Process: 74906 ExecStartPre=/sbin/multipath -A (code=exited, status=0/SUCCESS)
   Main PID: 74907 (multipathd)
     Status: "up"
      Tasks: 7
     Memory: 18.3M
        CPU: 23.008s
     CGroup: /system.slice/multipathd.service
             └─74907 /sbin/multipathd -d -s

Apr 25 00:23:50 ocp11-worker1 systemd[1]: Starting Device-Mapper Multipath Device Controller...
Apr 25 00:23:50 ocp11-worker1 multipathd[74907]: --------start up--------
Apr 25 00:23:50 ocp11-worker1 multipathd[74907]: read /etc/multipath.conf
Apr 25 00:23:50 ocp11-worker1 multipathd[74907]: path checkers start up
Apr 25 00:23:50 ocp11-worker1 systemd[1]: Started Device-Mapper Multipath Device Controller.
sh-5.1#
```

Show example

```
sh-5.1# cat /etc/multipath.conf
defaults {
    find_multipaths no
}
blacklist {
    device {
        product .*
        vendor  .*
    }
}
blacklist_exceptions {
    device {
        product LUN
        vendor  NETAPP
    }
}
sh-5.1#
```

Video Demonstration

The following video shows a demonstration of installing Trident using the Red Hat Certified Trident Operator.

Installing Trident 25.02.1 using the certified Trident Operator in OpenShift

Step 2: Prepare storage backend and StorageClass configuration files for your environment

Create TridentBackendConfig and StorageClass definitions for your environment. You can configure multiple storage protocols within your environment. Create YAML files for each protocol you want to use, and replace the placeholder values with your specific configuration details.

Complete either the on-premises or ROSA section below based on your environment, then proceed to Step 3.

On-premises OpenShift clusters

Create YAML files for each protocol you want to configure. You can configure one or more of the following protocols: NAS for NFS-based file storage, iSCSI for iSCSI block storage, NVMe/TCP for high-performance NVMe over TCP block storage, or FC for Fibre Channel block storage.

Create a TridentBackendConfig and StorageClass for each protocol. The backend configuration includes credentials stored in a Kubernetes Secret.

Essential guidelines for backend configuration basic options

1. The backend configuration files reference your **ONTAP SVM** and **management LIF**. +
2. You can use **username and password** to authenticate with ONTAP. It can be cluster admin or vserver admin username and password. +
3. You can also use a **client certificate** for authentication with ONTAP. For details about how to generate the client certificate, private key and CA certificate, please review Trident documentation here.

Essential guidelines for backend configuration options for NAS economy driver

The following parameters can be included for **NAS economy driver** configuration:

1. **qtreesPerFlexvol** The value must be in range 50-300, default value is 200. +
2. **limitVolumePoolSize** The value is the maximum FlexVol size that you request when using qtrees in ontap-nas-economy backend. No default value is set, and this parameter is not enforced by default. If you set this parameter, the Trident operator creates a new FlexVol when the current FlexVol reaches the specified size. This parameter can help you control capacity utilization in your cluster by restricting the size of FlexVols used for provisioning PVs. +
3. **denyNewVolumePools** This parameter restricts ontap-nas-economy backends from creating new FlexVol volumes to contain their qtrees. Only preexisting Flexvols are used for provisioning new PVs.

Essential guidelines for backend configuration options for SAN driver

The following parameters can be included for **SAN driver** configuration:

1. **limitVolumeSize** The value is the maximum volume size that you request when using SAN backend. No default value is set, and this parameter is not enforced by default. If you set this parameter, the Trident operator fails provisioning if requested volume size is above this value. You can use this parameter to restrict the maximum size of volumes it manages for LUNs. +
2. **lunsPerFlexvol** Maximum LUNs per Flexvol, must be in range [50, 200], default is 100. If you set this parameter, the Trident operator creates a new FlexVol when the current FlexVol reaches the specified number of LUNs. This parameter can help you control capacity utilization in your cluster by restricting the number of LUNs per FlexVol used for provisioning PVs.

Essential guidelines for backend configuration options for SAN economy driver

The following parameters can be included for **SAN economy driver** configuration:

1. **limitVolumePoolSize** The value is the maximum FlexVol size that you request when using SAN economy backend. No default value is set, and this parameter is not enforced by default. If you set this parameter, the Trident operator limits the maximum flexvol size that can be created for creating LUNs in the flexvol. This parameter can help you control capacity utilization in your cluster by restricting the size of FlexVols used for provisioning PVs. +
2. **denyNewVolumePools** This parameter restricts SAN economy backends from creating new FlexVol volumes to contain their LUNs. Only preexisting Flexvols are used for provisioning new PVs.

Essential guidelines for backend configuration options for provisioning volumes

> In the sample yaml files provided for each protocol, if storagePrefix is not provided, the default prefix “trident” is used. When using ontap-nas-economy and a storagePrefix that is 24 or more characters, the qtrees will not have the storage prefix embedded, though it will be in the volume name.

When you create a volume in ONTAP directly from system manager or from CLI, without providing values for all the optional parameters, there are some default values set automatically during provisioning of a volume as shown below.

```
NSOL-NetApp-A70-T19U05::> volume create -vserver oc-test-automation -volume test_ontap_default  -aggregate NSOL_NetApp_A70_T19U05a_NVME_SSD_1 -size 10GB
[Job 22836] Job succeeded: Successful
```

```
NSOL-NetApp-A70-T19U05::> volume show -vserver oc-test-automation -volume test_ontap_default -fields snapshot-policy,qos-adaptive-policy-group,snapshot-reserve-available,tiering-policy,uni
x-permissions,snapdir-access,space-guarantee
vserver           volume            unix-permissions space-guarantee snapdir-access snapshot-policy qos-adaptive-policy-group snapshot-reserve-available tiering-policy
----------------- ----------------- ---------------- ---------------- ------------- --------------- -------------------------- -------------------------- --------------
oc-test-automation test_ontap_default ---rwxr-xr-x     none           true          default         -                          511.8MB                    none
```

Here is an example of a volume created using Trident backend configuration. It does not have any options set explicitly in the defaults section except the name template.

```
oc describe tbc -n trident tbc-nas
### output truncated for brevity
Spec:
  Backend Name:  tbc-nas
  Credentials:
    Name:  tbc-nas-secret
  Defaults: # There are no defaults set except for the volume name
template.
    Name Template:       {{ .config.StoragePrefix }}_{{
.volume.Namespace }}_{{ .volume.RequestName }}
  Management LIF:        10.192.102.50
  Storage Driver Name:  ontap-nas
  Storage Prefix:        testautomation
  Svm:                   oc-test-automation
```

When the volume is created using this backend configuration, the options that are not set in the backend configuration file are set to the default values as shown below.

```
NSOL-NetApp-A70-T19U05::> volume show -vserver oc-test-automation -volume testautomation_default_test_trident_defaults_91653 -fields snapshot-policy,qos-adaptive-policy-group,snapshot-rese
rve-available,tiering-policy,unix-permissions,snapdir-access,space-guarantee
vserver           volume                                          unix-permissions space-guarantee snapdir-access snapshot-policy qos-adaptive-policy-group snapshot-reserve-available t
iering-policy
----------------- ----------------------------------------------- ---------------- ---------------- ------------- --------------- -------------------------- -------------------------- -
-------------
oc-test-automation testautomation_default_test_trident_defaults_91653 ---rwxrwxrwx     none           false         none          -                          0B                         n
one
```

However, when you set values in the default section in the trident backend configuration, you can create volumes that can have specific values for those parameters.
In the following example, snapshotPolicy is set to default policy (it is none if you do not set anything explicitly) and snapshotDir is set to true in the default section of the backend configuration yaml. With these settings, you can see that the backend volume gets those values. (snapdir-access is true and snapshot-policy is default)

```
[root@00-50-56-b5-01-5b pvcs-br]# oc describe tbc -n trident tbc-nas
Name:         tbc-nas
Namespace:    trident
Labels:       <none>
Annotations:  <none>
API Version:  trident.netapp.io/v1
Kind:         TridentBackendConfig
Metadata:
  Creation Timestamp:  2026-03-30T20:06:59Z
  Finalizers:
    trident.netapp.io
  Generation:        3
  Resource Version:  81358056
  UID:               4dc24339-ff66-4345-acc1-37ec73d14140
Spec:
  Backend Name:  tbc-nas
  Credentials:
    Name:  tbc-nas-secret
  Defaults:
    Name Template:      {{ .config.StoragePrefix }}_{{ .volume.Namespace }}_{{ .volume.RequestName }}
    Snapshot Dir:       true
    Snapshot Policy:    default
  Management LIF:       10.192.102.50
  Storage Driver Name:  ontap-nas
  Storage Prefix:       testautomation
  Svm:                  oc-test-automation
  Version:              1
Status:
  Backend Info:
    Backend Name:         tbc-nas
    Backend UUID:         210cc722-b394-489d-8e9e-49018d3e8710
  Deletion Policy:        delete
  Last Operation Status:  Success
  Message:                Backend 'tbc-nas' updated
  Phase:                  Bound
```

```
NSOL-NetApp-A70-T19U05::> volume show -vserver oc-test-automation -volume testautomation_default_test_trident_set_defaults_7567a -fields snapshot-policy,qos-adaptive-policy-group,snapshot-
reserve-available,tiering-policy,unix-permissions,snapdir-access,space-guarantee
vserver           volume                                                unix-permissions space-guarantee snapdir-access snapshot-policy qos-adaptive-policy-group snapshot-reserve-availab
le tiering-policy
----------------- ----------------------------------------------------- ---------------- -------------- -------------- --------------- ------------------------- ----------------------
-- --------------
oc-test-automation testautomation_default_test_trident_set_defaults_7567a ---rwxrwxrwx     none           true           default         -                         269.5MB
   none
```

```
NSOL-NetApp-A70-T19U05::> volume show -vserver oc-test-automation -volume testautomation_default_test_trident_set_defaults_7567a -fields snapshot-policy,snapdir-access
vserver           volume                                                snapdir-access snapshot-policy
----------------- ----------------------------------------------------- -------------- ---------------
oc-test-automation testautomation_default_test_trident_set_defaults_7567a true           default
```

For all other additional parameters, please review Trident documentation:

1. Trident NAS backend configuration options +
2. Trident SAN backend configuration options

NAS

Create a TridentBackendConfig and StorageClass for ONTAP NAS to enable NFS-based persistent storage provisioning. The backend configuration includes credentials stored in a Kubernetes Secret and references your ONTAP SVM and management LIF.

Sample backend secret and backend configuration file using username and password authentication (save as `tbc-nas.yaml`):

```
# tbc-nas.yaml
apiVersion: v1
kind: Secret
metadata:
  name: tbc-nas-secret
type: Opaque
stringData:
  username: <cluster admin username>
  password: <cluster admin password>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-nas
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: <ONTAP management LIF>
  backendName: tbc-nas
  svm: zoneb #<replace with your SVM name>
  storagePrefix: testzoneb #<replace with your prefix>
  defaults:
    nameTemplate: "{{ .config.StoragePrefix }}_{{ .volume.Namespace
}}_{{ .volume.RequestName }}"
  credentials:
    name: tbc-nas-secret
```

Sample backend secret and backend configuration file using client certificate authentication (save as `tbc-nas.yaml`):

```
# tbc-nas.yaml
apiVersion: v1
kind: Secret
metadata:
  name: ontap-nas-secret
  namespace: trident
type: Opaque
stringData:
  clientPrivateKey: <client private>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-nas
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-nas
  managementLIF: <ONTAP management LIF>
  backendName: tbc-nas
  svm: zoneb #<replace with your SVM name>
  storagePrefix: testzoneb #<replace with your prefix>
  credentials:
    name: ontap-nas-secret
  clientCertificate: <client certificate>
```

StorageClass definition (save as `sc-nas.yaml`):

```
# sc-nas.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-nas
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  media: "ssd"
  provisioningType: "thin"
  snapshots: "true"
allowVolumeExpansion: true
```

NAS economy

Create a TridentBackendConfig and StorageClass for ONTAP NAS economy driver to enable NFS-based persistent storage provisioning using qtrees. The backend configuration includes credentials stored in a

Kubernetes Secret and references your ONTAP SVM and management LIF.

Sample backend secret and backend configuration file using username and password authentication (save as `tbc-nas-economy.yaml`):

```
# tbc-nas-economy.yaml
apiVersion: v1
kind: Secret
metadata:
  name: tbc-nas-economy-secret
type: Opaque
stringData:
  username: <cluster admin username>
  password: <cluster admin password>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-nas-economy
spec:
  version: 1
  storageDriverName: ontap-nas-economy
  managementLIF: <ONTAP management LIF>
  backendName: tbc-nas-eco
  svm: <replace with your SVM name>
  storagePrefix: <replace with your prefix>
  defaults:
    nameTemplate: "{{ .config.StoragePrefix }}_{{ .volume.Namespace
}}_{{ .volume.RequestName }}"
  credentials:
    name: tbc-nas-economy-secret
```

StorageClass definition (save as `sc-nas-economy.yaml`):

```
# sc-nas-economy.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-nas-economy
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas-economy"
  media: "ssd"
  provisioningType: "thin"
  snapshots: "true"
allowVolumeExpansion: true
```

Please see the Essential guidelines for backend configuration options for NAS economy driver section above for details about additional parameters that can be included in the backend configuration file for ontap-nas-economy driver.

iSCSI SAN

Create a TridentBackendConfig and StorageClass for ONTAP SAN with iSCSI to enable iSCSI-based block storage provisioning. The backend configuration uses the ontap-san driver and includes credentials stored in a Kubernetes Secret. The sanType parameter defaults to iSCSI protocol if omitted.

Backend secret and backend configuration file using username password authentication (save as `tbc-iscsi.yaml`):

```
# tbc-iscsi.yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-iscsi-secret
type: Opaque
stringData:
  username: <cluster admin username>
  password: <cluster admin password>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: ontap-iscsi
spec:
  version: 1
  storageDriverName: ontap-san
  managementLIF: <management LIF>
  backendName: ontap-iscsi
  svm: <SVM name>
  credentials:
    name: backend-tbc-ontap-iscsi-secret
```

Backend secret and backend configuration file using client certificate authentication (save as `tbc-iscsi.yaml`):

```
# tbc-iscsi.yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-iscsi-secret
type: Opaque
stringData:
  clientPrivateKey: <client private key>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: ontap-iscsi
spec:
  version: 1
  storageDriverName: ontap-san
  managementLIF: <management LIF>
  backendName: ontap-iscsi
  svm: <SVM name>
  credentials:
    name: backend-tbc-ontap-iscsi-secret
  clientCertificate: <client certificate>
```

StorageClass definition (save as `sc-iscsi.yaml`):

```
# sc-iscsi.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-iscsi
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
  media: "ssd"
  provisioningType: "thin"
  fsType: ext4
  snapshots: "true"
allowVolumeExpansion: true
```

NVMe/TCP SAN

Create a TridentBackendConfig and StorageClass for ONTAP SAN with NVMe over TCP to enable high-performance block storage provisioning. The backend configuration uses the ontap-san driver optimized for NVMe/TCP transport and includes credentials stored in a Kubernetes Secret.

Backend secret and backend configuration file using username password authentication (save as `tbc-nvme.yaml`):

```
# tbc-nvme.yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-nvme-secret
type: Opaque
stringData:
  username: <cluster admin username>
  password: <cluster admin password>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nvme
spec:
  version: 1
  storageDriverName: ontap-san
  sanType: nvme
  managementLIF: <ONTAP management LIF>
  backendName: backend-tbc-ontap-nvme
  svm: <SVM name>
  credentials:
    name: backend-tbc-ontap-nvme-secret
```

Backend secret and backend configuration file using client certificate authentication (save as `tbc-nvme.yaml`):

```
# tbc-nvme.yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-ontap-nvme-secret
type: Opaque
stringData:
  clientPrivateKey: <client private key>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-tbc-ontap-nvme
spec:
  version: 1
  storageDriverName: ontap-san
  sanType: nvme
  managementLIF: <ONTAP management LIF>
  backendName: backend-tbc-ontap-nvme
  svm: <SVM name>
  credentials:
    name: backend-tbc-ontap-nvme-secret
  clientCertificate: <client certificate>
```

StorageClass definition (save as `sc-nvme.yaml`):

```
# sc-nvme.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-nvme
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
  media: "ssd"
  provisioningType: "thin"
  fsType: ext4
  snapshots: "true"
allowVolumeExpansion: true
```

If you plan to use ReadWriteMany (RWX) volumes with the NVMe protocol, a single RWX NVMe volume cannot be mounted by more than 64 nodes in a Kubernetes cluster. For more information about this limitation and information about the Super-subsystem model introduced in Trident 26.02 and the Per-volume subsystem model prior to Trident 26.02, see the Trident documentation: https://docs.netapp.com/us-en/trident/trident-use/ontap-san-prep.html#nvme-tcp-considerations.

FC SAN

Create a TridentBackendConfig and StorageClass for ONTAP SAN with Fibre Channel to enable FC-based block storage provisioning. The backend configuration uses the ontap-san driver with the FCP protocol specified and includes credentials stored in a Kubernetes Secret.

Use the username/password authentication method or client certificate authentication method to authenticate with ONTAP. For details about how to generate the client certificate, private key and CA certificate, please review Trident documentation here. Examples for both methods are provided below.

Show example

Backend secret and backend configuration file using ONTAP username and password authentication (save as `tbc-fc.yaml`):

