



Migration Toolkit for Virtualization 2.2

Installing and using the Migration Toolkit for Virtualization

Migrating from VMware vSphere or Red Hat Virtualization to Red Hat OpenShift
Virtualization

Migration Toolkit for Virtualization 2.2 Installing and using the Migration Toolkit for Virtualization

Migrating from VMware vSphere or Red Hat Virtualization to Red Hat OpenShift Virtualization

Red Hat Modernization and Migration Documentation Team
ccs-mms-docs@redhat.com

Legal Notice

Copyright © 2022 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

The Migration Toolkit for Virtualization (MTV) enables you to migrate virtual machines from VMware vSphere or Red Hat Virtualization to OpenShift Virtualization running on OpenShift Container Platform.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE	4
CHAPTER 1. ABOUT THE MIGRATION TOOLKIT FOR VIRTUALIZATION	5
1.1. ABOUT COLD AND WARM MIGRATION	5
1.1.1. Cold migration	5
1.1.2. Warm migration	5
CHAPTER 2. PREREQUISITES	7
2.1. SOFTWARE REQUIREMENTS	7
2.2. STORAGE SUPPORT AND DEFAULT MODES	7
2.3. NETWORK PREREQUISITES	8
2.3.1. Ports	8
2.4. SOURCE VIRTUAL MACHINE PREREQUISITES	9
2.5. RED HAT VIRTUALIZATION PREREQUISITES	9
2.6. VMWARE PREREQUISITES	9
2.6.1. Creating a VDDK image	9
2.6.2. Obtaining the SHA-1 fingerprint of a vCenter host	10
2.6.3. Increasing the NFC service memory of an ESXi host	11
2.7. SOFTWARE COMPATIBILITY GUIDELINES	11
CHAPTER 3. INSTALLING THE MTV OPERATOR	13
3.1. INSTALLING THE MTV OPERATOR BY USING THE OPENSIFT CONTAINER PLATFORM WEB CONSOLE	13
Obtaining the MTV web console URL	13
3.2. INSTALLING THE MTV OPERATOR FROM THE COMMAND LINE INTERFACE	14
Obtaining the MTV web console URL	15
CHAPTER 4. MIGRATING VIRTUAL MACHINES BY USING THE MTV WEB CONSOLE	16
4.1. ADDING PROVIDERS	16
4.1.1. Adding a VMware source provider	16
4.1.1.1. Selecting a migration network for a VMware source provider	17
4.1.2. Adding a Red Hat Virtualization source provider	18
4.1.3. Adding an OpenShift Virtualization provider	18
4.1.3.1. Selecting a migration network for an OpenShift Virtualization provider	19
4.2. CREATING A NETWORK MAPPING	19
4.3. CREATING A STORAGE MAPPING	20
4.4. CREATING A MIGRATION PLAN	20
4.5. RUNNING A MIGRATION PLAN	22
4.6. MIGRATION PLAN OPTIONS	23
4.7. CANCELING A MIGRATION	23
CHAPTER 5. MIGRATING VIRTUAL MACHINES FROM THE COMMAND LINE	25
5.1. MIGRATING VIRTUAL MACHINES	25
5.2. CANCELING A MIGRATION	30
CHAPTER 6. ADVANCED MIGRATION OPTIONS	32
6.1. CHANGING PRECOPY INTERVALS FOR WARM MIGRATION	32
6.2. CREATING CUSTOM RULES FOR THE VALIDATION SERVICE	32
6.2.1. About Rego files	32
6.2.2. Checking the default validation rules	33
6.2.3. Retrieving the Inventory service JSON	33
6.2.4. Creating a validation rule	40
6.2.5. Updating the inventory rules version	42

CHAPTER 7. UPGRADING THE MIGRATION TOOLKIT FOR VIRTUALIZATION	44
CHAPTER 8. UNINSTALLING THE MIGRATION TOOLKIT FOR VIRTUALIZATION	46
8.1. UNINSTALLING MTV BY USING THE OPENSIFT CONTAINER PLATFORM WEB CONSOLE	46
8.2. UNINSTALLING MTV FROM THE COMMAND LINE INTERFACE	46
CHAPTER 9. TROUBLESHOOTING	48
9.1. ARCHITECTURE	48
9.1.1. MTV custom resources and services	48
9.1.2. High-level migration workflow	49
9.1.3. Detailed migration workflow	49
9.2. LOGS AND CUSTOM RESOURCES	50
9.2.1. Collected logs and custom resource information	50
9.2.2. Downloading logs and custom resource information from the web console	51
9.2.3. Accessing logs and custom resource information from the command line interface	52

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see [our CTO Chris Wright's message](#).