```
# tbc-fc.yaml
apiVersion: v1
kind: Secret
metadata:
  name: tbc-fc-secret
type: Opaque
stringData:
  username: <cluster admin username>
  password: <cluster admin password>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-fc
spec:
  version: 1
  storageDriverName: ontap-san
  managementLIF: <ONTAP management LIF>
  backendName: tbc-fc
  svm: openshift-fc #<replace with your SVM name>
  sanType: fcp
  storagePrefix: demofc #<replace with your prefix>
  defaults:
    nameTemplate: "{{ .config.StoragePrefix }}_{{ .volume.Namespace
}}_{{ .volume.RequestName }}"
  credentials:
    name: tbc-fc-secret
```

Backend secret and backend configuration file using client certificate authentication (save as `tbc-fc.yaml`):

```
# tbc-fc.yaml
apiVersion: v1
kind: Secret
metadata:
  name: tbc-fc-secret
type: Opaque
stringData:
  clientPrivateKey: <client private key>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-fc
spec:
  version: 1
  storageDriverName: ontap-san
  managementLIF: <ONTAP management LIF>
  backendName: tbc-fc
  svm: openshift-fc #<replace with your SVM name>
  sanType: fcp
  storagePrefix: demofc #<replace with your prefix>
  defaults:
    nameTemplate: "{{ .config.StoragePrefix }}_{{ .volume.Namespace
}}_{{ .volume.RequestName }}"
  credentials:
    name: tbc-fc-secret
  clientCertificate: <client certificate>
```

```
# sc-fc.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-fc
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
  media: "ssd"
  provisioningType: "thin"
  fsType: ext4
  snapshots: "true"
allowVolumeExpansion: true
```

iSCSI SAN economy

Create a TridentBackendConfig and StorageClass for ONTAP SAN economy with iSCSI to enable iSCSI-based block storage provisioning. The backend configuration uses the ontap-san-economy driver with the iSCSI protocol and includes credentials stored in a Kubernetes Secret.

Backend secret and backend configuration file using client certificate authentication (save as `tbc-san-eco.yaml`):

```
# tbc-san-eco.yaml
---
apiVersion: v1
kind: Secret
metadata:
  name: ontap-san-cert-auth
  namespace: trident
type: Opaque
stringData:
  clientPrivateKey: <client private key>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: tbc-san-eco-cert
  namespace: trident
spec:
  version: 1
  storageDriverName: ontap-san-economy
  managementLIF: "<ONTAP management LIF>"
  backendName: tbc_san_eco
  svm: <replace with your SVM name>
  storagePrefix: <replace with your prefix>
  credentials:
    name: ontap-san-cert-auth
  clientCertificate: <client certificate>
```

```
# sc-san-eco.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-san-eco
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san-economy"
  provisioningType: "thin"
  snapshots: "true"
```

Review the Trident documentation for additional samples of yaml files, and information on access modes and volume modes supported by SAN drivers and NAS drivers.

1. Samples using NAS drivers
2. Samples using SAN drivers

ROSA clusters with Amazon FSx for NetApp ONTAP

Create YAML files for each protocol you want to configure. You can configure one or both of the following protocols: NAS for NFS-based file storage or iSCSI for block storage.

NAS

Create a TridentBackendConfig and StorageClass for Amazon FSx for NetApp ONTAP with ONTAP NAS to enable NFS-based persistent storage provisioning on ROSA clusters. The backend configuration uses Amazon FSx for NetApp ONTAP DNS names for management and data LIFs, and includes credentials stored in a Kubernetes Secret in the trident namespace.

Backend secret and backend configuration file (save as `tbc-fsx-nas.yaml`):

```
# tbc-fsx-nas.yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-fsx-ontap-nas-secret
type: Opaque
stringData:
  username: <FSx for ONTAP, for example fsxadmin>
  password: <FSx for ONTAP password>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: backend-fsx-ontap-nas
spec:
  version: 1
  backendName: fsx-ontap
  storageDriverName: ontap-nas
  managementLIF: <Management DNS name>
  dataLIF: <NFS DNS name>
  svm: <SVM NAME>
  credentials:
    name: backend-fsx-ontap-nas-secret
```

StorageClass definition (save as `sc-fsx-nas.yaml`):

```
# sc-fsx-nas.yaml (storage class name is trident-csi)
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: trident-csi
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-nas"
  fsType: "ext4"
allowVolumeExpansion: true
reclaimPolicy: Retain
```

iSCSI

Create a TridentBackendConfig and StorageClass for Amazon FSx for NetApp ONTAP with ONTAP SAN to enable iSCSI-based block storage provisioning on ROSA clusters. The backend configuration uses the ontap-san driver and includes credentials stored in a Kubernetes Secret. Ensure that worker nodes are prepared for iSCSI access.

Backend secret and backend configuration file (save as `tbc-fsx-iscsi.yaml`):

```
# tbc-fsx-iscsi.yaml
apiVersion: v1
kind: Secret
metadata:
  name: backend-tbc-fsx-iscsi-secret
type: Opaque
stringData:
  username: <FSx for ONTAP, for example fsxadmin>
  password: <FSx for ONTAP password>
---
apiVersion: trident.netapp.io/v1
kind: TridentBackendConfig
metadata:
  name: fsx-iscsi
spec:
  version: 1
  storageDriverName: ontap-san
  managementLIF: <Management DNS name>
  backendName: fsx-iscsi
  svm: <SVM name>
  credentials:
    name: backend-tbc-fsx-iscsi-secret
```

StorageClass definition (save as `sc-fsx-iscsi.yaml`):

```
# sc-fsx-iscsi.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: sc-fsx-iscsi
provisioner: csi.trident.netapp.io
parameters:
  backendType: "ontap-san"
  media: "ssd"
  provisioningType: "thin"
  fsType: ext4
  snapshots: "true"
allowVolumeExpansion: true
```

Step 3: Create a VolumeSnapshotClass configuration file

Create a VolumeSnapshotClass definition for both on-premises and ROSA deployments. This configuration enables snapshot-based operations for persistent volumes.

VolumeSnapshotClass definition (save as `snapshot-class.yaml`):

```
# snapshot-class.yaml
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
  name: trident-snapshotclass
driver: csi.trident.netapp.io
deletionPolicy: Retain
```

Step 4: Apply the configuration files to your cluster

Apply the configuration files you created in the previous steps to your OpenShift cluster.

Steps

1. Apply the TridentBackendConfig and StorageClass files for each protocol you configured.

   For on-premises clusters:

```
oc create -f tbc-nas.yaml -n trident
oc create -f sc-nas.yaml

oc create -f tbc-nas-economy.yaml -n trident
oc create -f sc-nas-economy.yaml

oc create -f tbc-iscsi.yaml -n trident
oc create -f sc-iscsi.yaml

oc create -f tbc-nvme.yaml -n trident
oc create -f sc-nvme.yaml

oc create -f tbc-fc.yaml -n trident
oc create -f sc-fc.yaml
```

For ROSA clusters:

```
oc create -f tbc-fsx-nas.yaml -n trident
oc create -f sc-fsx-nas.yaml

oc create -f tbc-fsx-iscsi.yaml -n trident
oc create -f sc-fsx-iscsi.yaml
```

2. Apply the VolumeSnapshotClass configuration.

```
oc create -f snapshot-class.yaml
```

3. Verify that the resources were created successfully.

Check TridentBackendConfig objects:

```
oc get tbc -n trident
```

Check StorageClass objects:

```
oc get storageclass
```

Check VolumeSnapshotClass:

```
oc get volumesnapshotclass
```

Step 5: Set the default Trident storage and snapshot classes

Set the Trident StorageClass and VolumeSnapshotClass as the defaults in the OpenShift cluster. This is required for OpenShift Virtualization to create golden image sources for VM templates.

Steps

1. Set the default Trident StorageClass.

   Set a Trident-backed StorageClass as the cluster default so that PersistentVolumeClaims automatically use it when no storage class is specified. You need to configure two annotations: one for the cluster-wide default and one specific to OpenShift Virtualization.

   a. Set the cluster-wide default StorageClass annotation.

      Ensure that only one StorageClass is set as default. If another StorageClass is already set to default, set its annotation to false.

      In the console, edit the annotation:

      ```
      storageclass.kubernetes.io/is-default-class: "true"
      ```

      From the CLI:

      ```
      kubectl patch storageclass <storage class name> -p '{"metadata":
      {"annotations":{"storageclass.kubernetes.io/is-default-
      class":"true"}}}'
      ```

   b. Set the OpenShift Virtualization-specific default annotation.

      OpenShift Virtualization uses a specific annotation that takes precedence over the cluster's general `is-default-class` annotation. If another StorageClass is already set to default, set its annotation to false.

      In the console, edit the annotation:

      ```
      storageclass.kubevirt.io/is-default-virt-class: "true"
      ```

      From the CLI:

      ```
      kubectl patch storageclass <storage class name> -p '{"metadata":
      {"annotations":{"storageclass.kubevirt.io/is-default-virt-class":
      "true"}}}'
      ```

2. Set the default Trident VolumeSnapshotClass.

   Set a Trident-backed VolumeSnapshotClass as the cluster default to enable snapshot-based operations for persistent volumes. This ensures that VolumeSnapshots automatically use the Trident CSI driver when no

snapshot class is specified and allows OpenShift Virtualization to create snapshots of golden images.

Ensure that only one VolumeSnapshotClass is set as default. If another VolumeSnapshotClass is already set to default, set its annotation to false.

In the console, edit the annotation:

```
snapshot.storage.kubernetes.io/is-default-class: "true"
```

From the CLI:

```
oc patch volumesnapshotclass <snapshot class name> --type=merge -p
'{"metadata":{"annotations":{"snapshot.storage.kubernetes.io/is-default-
class":"true"}}}'
```

Requirements to deploy Red Hat OpenShift Virtualization with ONTAP

Review the prerequisites and additional details to deploy OpenShift virtualization with ONTAP storage systems.

Prerequisites

- A Red Hat OpenShift cluster (later than version 4.6) installed on bare-metal infrastructure with RHCOS worker nodes
- Deploy Machine Health Checks to maintain HA for VMs
- A NetApp ONTAP cluster, with SVM configured with the correct protocol.
- Trident installed on the OpenShift cluster
- A Trident backend configuration created
- A StorageClass configured on the OpenShift cluster with Trident as the provisioner
- A Trident VolumeSnapshotClass configured on the OpenShift cluster with Trident as the provisioner
- Trident StorageClass configured as default for the OpenShift cluster or OpenShift Virtualization.

For the above Trident prerequisites, see Trident installation section for details.

- Cluster-admin access to Red Hat OpenShift cluster
- Admin access to NetApp ONTAP cluster
- An admin workstation with tridentctl and oc tools installed and added to $PATH

Because OpenShift Virtualization is managed by an operator installed on the OpenShift cluster, it imposes additional overhead on memory, CPU, and storage, which must be accounted for while planning the hardware requirements for the cluster. See the documentation here for more details.

Optionally, you can also specify a subset of the OpenShift cluster nodes to host the OpenShift Virtualization operators, controllers, and VMs by configuring node placement rules. To configure node placement rules for

OpenShift Virtualization, follow the documentation here.

For the storage backing OpenShift Virtualization, NetApp recommends having a dedicated StorageClass that requests storage from a particular Trident backend, which in turn is backed by a dedicated SVM. This maintains a level of multitenancy with regard to the data being served for VM-based workloads on the OpenShift cluster.

NetApp Trident CSI, Storage Profiles, volume access modes for OpenShift Virtualization

In OpenShift Virtualization, boot sources are disk images used to provision virtual machines (VMs) from templates. They are typically stored as DataVolumes or Persistent Volume Claims (PVCs) within the openshift-virtualization-os-images namespace. In order for OpenShift Virtualization to function correctly, there are two storage configuration tasks that are mandatory, prior to installing the OpenShift Virtualization using the operator:

1. You must configure a default StorageClass for the cluster. Otherwise, OpenShift Virtualization cannot automatically import boot source images.
2. You must configure storage profiles if your storage provider is not recognized by CDI. A storage profile provides recommended storage settings based on the associated StorageClass. One of the critical functions of StorageProfile is defining cloning strategies for a StorageClass. This determines how volumes are cloned (e.g., using snapshots, CSI clones, or host-assisted copies), optimizing performance and resource usage based on the capabilities of the underlying storage vendor.
   Trident CSI driver is recognized by CDI, and a default storage profile is automatically created for any Trident StorageClass created in the cluster. The cloning strategy for this default storage profile is set to snapshot, which is the recommended cloning strategy for Trident CSI driver. You must create the VolumeSnapshotClass and StorageClass for Trident and set the StorageClass as default before installing OpenShift Virtualization. This ensures that the boot source images are available as VolumeSnapshots in the openshift-virtualization-os-images namespace, to be able to provision VMs efficiently without any errors.

For certain supported operating systems (like RHEL, Fedora, or CentOS), OpenShift Virtualization automatically provides and updates boot sources. These are downloaded when OpenShift Virtualization is installed and they are managed and updated to the latest OS version automatically. If an OS is not automatically provided, an administrator must manually prepare and attach a custom image.
When you use NetApp Trident as the CSI, you get the following additional benefits:

1. The boot source images will be available as snapshots in the openshift-virtualization-os-images namespace. When VMs are provisioned, the boot image is cloned from the snapshot, thus making the VM boot much faster.
2. A storage profile is automatically created for each Trident StorageClass created and is automatically configured with clone strategy set to snapshot. Access Mode is set to ReadWriteMany. There is no need for manual configuration.

It is very critical that you create a Trident StorageClass and Trident VolumeSnapshotClass and set them as default prior to installing the OpenShift Virtualization in your cluster. This will ensure that the boot source images will be available as VolumeSnapshots in openshift-virtualization-os-images namespace, to be able to provision VMs efficiently without any errors.

Another advantage of using Trident CSI:
. In OpenShift Virtualization, the boot disks of the VMs need to have RWX access modes to do Live migration of VMs. Trident supports **ReadWriteMany** access mode for volumes provisioned using NAS protocols as well as using SAN protocols (when volumeMode is set to Block and not FileSystem). You can use either protocol for the boot disk. The Storage profile object will be automatically configured based on the StorageClass settings

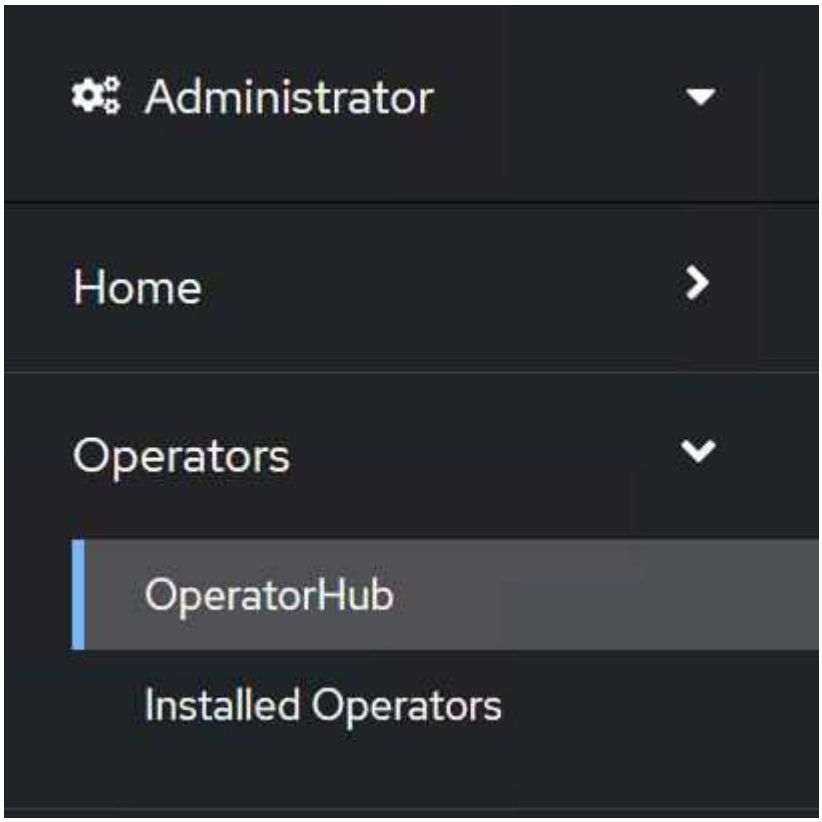
and the boot disk access modes will be appropriately set when provisioning the VMs.

After you have set the Trident StorageClass as default for the cluster, you can create additional StorageClasses for Trident with different parameters and they will be automatically recognized by OpenShift Virtualization with appropriate storage profiles created for them. You can then specify the StorageClass to be used for the VM's boot disk and data disks in the VM definition.

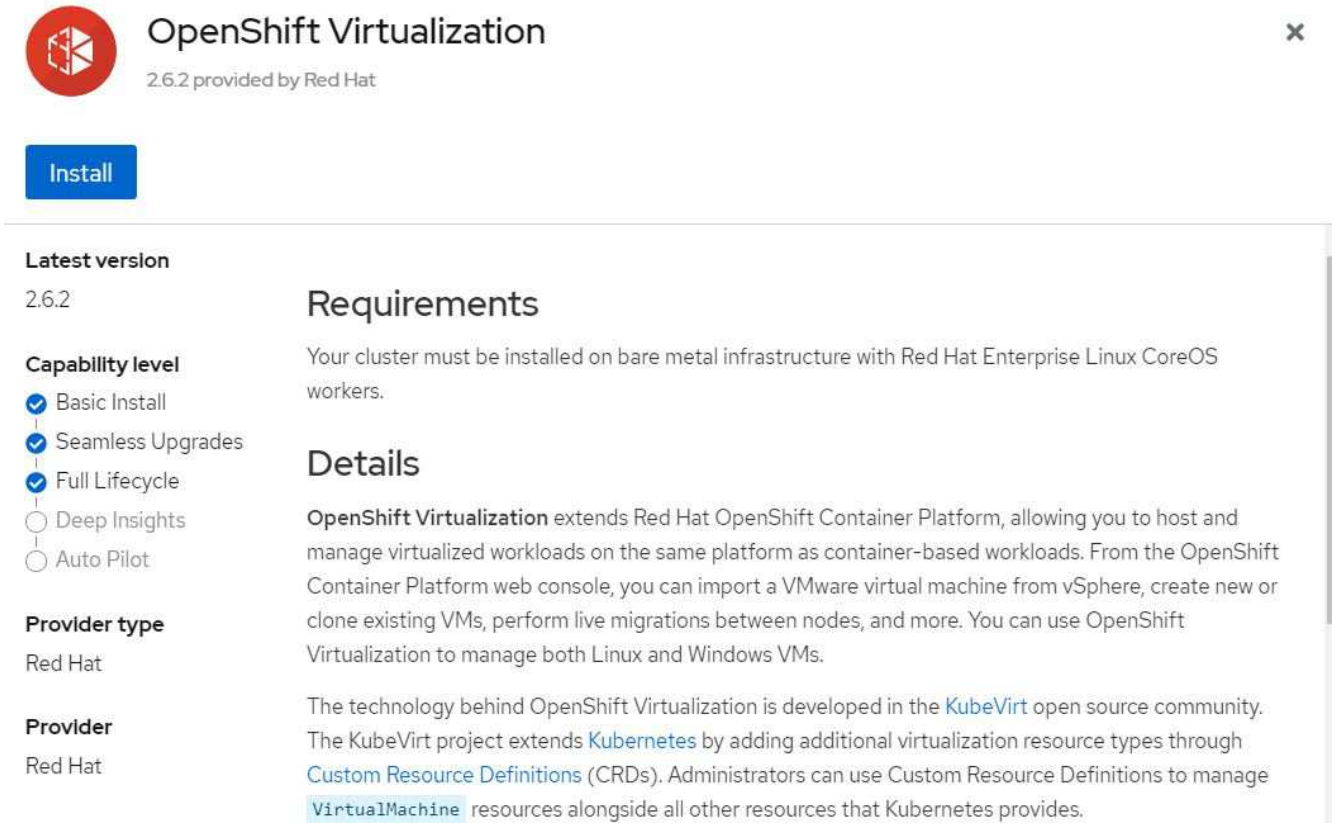
Installing OpenShift Virtualization on the OpenShift cluster is a straightforward process using the OpenShift Virtualization operator available in the OpenShift OperatorHub. For detailed steps on how to install OpenShift Virtualization, see the documentation here.

Install OpenShift Virtualization on a Red Hat OpenShift bare-metal cluster. This procedure includes logging in with cluster-admin access, navigating to the OperatorHub, and installing the OpenShift Virtualization operator.

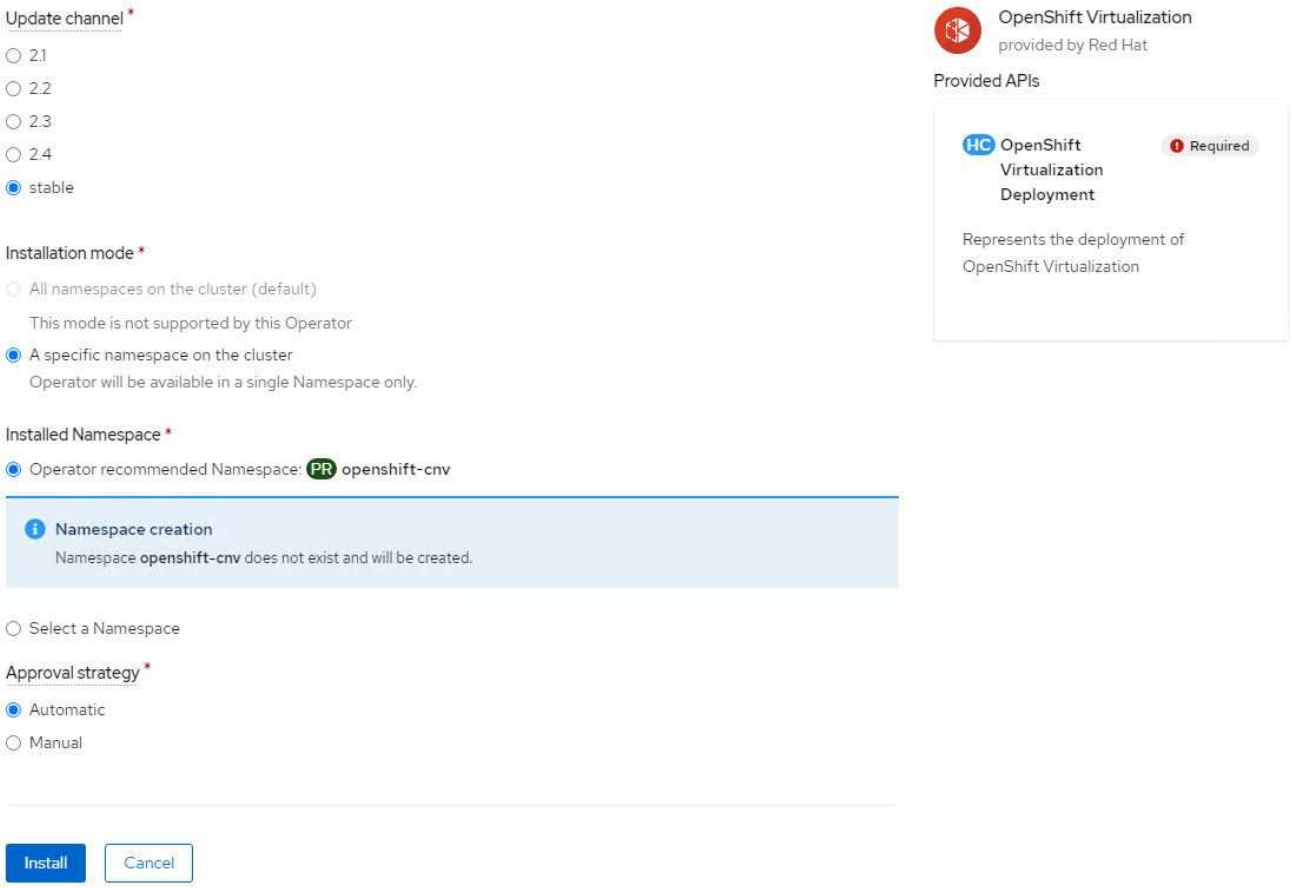
1. Log into the Red Hat OpenShift bare-metal cluster with cluster-admin access.
2. Select Administrator from the Perspective drop down.
3. Navigate to Operators > OperatorHub and search for OpenShift Virtualization.



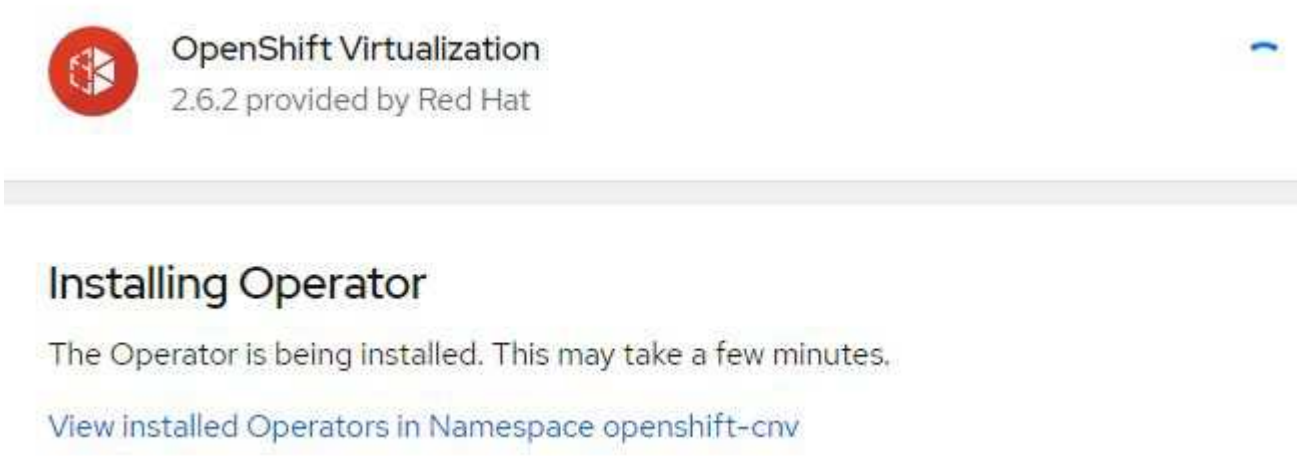
4. Select the OpenShift Virtualization tile and click Install.



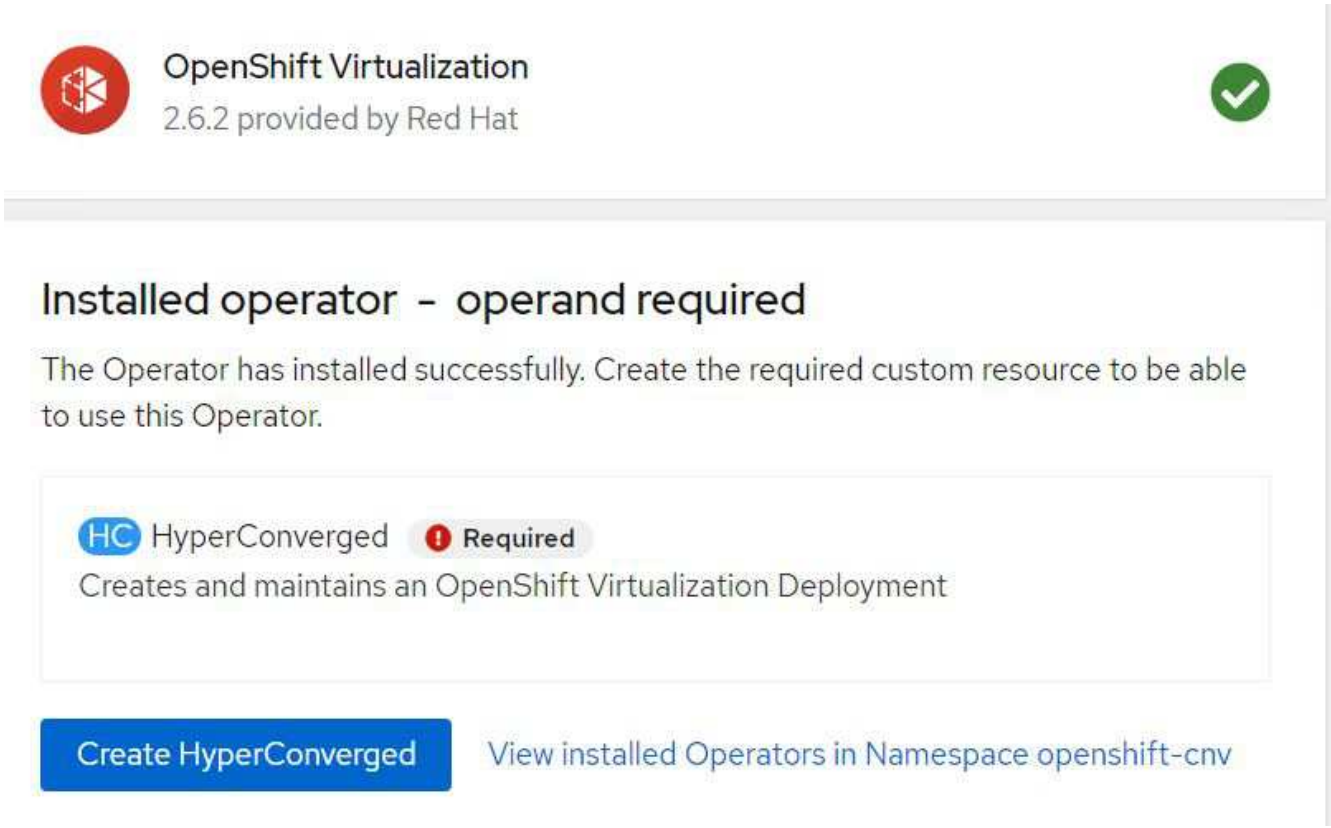
5. On the Install Operator screen, leave all default parameters and click Install.



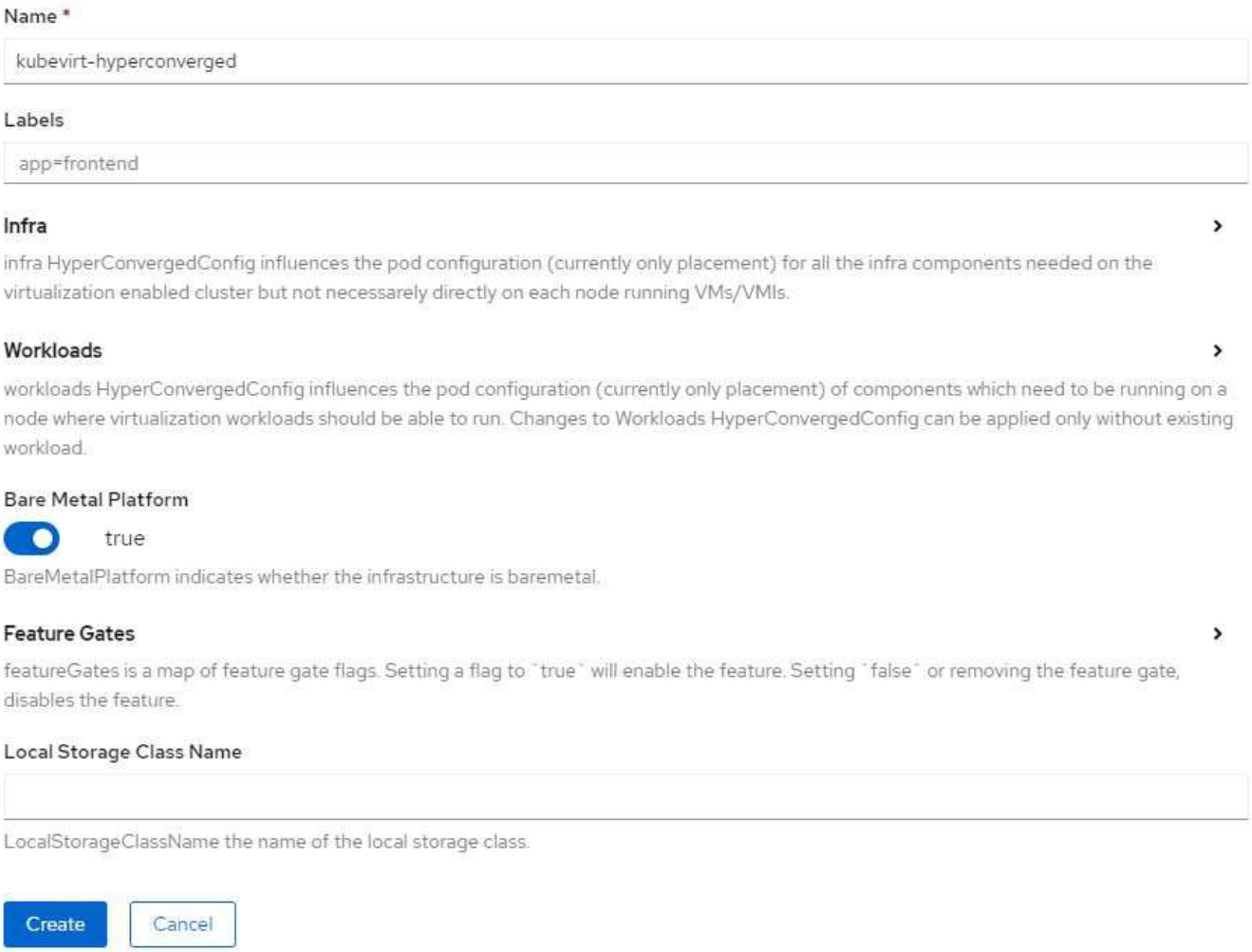
6. Wait for the operator installation to complete.



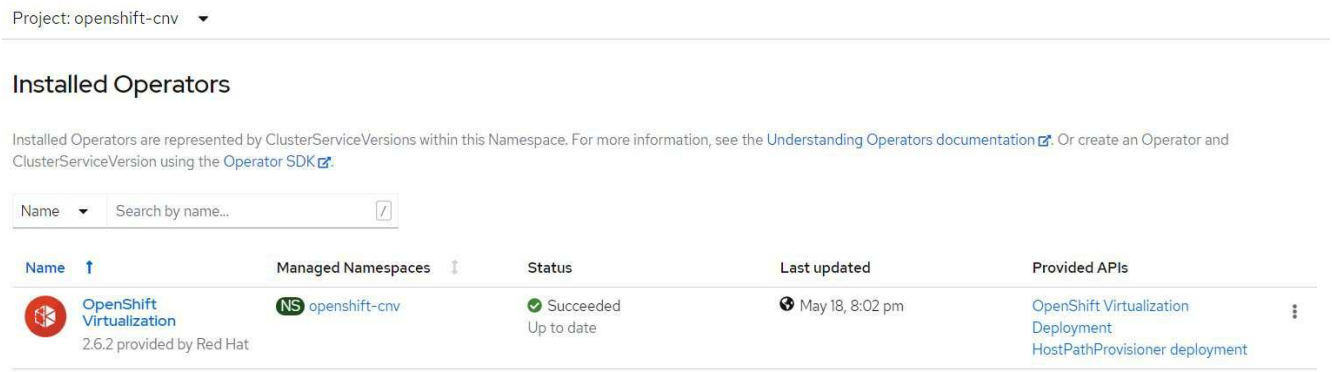
7. After the operator has installed, click Create HyperConverged.



8. On the Create HyperConverged screen, click Create, accepting all default parameters. This step starts the installation of OpenShift Virtualization.



9. After all the pods move to the Running state in the openshift-cnv namespace and the OpenShift Virtualization operator is in the Succeeded state, the operator is ready to use. VMs can now be created on the OpenShift cluster.



Create a VM using ONTAP storage with Red Hat OpenShift Virtualization

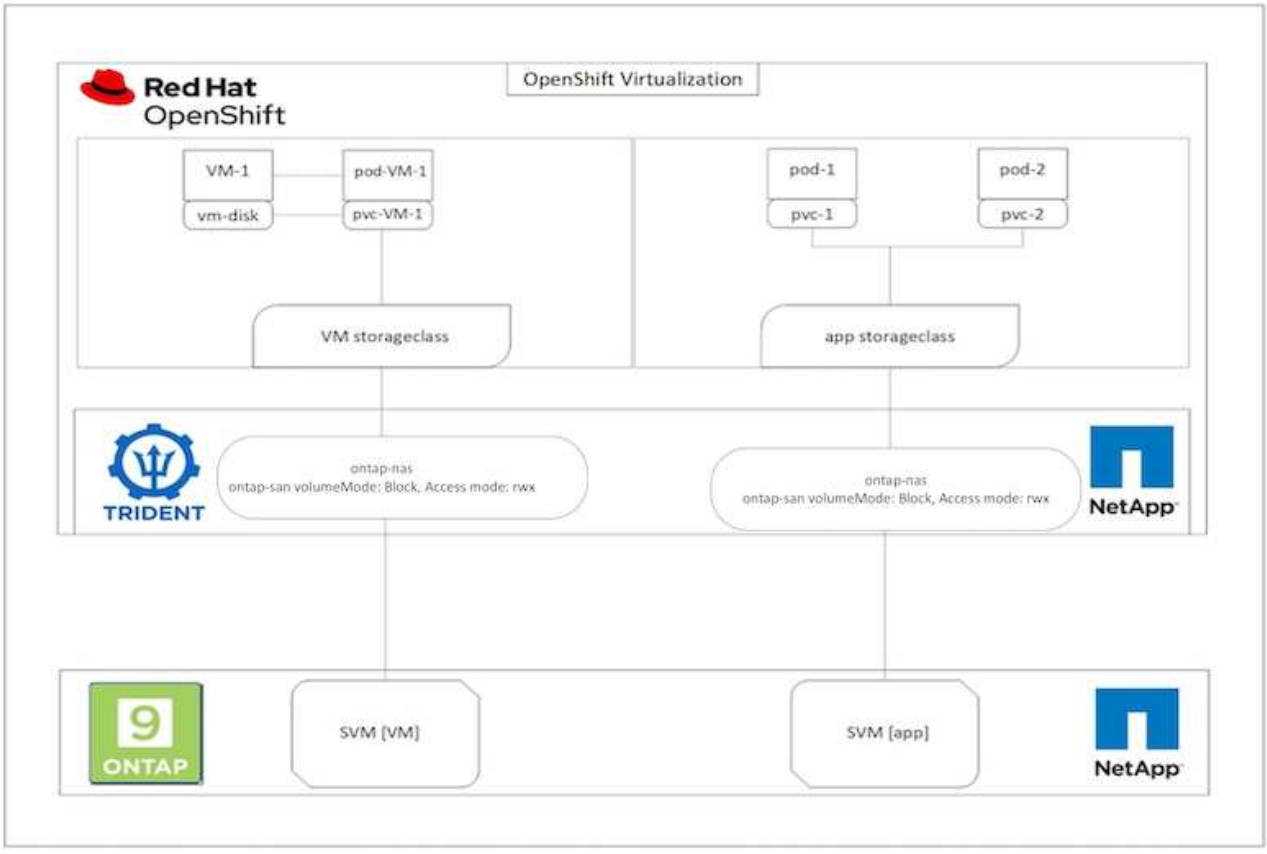
Create a VM with OpenShift Virtualization. This procedure includes selecting an operating system template, configuring StorageClasses, and customizing VM parameters to meet specific requirements.

As a prerequisite, you should have already created the Trident backend, the StorageClass and the volume snapshot class objects. You can refer to the Trident installation section for details.

Create VM

VMs are stateful deployments that require volumes to host the operating system and data. With CNV, because the VMs are run as pods, the VMs are backed by PVs hosted on NetApp ONTAP through Trident. These volumes are attached as disks and store the entire filesystem including the boot source of the VM.

Trident StorageClass must be set as default StorageClass on the cluster to ensure that the PVCs created for the VM disks are provisioned using the Trident backend. You can have multiple StorageClasses on the cluster, but the default StorageClass must be used for provisioning the PVC for VM boot disks. Additional disks can use other StorageClasses if required.



To quickly create a virtual machine on the OpenShift cluster, complete the following steps:

1. Navigate to Virtualization > Virtual Machines and click Create.
2. Select From template.
3. Select the desired operating system for which the boot source is available.
4. Check the checkbox Start the VirtualMachine after creation.
5. Click Quick create VirtualMachine.

The virtual machine is created and started and comes to the **Running** state. It automatically creates a PVC and a corresponding PV for the boot disk using the default StorageClass. In order to be able to live migrate the VM in the future, you must ensure that the StorageClass used for the disks can support RWX volumes. This is a requirement for live migration. ontap-nas and ontap-san (volumeMode block for iSCSI and NVMe/TCP protocols) can support RWX access modes for the volumes created using the respective StorageClasses.

To configure ontap-san StorageClass on the cluster see the Section for configuring StorageClass.

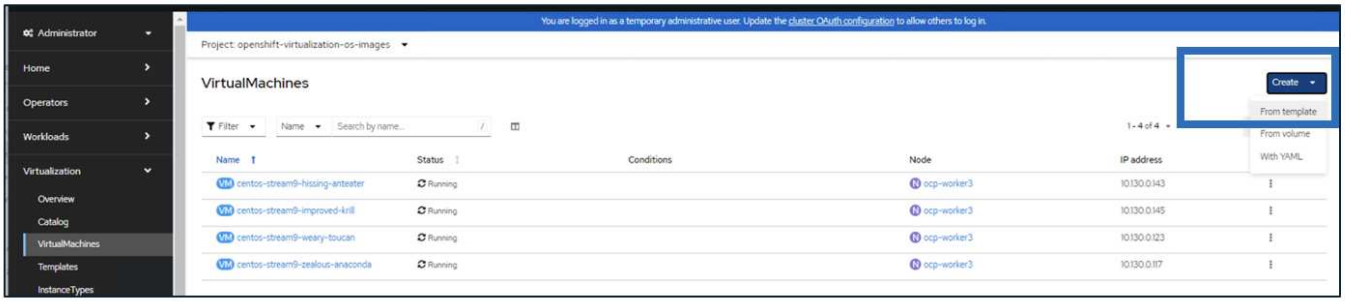
Clicking on Quick create VirtualMachine will use the default StorageClass to create the PVC and PV for the bootable root disk for the VM. You can select a different StorageClass for the disk, by selecting Customize VirtualMachine > Customize VirtualMachine parameters > Disks and then editing the disk to use the required StorageClass.

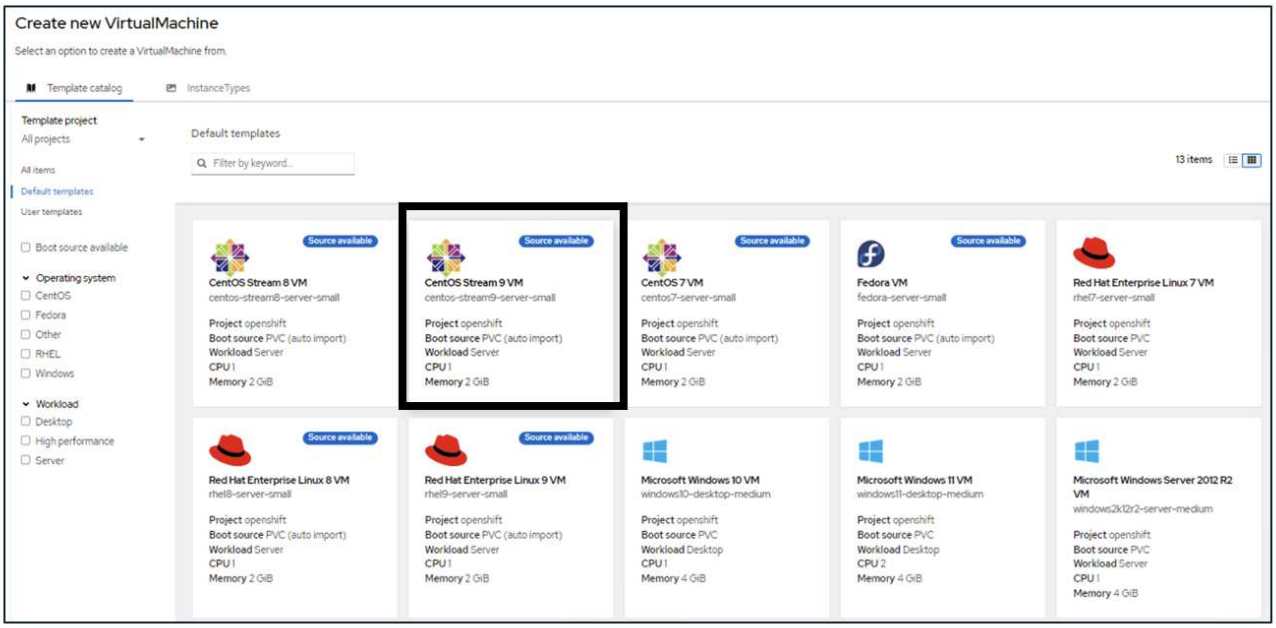
Typically block access mode is preferred compared to file systems while provisioning the VM disks.

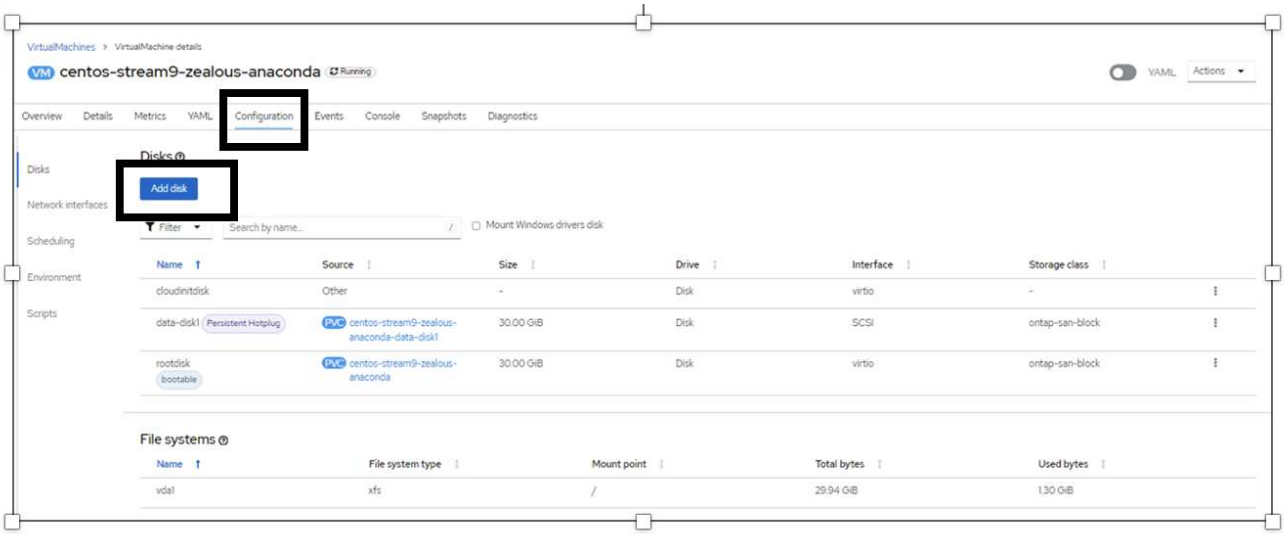
To customize the virtual machine creation after you have selected the OS template, click on Customize VirtualMachine instead of Quick create.

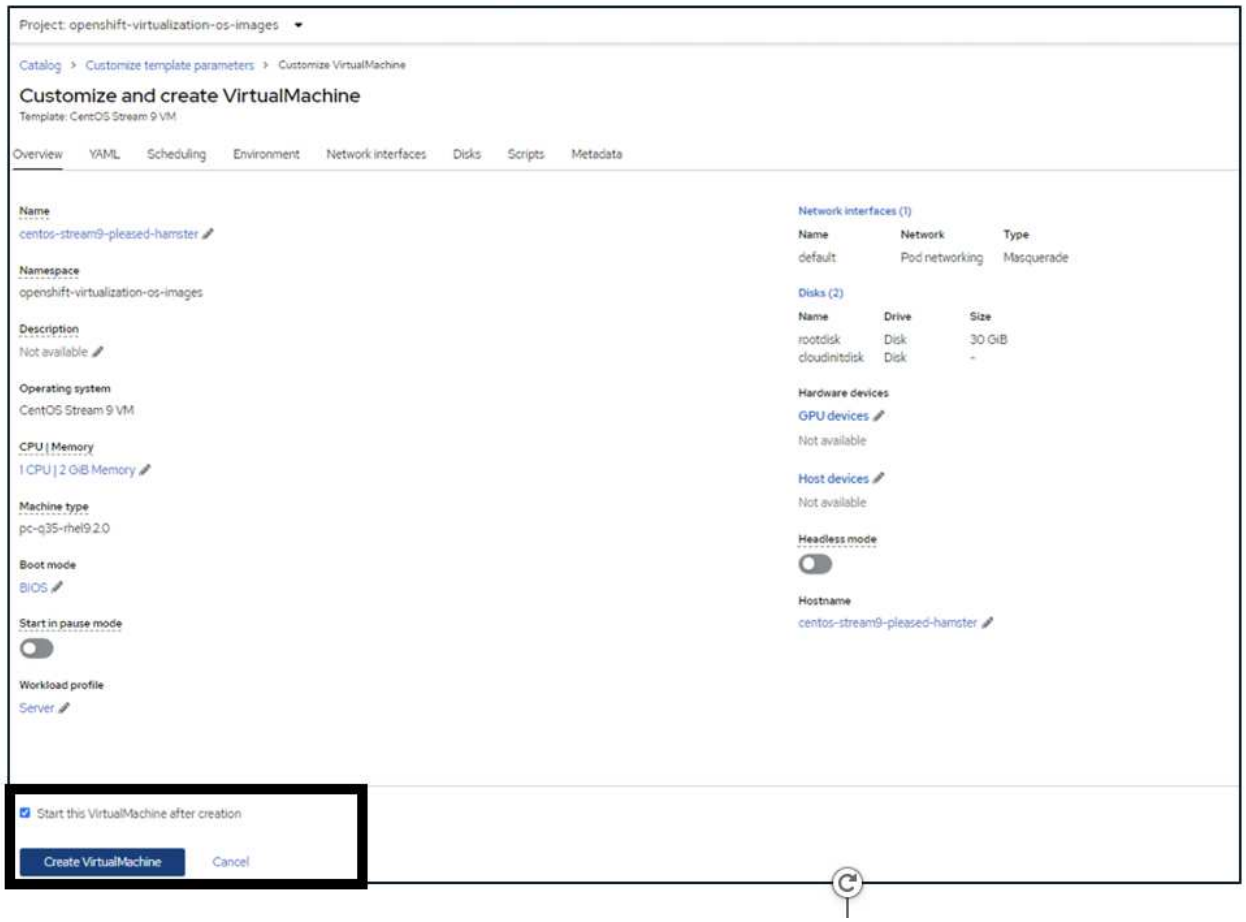
1. If the selected operating system has boot source configured, you can click on **Customize VirtualMachine parameters**.
2. If the selected operating system has no boot source configured, you must configure it. You can see details about the procedures shown in the documentation.
3. After Configuring the boot disk, you can click on **Customize VirtualMachine parameters**.
4. You can customize the VM from the tabs on this page. For eg. click on the **Disks** tab and then click on **Add disk** to add another disk to the VM.
5. Click Create Virtual Machine to create the virtual machine; this spins up a corresponding pod in the background.

When a boot source is configured for a template or an operating system from an URL or from a registry, it creates a PVC in the `openshift-virtualization-os-images` project and downloads the KVM guest image to the PVC. You must make sure that template PVCs have enough provisioned space to accommodate the KVM guest image for the corresponding OS. These PVCs are then cloned and attached as rootdisk to virtual machines when they are created using the respective templates in any project.




Create new VirtualMachine
Select an option to create a VirtualMachine from.
Template catalog
InstanceTypes
Template project
All projects
All items
Default templates
User templates
Boot source available
Operating system
CentOS
Fedora
Other
RHEL
Windows
Workload
Desktop
High performance
Server
Default templates
Filter by keyword...
13 items
Source available
CentOS Stream 8 VM
centos-stream8-server-small
Project openshift
Boot source PVC (auto import)
Workload Server
CPU 1
Memory 2 GiB
CentOS Stream 9 VM
centos-stream9-server-small
Project openshift
Boot source PVC (auto import)
Workload Server
CPU 1
Memory 2 GiB
CentOS 7 VM
centos7-server-small
Project openshift
Boot source PVC (auto import)
Workload Server
CPU 1
Memory 2 GiB
Fedora VM
fedora-server-small
Project openshift
Boot source PVC (auto import)
Workload Server
CPU 1
Memory 2 GiB
Red Hat Enterprise Linux 7 VM
rhel7-server-small
Project openshift
Boot source PVC
Workload Server
CPU 1
Memory 2 GiB
Red Hat Enterprise Linux 8 VM
rhel8-server-small
Project openshift
Boot source PVC (auto import)
Workload Server
CPU 1
Memory 2 GiB
Red Hat Enterprise Linux 9 VM
rhel9-server-small
Project openshift
Boot source PVC (auto import)
Workload Server
CPU 1
Memory 2 GiB
Microsoft Windows 10 VM
windows10-desktop-medium
Project openshift
Boot source PVC
Workload Desktop
CPU 1
Memory 4 GiB
Microsoft Windows 11 VM
windows11-desktop-medium
Project openshift
Boot source PVC
Workload Desktop
CPU 2
Memory 4 GiB
Microsoft Windows Server 2012 R2 VM
windows2k12r2-server-medium
Project openshift
Boot source PVC
Workload Server
CPU 1
Memory 4 GiB


VirtualMachines > VirtualMachine details
VM centos-stream9-zealous-anaconda Running
VAML
Actions
Overview
Details
Metrics
YAML
Configuration
Events
Console
Snapshots
Diagnostics
Disks
Network interfaces
Scheduling
Environment
Scripts
Disks
Add disk
Filter
Search by name...
Mount Windows drivers disk
Name
Source
Size
Drive
Interface
Storage class
cloudinitdisk
Other
-
Disk
virtio
-
data-disk1
Persistent Hotplug
PVC centos-stream9-zealous-anaconda-data-disk1
30.00 GiB
Disk
SCSI
ontap-san-block
rootdisk
bootable
PVC centos-stream9-zealous-anaconda
30.00 GiB
Disk
virtio
ontap-san-block
File systems
Name
File system type
Mount point
Total bytes
Used bytes
vda1
xfs
/
29.94 GiB
1.30 GiB



Video Demonstration

The following video shows a demonstration of creating a VM in OpenShift Virtualization using iSCSI storage.

Create a VM in OpenShift Virtualization using Block Storage

Create a VM with Red Hat OpenShift Virtualization using ONTAP storage through alternative methods

In addition to the method of creating a VM using the OpenShift Virtualization UI, there are other methods to create VMs on OpenShift Virtualization. This section provides an overview of some of those methods.

Create a VM from a YAML file with Red Hat OpenShift Virtualization

A VM can also be created by applying a YAML file with the `oc apply` command. This method is useful for users who prefer using CLI or want to automate VM creation using scripts. To create a VM using this method, you can create a YAML file that defines the VM and its associated resources, such as PVCs for the VM disks. Then, you can apply the YAML file using the `oc apply -f <yaml-file>` command to create the VM on the OpenShift cluster.

Show example

```
apiVersion: kubevirt.io/v1
kind: VirtualMachine
metadata:
  name: test-vms-vm
  namespace: test-5vms
  labels:
    app: test-vms-vm
    vm.kubevirt.io/template: centos-stream9-server-small
spec:
  dataVolumeTemplates:
    - apiVersion: cdi.kubevirt.io/v1beta1
      kind: DataVolume
      metadata:
        name: test-vms-vm-rootdisk
      spec:
        sourceRef:
          kind: DataSource
          name: centos-stream9
          namespace: openshift-virtualization-os-images
        storage:
          resources:
            requests:
              storage: 30Gi
  runStrategy: Always
  template:
    metadata:
      labels:
        kubevirt.io/domain: test-vms-vm
    spec:
      domain:
        cpu:
          cores: 1
          sockets: 1
          threads: 1
        devices:
          disks:
            - bootOrder: 1
              disk:
                bus: virtio
              name: rootdisk
            - disk:
                bus: virtio
              name: cloudinitdisk
          interfaces:
```

```
        - masquerade: {}
          model: virtio
          name: default
      resources:
        requests:
          memory: 2Gi
    networks:
      - name: default
        pod: {}
    terminationGracePeriodSeconds: 180
    volumes:
      - dataVolume:
          name: test-vms-vm-rootdisk
        name: rootdisk
      - cloudInitNoCloud:
          userData: |-
            #cloud-config
            user: admin
            password: <replace with your password>
            chpasswd: { expire: False }
        name: cloudinitdisk
```

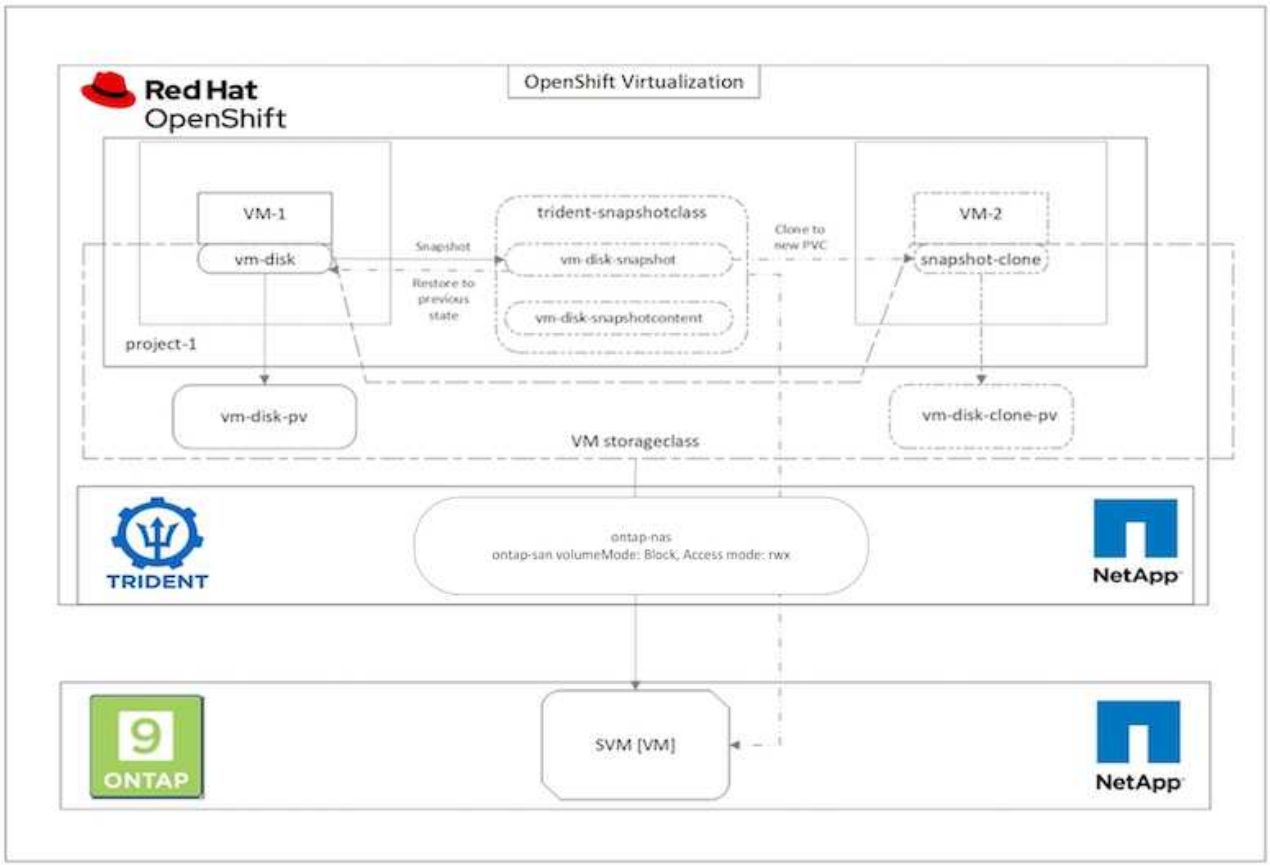
Create a VM from a snapshot copy with Red Hat OpenShift Virtualization

You can create a VM from a snapshot with OpenShift Virtualization. This procedure includes creating a VolumeSnapshotClass, taking a snapshot of the VM's persistent volume claim (PVC), restoring the snapshot to a new PVC, and deploying a new VM that uses the restored PVC as the root disk.

With Trident and Red Hat OpenShift, users can take a snapshot of a persistent volume on Storage Classes provisioned by it. With this feature, users can take a point-in-time copy of a volume and use it to create a new volume or restore the same volume back to a previous state. This enables or supports a variety of use-cases, from rollback to clones to data restore.

For Snapshot operations in OpenShift, the resources VolumeSnapshotClass, VolumeSnapshot, and VolumeSnapshotContent must be defined.

- A VolumeSnapshotContent is the actual snapshot taken from a volume in the cluster. It is cluster-wide resource analogous to PersistentVolume for storage.
- A VolumeSnapshot is a request for creating the snapshot of a volume. It is analogous to a PersistentVolumeClaim.
- VolumeSnapshotClass lets the administrator specify different attributes for a VolumeSnapshot. It allows you to have different attributes for different snapshots taken from the same volume.


Red Hat OpenShift
OpenShift Virtualization
VM-1
vm-disk
Snapshot
Restore to previous state
trident-snapshotclass
vm-disk-snapshot
vm-disk-snapshotcontent
Clone to new PVC
VM-2
snapshot-clone
project-1
vm-disk-pv
vm-disk-clone-pv
VM storageclass
TRIDENT
ontap-nas
ontap-san volumeMode: Block, Access mode: rwx
NetApp
9
ONTAP
SVM [VM]
NetApp

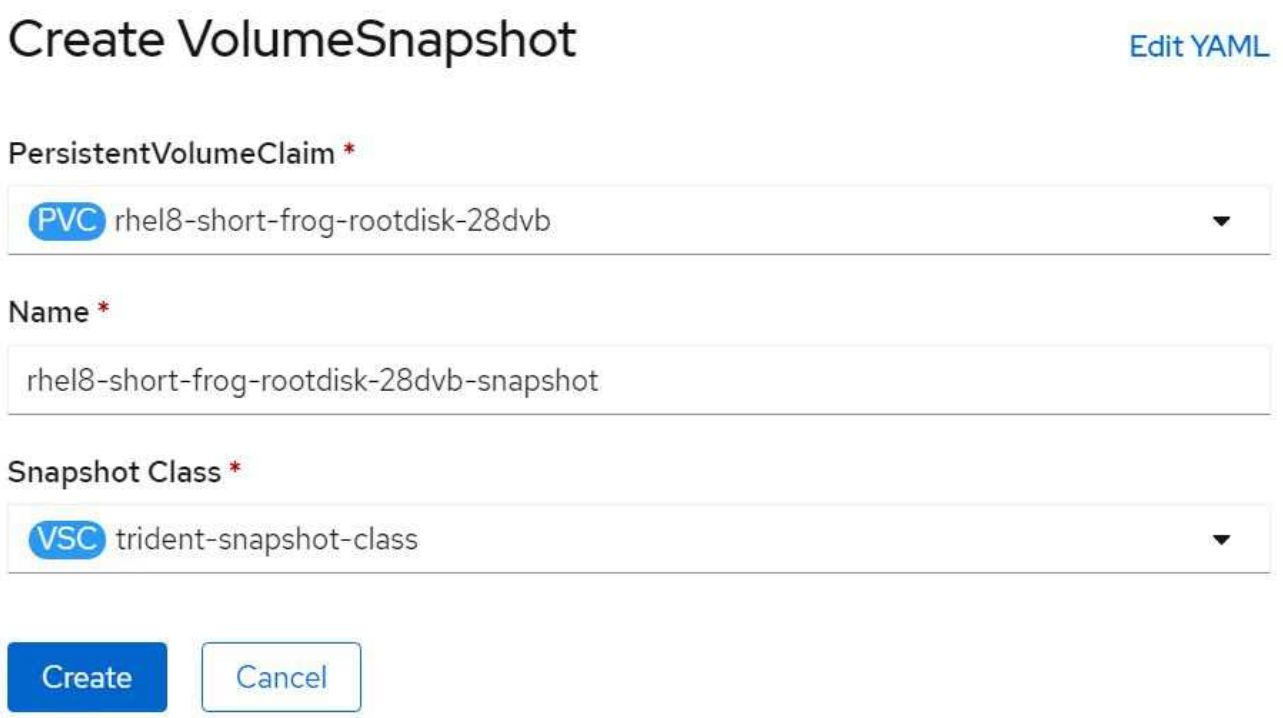
Show example

Prerequisite

You must have an existing VM with a PVC attached to it that is provisioned on a Trident Storage Class, and the VM must be in a running state before you can take a snapshot of the PVC.
You must have a VolumeSnapshotClass created that can then be used to create a VolumeSnapshot. This would have been created as part of the prerequisites for installing OpenShift Virtualization. You can refer to the Step 3 to create a VolumeSnapshotClass configuration for details.

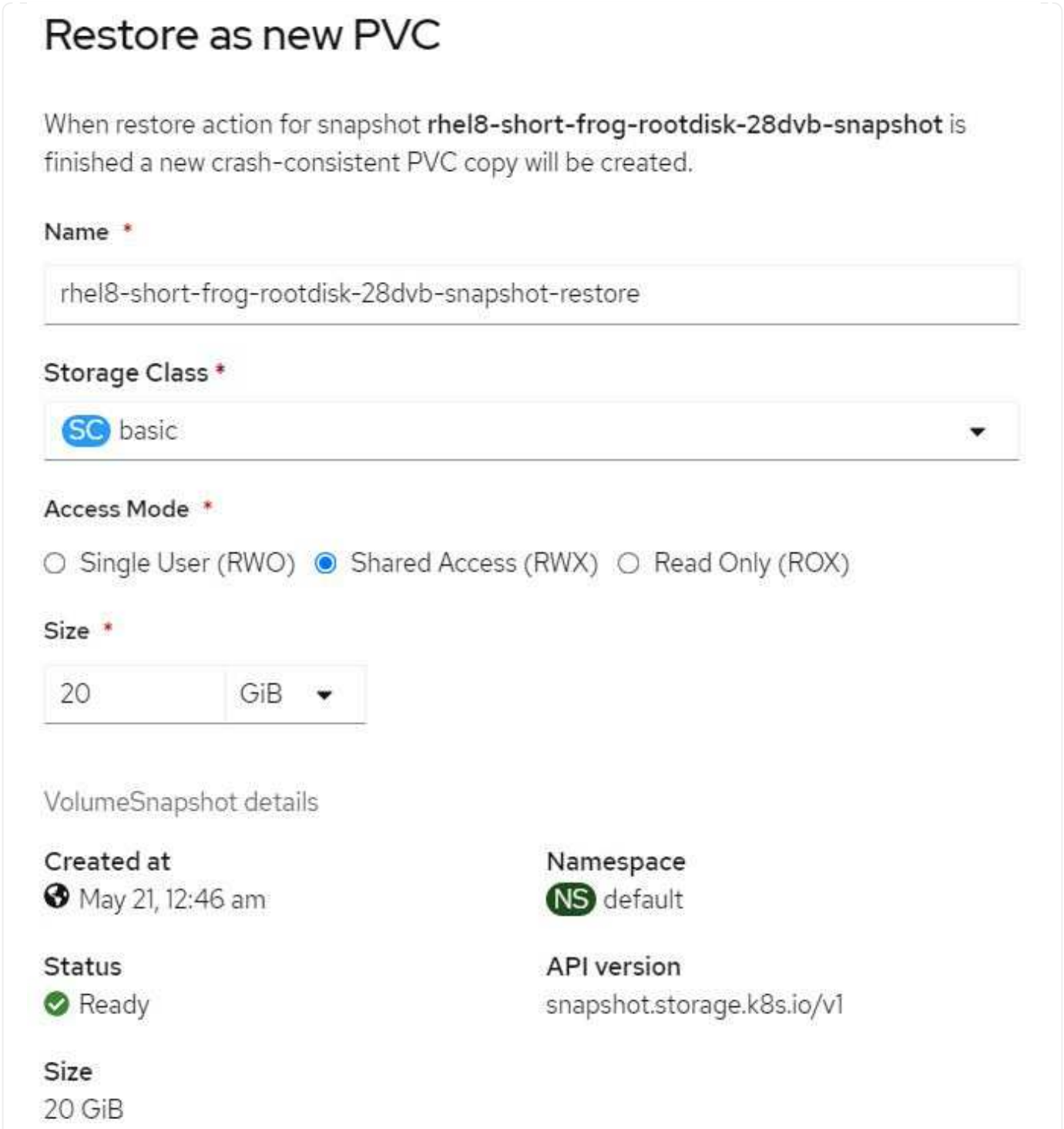
To create Snapshot of a VM, complete the following steps:

1. Identify the PVC that is attached to the source VM and then create a Snapshot of that PVC. Navigate to `Storage > VolumeSnapshots` and click Create VolumeSnapshots.
2. Select the PVC that you want to create the Snapshot for, enter the name of the Snapshot or accept the default, and select the appropriate VolumeSnapshotClass. Then click Create.
3. This creates the snapshot of the PVC at that point in time.



Create a new VM from the snapshot

1. First, restore the Snapshot into a new PVC. Navigate to Storage > VolumeSnapshots, click the ellipsis next to the Snapshot that you wish to restore, and click Restore as new PVC.
2. Enter the details of the new PVC and click Restore. This creates a new PVC.



1. Next, create a new VM from this PVC. Navigate to Virtualization > Virtual Machines and click Create > With YAML.
2. In the spec > template > spec > volumes section, specify the new PVC created from Snapshot instead of from the container disk. Provide all other details for the new VM according to your requirements.

```
- name: rootdisk
  persistentVolumeClaim:
    claimName: rhel8-short-frog-rootdisk-28dvb-snapshot-restore
```

1. Click Create to create the new VM.
2. After the VM is created successfully, access and verify that the new VM has the same state as that of the VM whose PVC was used to create the snapshot at the time when the snapshot was created.

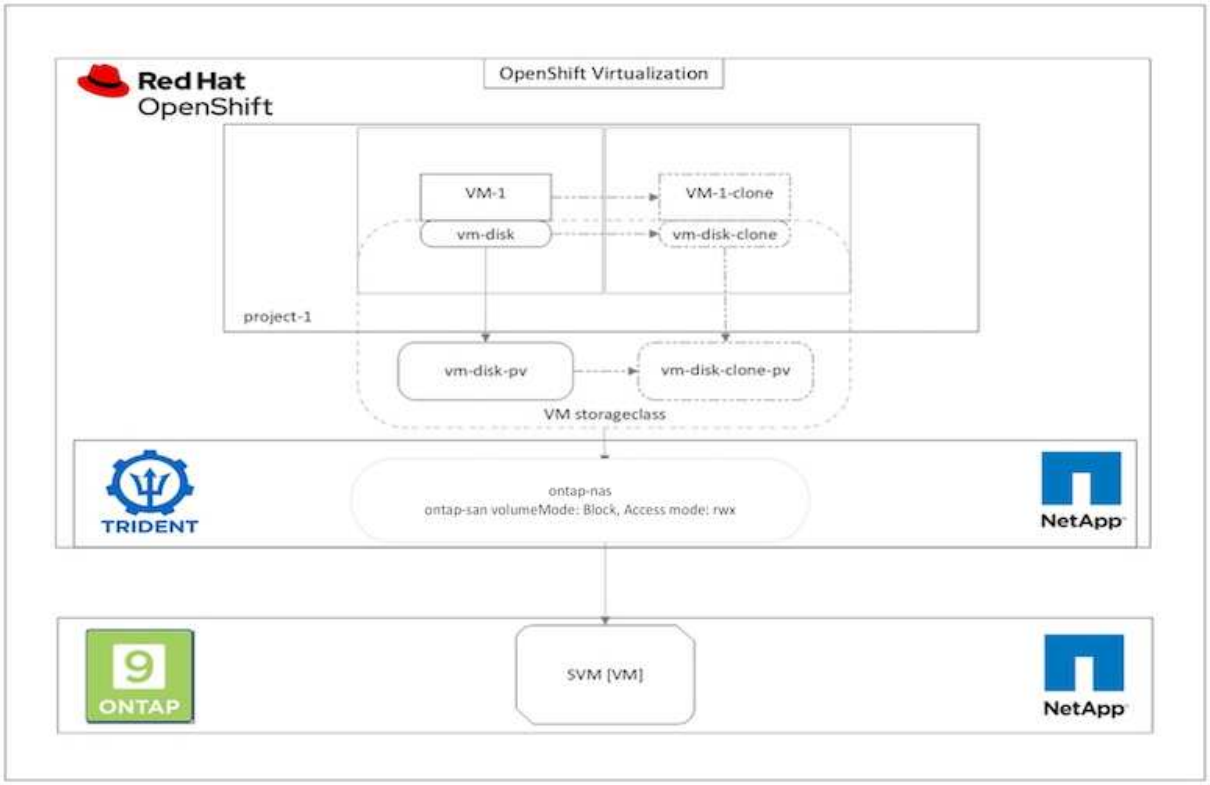
Clone a VM in OpenShift Virtualization using Trident

This procedure includes leveraging Trident CSI volume cloning, allowing you to create a new VM by shutting down the source VM or keep it running.

Show example

VM cloning

Cloning an existing VM in OpenShift is achieved with the support of Trident's Volume CSI cloning feature. CSI volume cloning allows for creation of a new PVC using an existing PVC as the data source by duplicating its PV. After the new PVC is created, it functions as a separate entity and without any link to or dependency on the source PVC.



There are certain restrictions with CSI volume cloning to consider:

1. Source PVC and destination PVC must be in the same project.
2. Cloning is supported within the same storage class.
3. Cloning can be performed only when source and destination volumes use the same VolumeMode setting; for example, a block volume can only be cloned to another block volume.

VMs in an OpenShift cluster can be cloned in two ways:

1. By shutting down the source VM
2. By keeping the source VM live

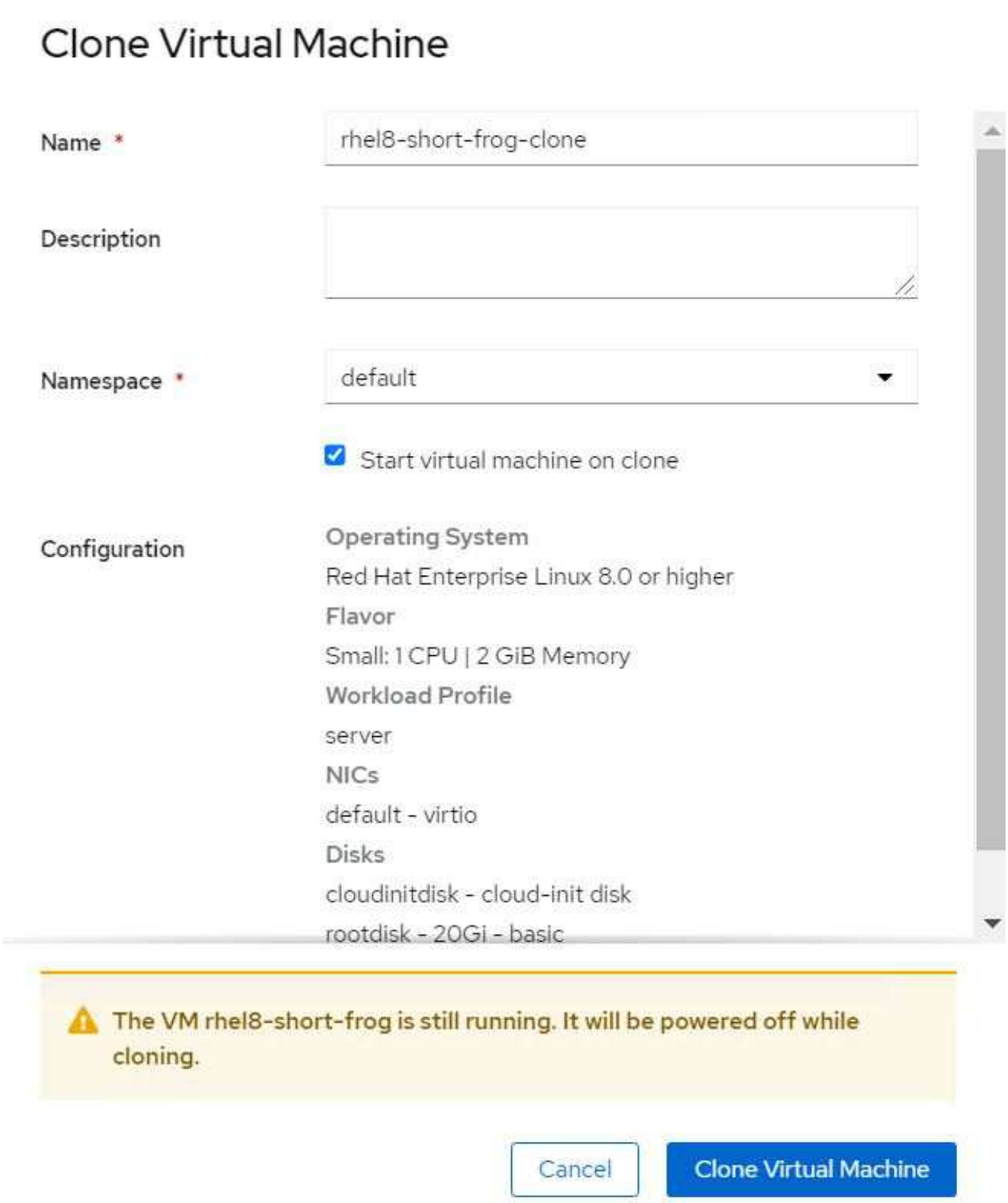
By Shutting down the source VM

Cloning an existing VM by shutting down the VM is a native OpenShift feature that is implemented with support from Trident. Complete the following steps to clone a VM.

1. Navigate to Workloads > Virtualization > Virtual Machines and click the ellipsis next to the virtual

machine you wish to clone.

2. Click Clone Virtual Machine and provide the details for the new VM.

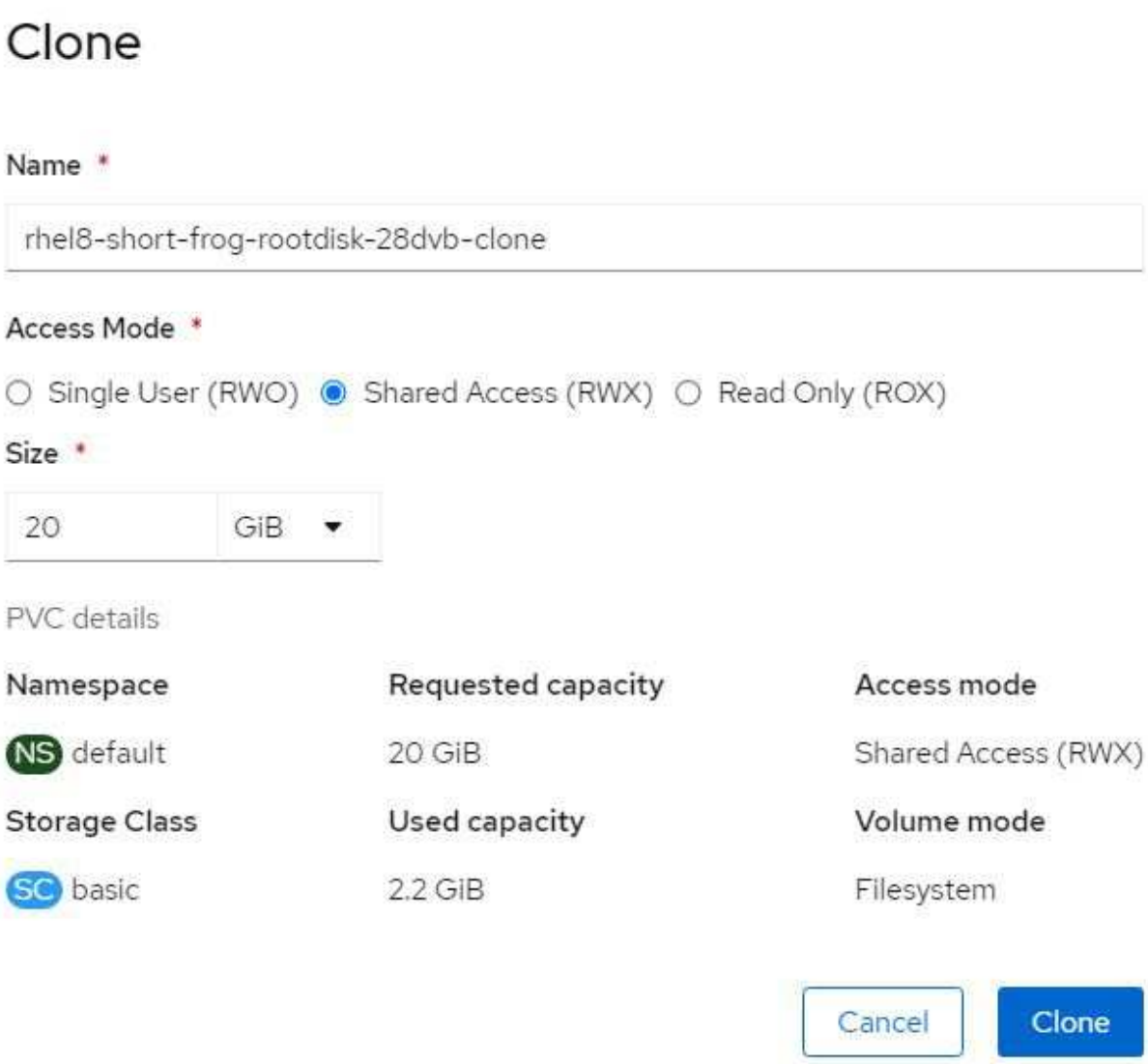


3. Click Clone Virtual Machine; this shuts down the source VM and initiates the creation of the clone VM.
4. After this step is completed, you can access and verify the content of the cloned VM.

By keeping the source VM live

An existing VM can also be cloned by cloning the existing PVC of the source VM and then creating a new VM using the cloned PVC. This method does not require you to shut down the source VM. Complete the following steps to clone a VM without shutting it down.

1. Navigate to Storage > PersistentVolumeClaims and click the ellipsis next to the PVC that is attached to the source VM.
2. Click Clone PVC and furnish the details for the new PVC.



3. Then click Clone. This creates a PVC for the new VM.
4. Navigate to Workloads > Virtualization > Virtual Machines and click Create > With YAML.
5. In the spec > template > spec > volumes section, attach the cloned PVC instead of the container disk. Provide all other details for the new VM according to your requirements.

```
- name: rootdisk
  persistentVolumeClaim:
    claimName: rhel8-short-frog-rootdisk-28dvb-clone
```

1. Click Create to create the new VM.
2. After the VM is created successfully, access and verify that the new VM is a clone of the source VM.

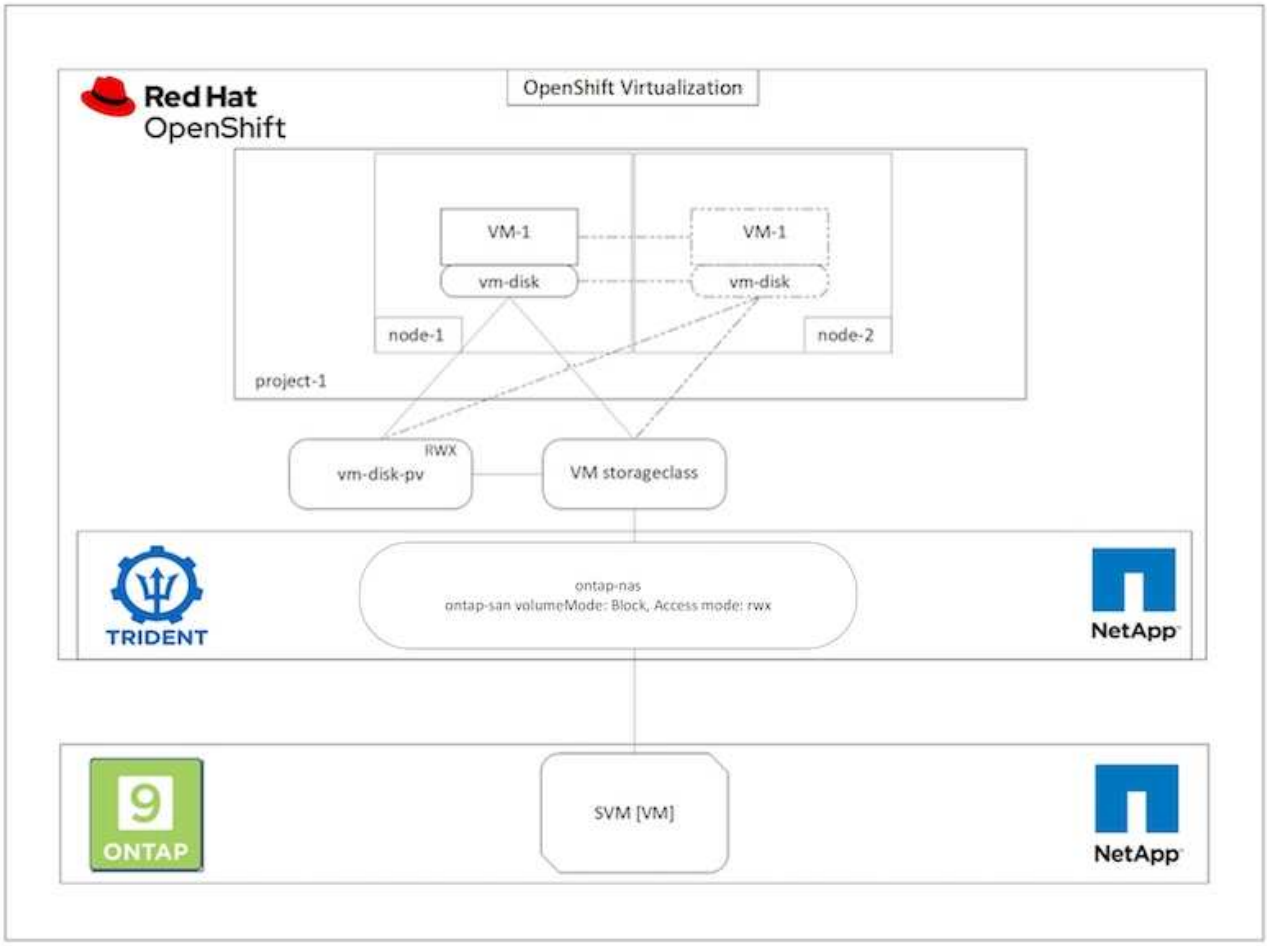
Migrate a VM between two nodes in a Red Hat OpenShift cluster

Migrate a VM in OpenShift Virtualization between two nodes in the cluster with no downtime. This procedure includes confirming the disks use RWX-compatible storage classes, initiating the migration, and monitoring the progress.

VM Live Migration

Live Migration is a process of migrating a VM instance from one node to another in an OpenShift cluster with no downtime. For live migration to work in an OpenShift cluster, VMs must be bound to PVCs with shared ReadWriteMany access mode. Trident backends configured using ontap-nas drivers support RWX access mode for FileSystem protocols nfs and smb. Refer to the documentation here. Trident backends configured using ontap-san drivers support RWX access mode for block volumeMode for iSCSI and NVMe/TCP protocols. Refer to the documentation here.

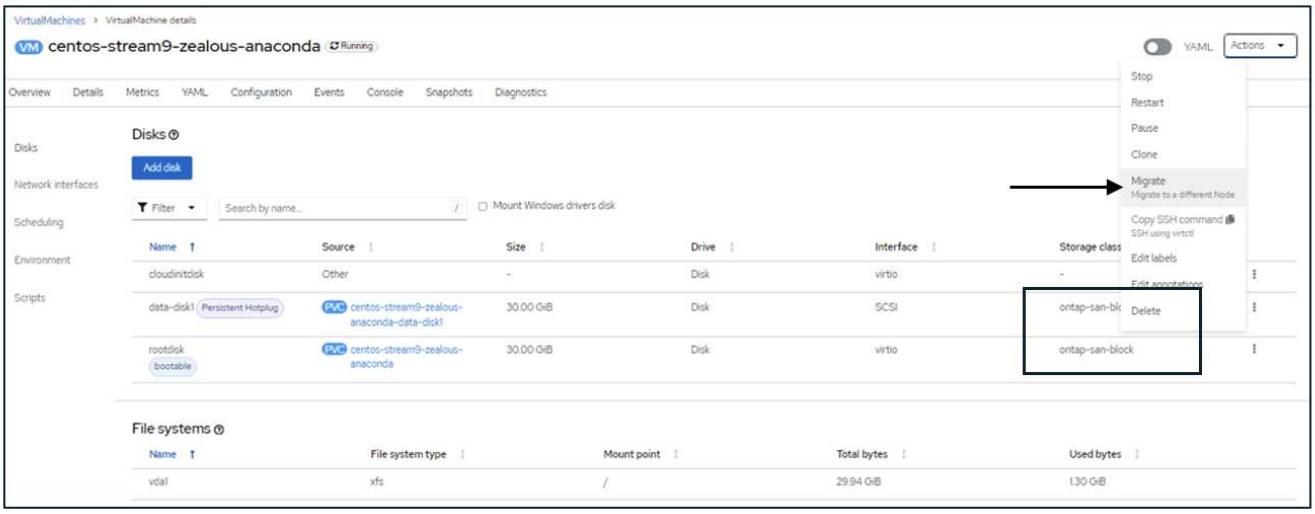
Therefore, for live migration to succeed, the VMs must be provisioned with disks (boot disks and additional hot plug disks) with PVCs using ontap-nas or ontap-san (volumeMode: Block) storage classes. When the PVCs are created, Trident creates ONTAP volumes in an SVM which is NFS-enabled or iSCSI enabled.

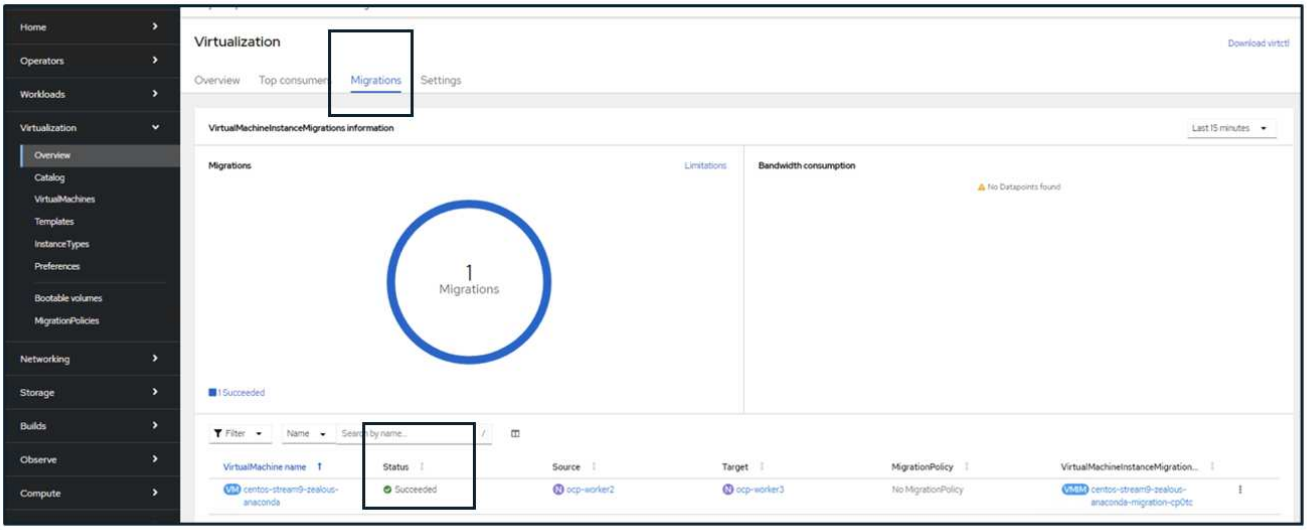


To perform a live migration of a VM that has been created previously and is in a Running state perform the following steps:

1. Select the VM that you want to live-migrate.
2. Click on **Configuration** tab.
3. Ensure that all the disks of the VM are created using Storage classes that can support RWX access mode.
4. Click on **Actions** on the right corner and then select **Migrate**.
5. To look at the progression of the Migration, go to Virtualization > Overview on the left hand side menu and then click on the **Migrations** tab.
   The Migration of the VM will transition from **Pending** to **Scheduling** to **Succeeded**

A VM instance in an OpenShift cluster automatically migrates to another node when the original node is placed into maintenance mode if the evictionStrategy is set to LiveMigrate.





Storage migration of VMs between storage classes in OpenShift Virtualization

In OpenShift Virtualization, you can migrate a virtual machine's (VM) disks to a different storage class directly through the web console without deleting the VM. This process allows you to move either the entire VM storage or specific volumes.

The following steps outline how to migrate a VM's storage class:

1. Ensure that the target storage class is properly configured and available in your OpenShift cluster. This includes having the necessary storage backend (e.g., NetApp ONTAP) set up and accessible.

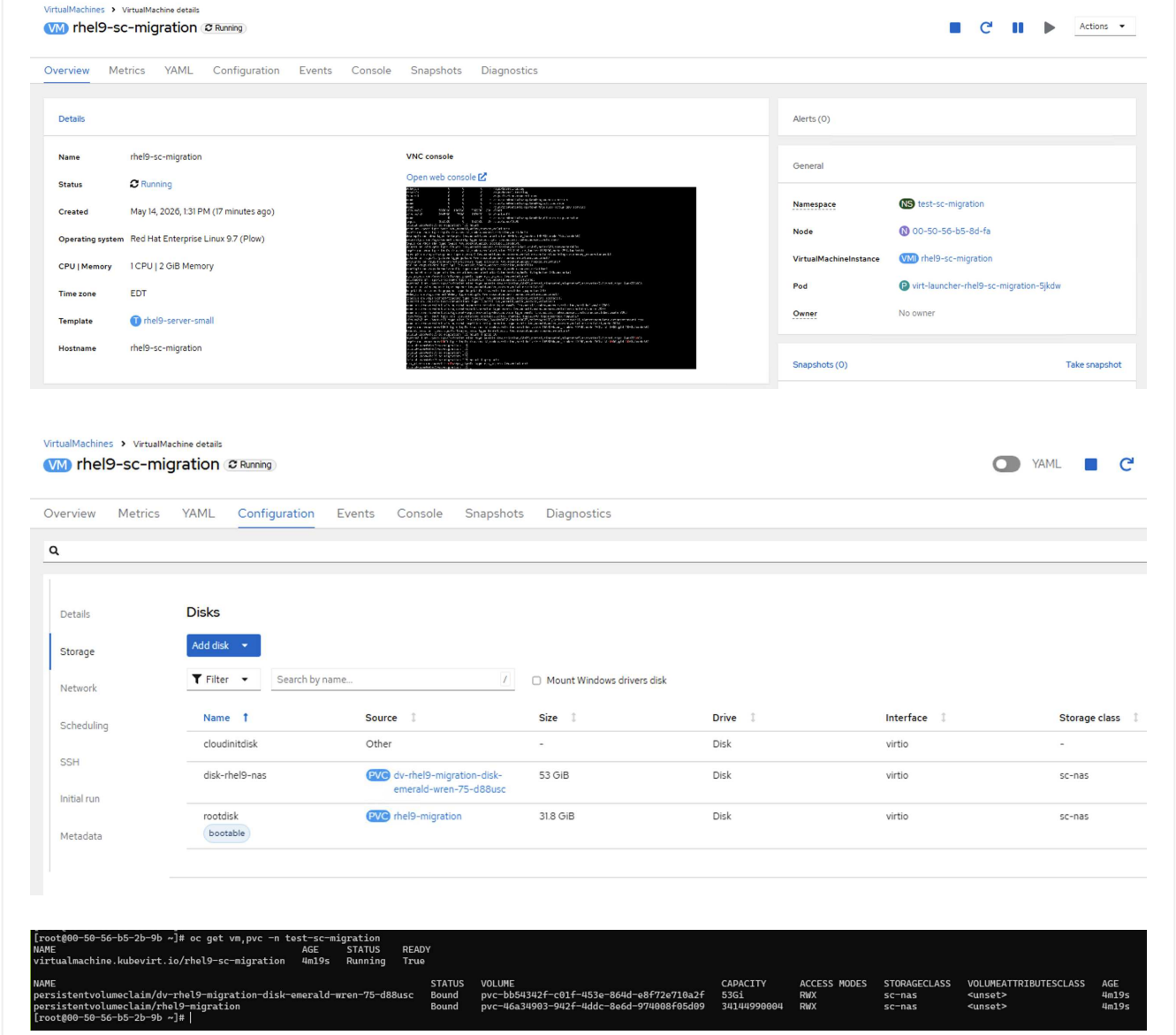
Show example

```
[root@00-50-56-b5-2b-9b ~]# oc get tbe -n trident
NAME         BACKEND        BACKEND UUID
tbe-k2vqg    tbc-san        3e9dc8ae-3b29-44c8-99a9-affdeaa6b4cc
tbe-mtp5n    tbc-nas-eco    a59e8053-bb00-4410-9b12-1dc4d283928b
tbe-nwqvq    tbc-nas-50     8b49f02f-3bc3-4720-b60f-bf90c3600008
tbe-wtp9h    tbc_san_eco    61745526-04cd-4acb-8f1b-daa9aa79f05a
tbe-zfp46    tbc-nas        259a1c2a-0a84-488d-ab62-ccea45499e09
```

```
[root@00-50-56-b5-2b-9b ~]# oc get sc
NAME              PROVISIONER             RECLAIMPOLICY   VOLUMEBINDINGMODE   ALLOWVOLUMEEXPANSION   AGE
sc-nas (default)  csi.trident.netapp.io   Delete          Immediate           true                   35d
sc-nas-50         csi.trident.netapp.io   Delete          Immediate           true                   22d
sc-nas-economy    csi.trident.netapp.io   Delete          Immediate           true                   20d
sc-san            csi.trident.netapp.io   Delete          Immediate           false                  21d
```

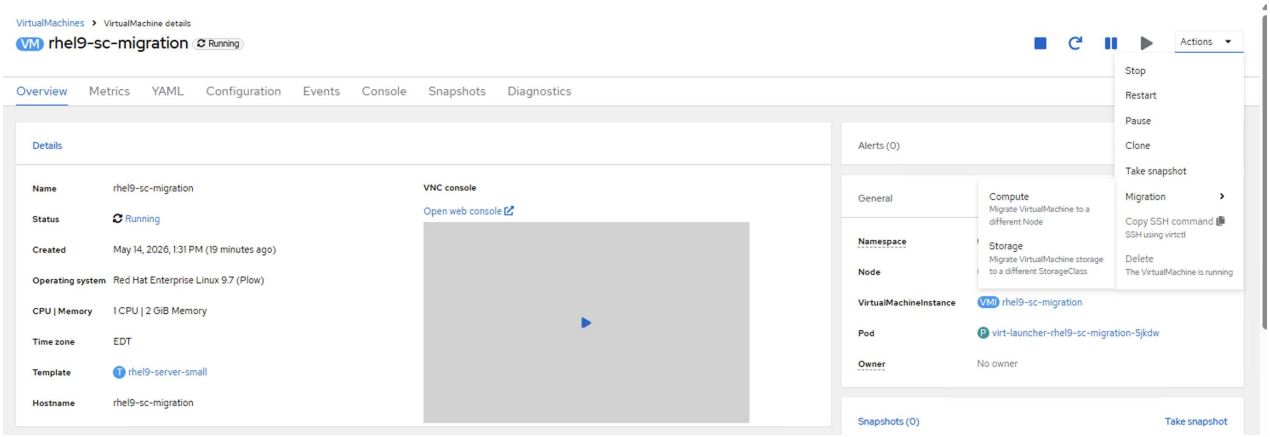
2. In the OpenShift Container Platform web console, navigate to the Virtualization perspective and select the VM you want to migrate.

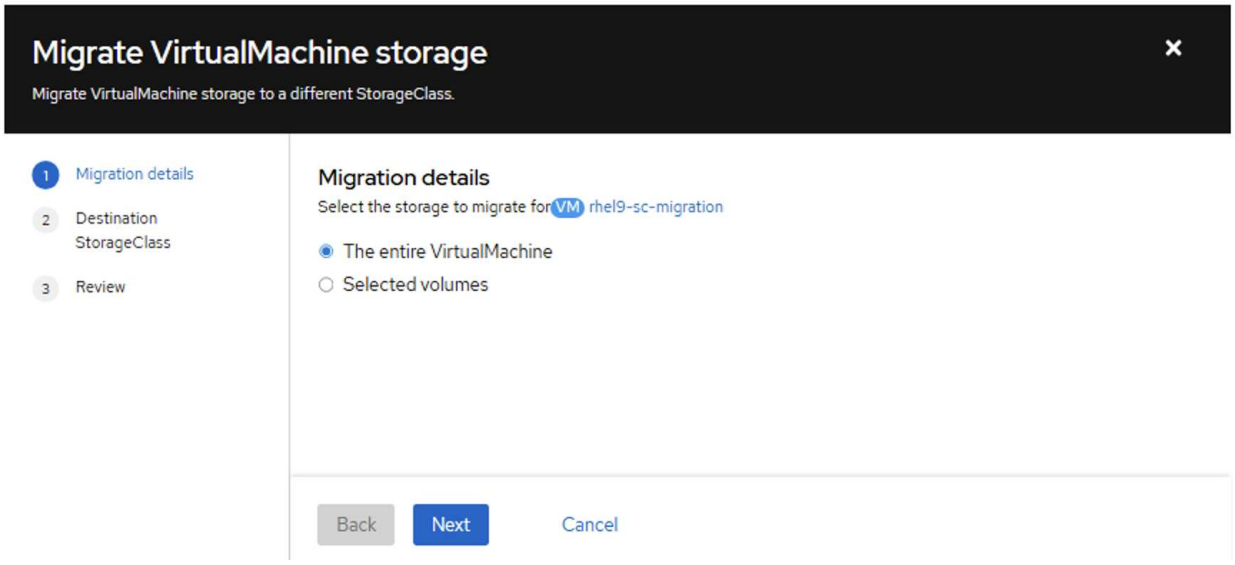
Show example

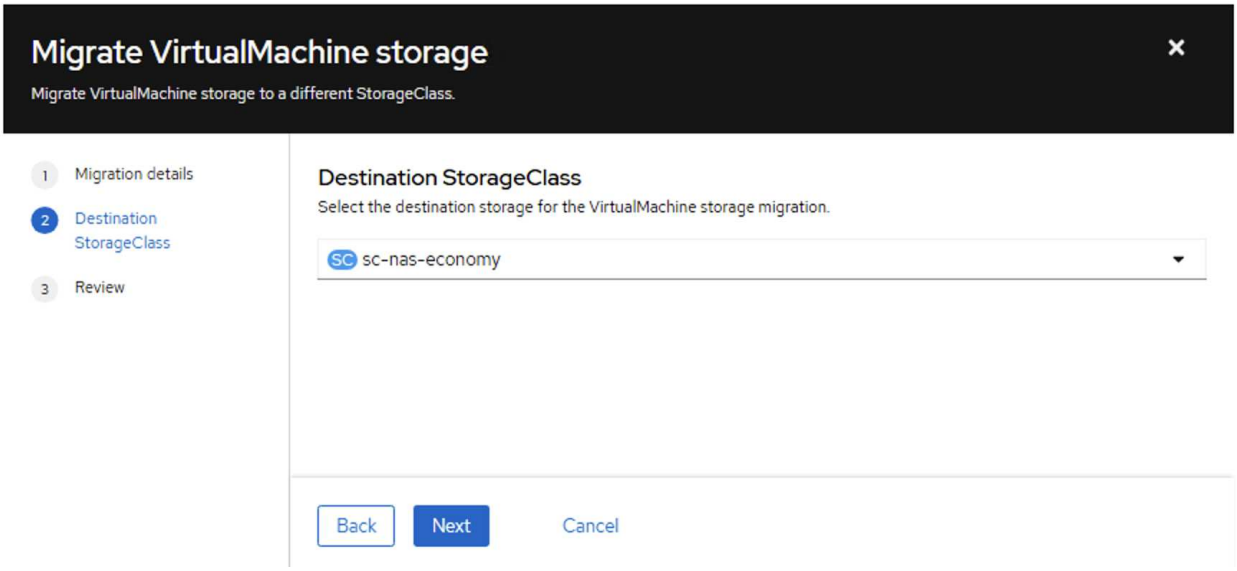


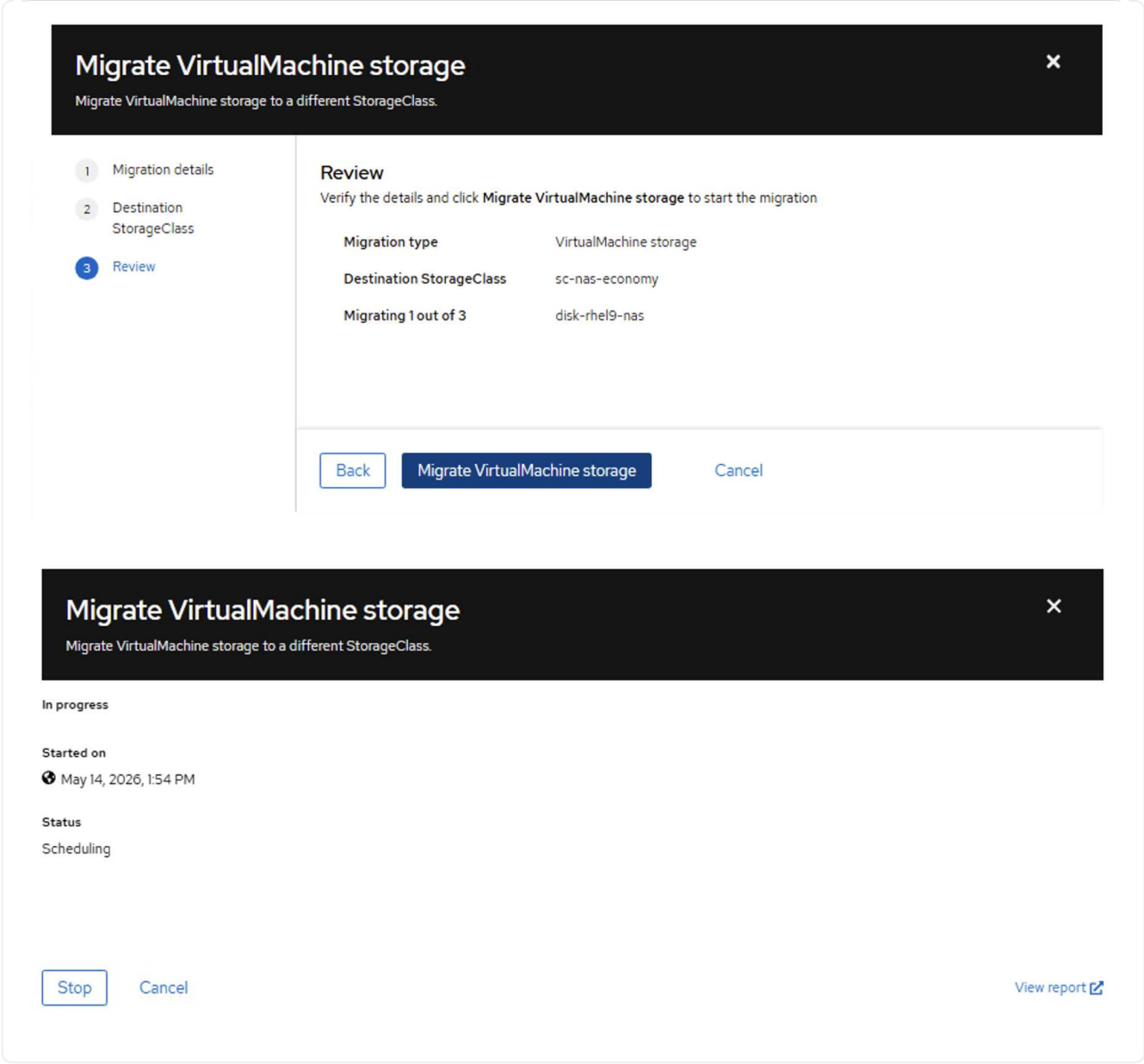
3. Click Actions→Migrate→Migrate Storage. Select the volumes to migrate and the target storage class, then click Migrate. The migration process will start and you can monitor the progress in the Events tab of the VM. Once the migration is complete, the VM's disks will be using the new storage class.

Show example









4. After the migration is complete, you can verify that the VM's disks are now using the new storage class by checking the VM's details in the web console or using the OpenShift CLI.

Show example

```
[root@00-50-56-b5-2b-9b ~]# oc get vm,pvc -n test-sc-migration
NAME                                          AGE   STATUS      READY
virtualmachine.kubevirt.io/rhel9-sc-migration   24m   Migrating   True

NAME                                                                                  STATUS   VOLUME                                     CAPACITY      ACCESS MODES   STORAGECLASS      VOLUMEATTRIBUTESCLASS   AGE
persistentvolumeclaim/dv-rhel9-migration-disk-emerald-wren-75-d88usc                  Bound    pvc-bb54342f-c01f-453e-864d-e8f72e710a2f   53Gi          RWX            sc-nas            <unset>                 24m
persistentvolumeclaim/dv-rhel9-migration-disk-emerald-wren-75-d88usc-mig-839zfb       Bound    pvc-75a885f0-824e-407c-ab29-8cc546dbdb3b   60322815673   RWX            sc-nas-economy    <unset>                 2m17s
persistentvolumeclaim/rhel9-migration                                                 Bound    pvc-46a34903-942f-4ddc-8e6d-974008f05d09   34144990004   RWX            sc-nas            <unset>                 24m
[root@00-50-56-b5-2b-9b ~]#
```

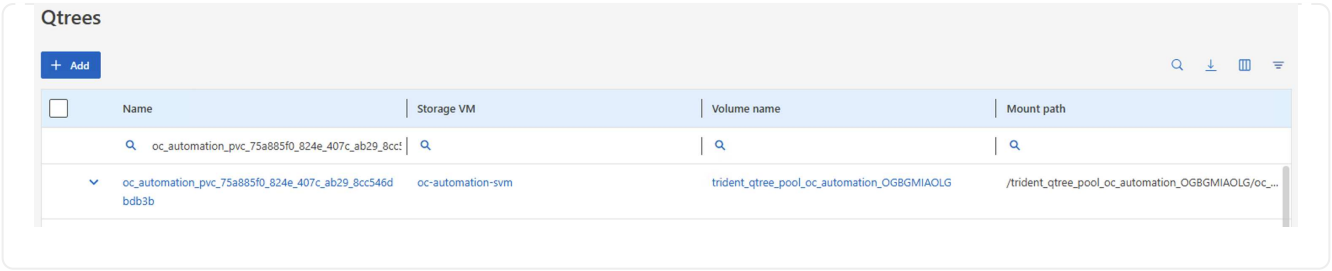
```
[root@00-50-56-b5-2b-9b ~]# oc get all -n test-sc-migration
Warning: apps.openshift.io/v1 DeploymentConfig is deprecated in v4.14+, unavailable in v4.10000+
Warning: kubevirt.io/v1 VirtualMachineInstancePresets is now deprecated and will be removed in v2.
NAME                                          READY   STATUS      RESTARTS   AGE
pod/virt-launcher-rhel9-sc-migration-44dc7   1/1     Running     0          2m50s
pod/virt-launcher-rhel9-sc-migration-5jkdw   0/1     Completed   0          25m

NAME              TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
service/headless   ClusterIP   None         <none>        5434/TCP   25m

NAME                                                                                  PHASE       PROGRESS   RESTARTS   AGE
datavolume.cdi.kubevirt.io/dv-rhel9-migration-disk-emerald-wren-75-d88usc             Succeeded   100.0%                25m
datavolume.cdi.kubevirt.io/dv-rhel9-migration-disk-emerald-wren-75-d88usc-mig-839zfb  Succeeded   100.0%                3m7s
datavolume.cdi.kubevirt.io/rhel9-migration                                            Succeeded   100.0%                25m

NAME                                                                                  PHASE       VMI
virtualmachineinstancemigration.kubevirt.io/kubevirt-workload-update-lbfj9   Succeeded   rhel9-sc-migration

NAME                                          AGE   PHASE     IP            NODENAME            READY
virtualmachineinstance.kubevirt.io/rhel9-sc-migration   25m   Running   10.131.0.22   00-50-56-b5-84-74   True

NAME                                          AGE   STATUS    READY
virtualmachine.kubevirt.io/rhel9-sc-migration   25m   Running   True
[root@00-50-56-b5-2b-9b ~]#
[root@00-50-56-b5-2b-9b ~]#
[root@00-50-56-b5-2b-9b ~]# oc get vm,pvc -n test-sc-migration
NAME                                          AGE   STATUS    READY
virtualmachine.kubevirt.io/rhel9-sc-migration   26m   Running   True

NAME                                                                                  STATUS   VOLUME                                     CAPACITY      ACCESS MODES   STORAGECLASS      VOLUMEATTRIBUTESCLASS   AGE
persistentvolumeclaim/dv-rhel9-migration-disk-emerald-wren-75-d88usc                  Bound    pvc-bb54342f-c01f-453e-864d-e8f72e710a2f   53Gi          RWX            sc-nas            <unset>                 26m
persistentvolumeclaim/dv-rhel9-migration-disk-emerald-wren-75-d88usc-mig-839zfb       Bound    pvc-75a885f0-824e-407c-ab29-8cc546dbdb3b   60322815673   RWX            sc-nas-economy    <unset>                 3m28s
persistentvolumeclaim/rhel9-migration                                                 Bound    pvc-46a34903-942f-4ddc-8e6d-974008f05d09   34144990004   RWX            sc-nas            <unset>                 26m
[root@00-50-56-b5-2b-9b ~]#
```

```
[root@00-50-56-b5-2b-9b ~]# oc describe pv pvc-46a34903-942f-4ddc-8e6d-974008f05d09 -n test-sc-migration
Name:            pvc-46a34903-942f-4ddc-8e6d-974008f05d09
Labels:          <none>
Annotations:     pv.kubernetes.io/provisioned-by: csi.trident.netapp.io
                 volume.kubernetes.io/provisioner-deletion-secret-name:
                 volume.kubernetes.io/provisioner-deletion-secret-namespace:
Finalizers:      [external-provisioner.volume.kubernetes.io/finalizer kubernetes.io/pv-protection external-attacher/csi-trident-netapp-io]
StorageClass:    sc-nas
Status:          Bound
Claim:           test-sc-migration/rhel9-migration
Reclaim Policy:  Delete
Access Modes:    RWX
VolumeMode:      Filesystem
Capacity:        34144990004
Node Affinity:   <none>
Message:
Source:
    Type:              CSI (a Container Storage Interface (CSI) volume source)
    Driver:            csi.trident.netapp.io
    FSType:
    VolumeHandle:      pvc-46a34903-942f-4ddc-8e6d-974008f05d09
    ReadOnly:          false
    VolumeAttributes:      backendUUID=259a1c2a-0a84-488d-ab62-ccea45499e09
                           internalName=oc_automation_pvc_46a34903_942f_4ddc_8e6d_974008f05d09
                           name=pvc-46a34903-942f-4ddc-8e6d-974008f05d09
                           protocol=file
                           storage.kubernetes.io/csiProvisionerIdentity=1778505459781-8296-csi.trident.netapp.io
Events:            <none>
[root@00-50-56-b5-2b-9b ~]#
```

```
NSOL-NetApp-A70-T19U01::> volume show -vserver oc-automation-svm -volume oc_automation_pvc_46a34903_942f_4ddc_8e6d_974008f05d09

                                    Vserver Name: oc-automation-svm
                                     Volume Name: oc_automation_pvc_46a34903_942f_4ddc_8e6d_974008f05d09
                                  Aggregate Name: NSOL_NetApp_A70_T19U01a_NVME_SSD_1
     List of Aggregates for FlexGroup Constituents: NSOL_NetApp_A70_T19U01a_
                                                    NVME_SSD_1
                                Encryption Type: none
                List of Nodes Hosting the Volume: NSOL-NetApp-A70-T19U01a
                                     Volume Size: 31.80GB
                              Volume Data Set ID: 1716
                       Volume Master Data Set ID: 2152408880
                                    Volume State: online
                                    Volume Style: flex
                           Extended Volume Style: flexvol
                         FlexCache Endpoint Type: none
                          Is Cluster-Mode Volume: true
                           Is Constituent Volume: false
                  Number of Constituent Volumes: -
                                 Export Policy: default
                                       User ID: 0
                                      Group ID: 0
                                Security Style: unix
                              UNIX Permissions: ---rwxrwxrwx
                                  Junction Path: /oc_automation_pvc_46a34903_942f_4ddc_8e6d_974008f05d09
                            Junction Path Source: RW_volume
                                 Junction Active: true
                          Junction Parent Volume: oc_automation_svm_root
                                         Comment:
                                  Available Size: 30.03GB
                                Filesystem Size: 31.80GB
                           Total User-Visible Size: 31.80GB
                                       Used Size: 1.76GB
                                  Used Percentage: 5%
            Volume Nearly Full Threshold Percent: 95%
                  Volume Full Threshold Percent: 98%
                                 Maximum Autosize: 38.16GB
                                 Minimum Autosize: 31.80GB
```

```
[root@00-50-56-b5-2b-9b ~]# oc get all -n test-sc-migration
Warning: apps.openshift.io/v1 DeploymentConfig is deprecated in v4.14+, unavailable in v4.10000+
Warning: kubevirt.io/v1 VirtualMachineInstancePresets is now deprecated and will be removed in v2.
NAME                                        READY   STATUS      RESTARTS   AGE
pod/virt-launcher-rhel9-sc-migration-44dc7   1/1     Running     0          2m50s
pod/virt-launcher-rhel9-sc-migration-5jkdw   0/1     Completed   0          25m

NAME              TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE
service/headless  ClusterIP   None         <none>        5434/TCP   25m

NAME                                                                    PHASE       PROGRESS   RESTARTS   AGE
datavolume.cdi.kubevirt.io/dv-rhel9-migration-disk-emerald-wren-75-d88usc              Succeeded   100.0%                25m
datavolume.cdi.kubevirt.io/dv-rhel9-migration-disk-emerald-wren-75-d88usc-mig-839zfb   Succeeded   100.0%                3m7s
datavolume.cdi.kubevirt.io/rhel9-migration                                             Succeeded   100.0%                25m

NAME                                                                PHASE       VMI
virtualmachineinstancemigration.kubevirt.io/kubevirt-workload-update-lbfj9   Succeeded   rhel9-sc-migration

NAME                                        AGE   PHASE     IP            NODENAME          READY
virtualmachineinstance.kubevirt.io/rhel9-sc-migration   25m   Running   10.131.0.22   00-50-56-b5-84-74   True

NAME                                      AGE   STATUS    READY
virtualmachine.kubevirt.io/rhel9-sc-migration   25m   Running   True
[root@00-50-56-b5-2b-9b ~]#
[root@00-50-56-b5-2b-9b ~]#
[root@00-50-56-b5-2b-9b ~]# oc get vm,pvc -n test-sc-migration
NAME                                      AGE   STATUS    READY
virtualmachine.kubevirt.io/rhel9-sc-migration   26m   Running   True

NAME                                                                                  STATUS   VOLUME                                     CAPACITY      ACCESS MODES   STORAGECLASS      VOLUMEATTRIBUTESCLASS   AGE
persistentvolumeclaim/dv-rhel9-migration-disk-emerald-wren-75-d88usc                  Bound    pvc-bb54342f-c01f-453e-864d-e8f72e710a2f   53Gi          RWX            sc-nas            <unset>                 26m
persistentvolumeclaim/dv-rhel9-migration-disk-emerald-wren-75-d88usc-mig-839zfb       Bound    pvc-75a885f0-824e-407c-ab29-8cc546dbdb3b   60322815673   RWX            sc-nas-economy    <unset>                 3m28s
persistentvolumeclaim/rhel9-migration                                                 Bound    pvc-46a34903-942f-4ddc-8e6d-974008f05d09   34144990004   RWX            sc-nas            <unset>                 26m
[root@00-50-56-b5-2b-9b ~]#
```

```
[root@00-50-56-b5-2b-9b ~]# oc describe pv -n test-sc-migration pvc-75a885f0-824e-407c-ab29-8cc546dbdb3b
Name:            pvc-75a885f0-824e-407c-ab29-8cc546dbdb3b
Labels:          <none>
Annotations:     pv.kubernetes.io/provisioned-by: csi.trident.netapp.io
                 volume.kubernetes.io/provisioner-deletion-secret-name:
                 volume.kubernetes.io/provisioner-deletion-secret-namespace:
Finalizers:      [external-provisioner.volume.kubernetes.io/finalizer kubernetes.io/pv-protection external-attacher/csi-trident-netapp-io]
StorageClass:    sc-nas-economy
Status:          Bound
Claim:           test-sc-migration/dv-rhel9-migration-disk-emerald-wren-75-d88usc-mig-839zfb
Reclaim Policy:  Delete
Access Modes:    RWX
VolumeMode:      Filesystem
Capacity:        60322815673
Node Affinity:   <none>
Message:
Source:
    Type:              CSI (a Container Storage Interface (CSI) volume source)
    Driver:            csi.trident.netapp.io
    FSType:
    VolumeHandle:      pvc-75a885f0-824e-407c-ab29-8cc546dbdb3b
    ReadOnly:          false
    VolumeAttributes:      backendUUID=a59e8053-bb00-4410-9b12-1dc4d283928b
                           internalID=/svm/oc-automation-svm/flexvol/trident_qtree_pool_oc_automation_OGBGMIAOLG/qtree/oc_automation_pvc_75a885f0_824e_407c_ab29_8cc546...
                           internalName=oc_automation_pvc_75a885f0_824e_407c_ab29_8cc546dbdb3b
                           name=pvc-75a885f0-824e-407c-ab29-8cc546dbdb3b
                           protocol=file
                           storage.kubernetes.io/csiProvisionerIdentity=1778505459781-8296-csi.trident.netapp.io
Events:            <none>
```


Qtrees
Add
Name
Storage VM
Volume name
Mount path
oc_automation_pvc_75a885f0_824e_407c_ab29_8cc
oc_automation_pvc_75a885f0_824e_407c_ab29_8cc546dbdb3b
oc-automation-svm
trident_qtree_pool_oc_automation_OGBGMIAOLG
/trident_qtree_pool_oc_automation_OGBGMIAOLG/oc_...

Copyright information

Trademark information