CHAPTER 1. ABOUT THE MIGRATION TOOLKIT FOR VIRTUALIZATION

You can migrate virtual machines from VMware vSphere or Red Hat Virtualization to OpenShift Virtualization with the Migration Toolkit for Virtualization (MTV).

Additional resources

- [Performance recommendations for migrating from VMware vSphere to OpenShift Virtualization](#) .
- [Performance recommendations for migrating from Red Hat Virtualization to OpenShift Virtualization](#).

1.1. ABOUT COLD AND WARM MIGRATION

MTV supports cold migration from Red Hat Virtualization and warm migration from VMware vSphere.

1.1.1. Cold migration

Cold migration is the default migration type. The source virtual machines are shut down while the data is copied.

1.1.2. Warm migration

Most of the data is copied during the *precopy* stage while the source virtual machines (VMs) are running.

Then the VMs are shut down and the remaining data is copied during the *cutover* stage.

Precopy stage

The VMs are not shut down during the precopy stage.

The VM disks are copied incrementally using [changed block tracking \(CBT\)](#) snapshots. The snapshots are created at one-hour intervals by default. You can change the snapshot interval by updating the **forklift-controller** deployment.



IMPORTANT

You must enable CBT for each source VM and each VM disk.

A VM can support up to 28 CBT snapshots. If the source VM has too many CBT snapshots and the **Migration Controller** service is not able to create a new snapshot, warm migration might fail. The **Migration Controller** service deletes each snapshot when the snapshot is no longer required.

The precopy stage runs until the cutover stage is started manually or is scheduled to start.

Cutover stage

The VMs are shut down during the cutover stage and the remaining data is migrated. Data stored in RAM is not migrated.

You can start the cutover stage manually by using the MTV console or you can schedule a cutover time in the **Migration** manifest.

CHAPTER 2. PREREQUISITES

Review the following prerequisites to ensure that your environment is prepared for migration.

2.1. SOFTWARE REQUIREMENTS

You must install [compatible versions](#) of OpenShift Container Platform and OpenShift Virtualization.

2.2. STORAGE SUPPORT AND DEFAULT MODES

MTV uses the following default volume and access modes for supported storage.



NOTE

If the OpenShift Virtualization storage does not support [dynamic provisioning](#), MTV applies the default settings:

- **Filesystem** volume mode
Filesystem volume mode is slower than **Block** volume mode.
- **ReadWriteOnce** access mode
ReadWriteOnce access mode does not support live virtual machine migration.

Table 2.1. Default volume and access modes

Provisioner	Volume mode	Access mode
kubernetes.io/aws-efs	Block	ReadWriteOnce
kubernetes.io/azure-disk	Block	ReadWriteOnce
kubernetes.io/azure-file	Filesystem	ReadWriteMany
kubernetes.io/cinder	Block	ReadWriteOnce
kubernetes.io/gce-pd	Block	ReadWriteOnce
kubernetes.io/hostpath-provisioner	Filesystem	ReadWriteOnce
manila.csi.openstack.org	Filesystem	ReadWriteMany
openshift-storage.cephfs.csi.ceph.com	Filesystem	ReadWriteMany
openshift-storage.rbd.csi.ceph.com	Block	ReadWriteOnce
kubernetes.io/rbd	Block	ReadWriteOnce

Provisioner	Volume mode	Access mode
kubernetes.io/vsphere-volume	Block	ReadWriteOnce

2.3. NETWORK PREREQUISITES

The following prerequisites apply to all migrations:

- IP addresses, VLANs, and other network configuration settings must not be changed before or after migration. The MAC addresses of the virtual machines are preserved during migration.
- The network connections between the source environment, the OpenShift Virtualization cluster, and the replication repository must be reliable and uninterrupted.
- If you are mapping more than one source and destination network, you must create a [network attachment definition](#) for each additional destination network.

2.3.1. Ports

The firewalls must enable traffic over the following ports:

Table 2.2. Network ports required for migrating from VMware vSphere

Port	Protocol	Source	Destination	Purpose
443	TCP	OpenShift nodes	VMware vCenter	VMware provider inventory Disk transfer authentication
443	TCP	OpenShift nodes	VMware ESXi hosts	Disk transfer authentication
902	TCP	OpenShift nodes	VMware ESXi hosts	Disk transfer data copy

Table 2.3. Network ports required for migrating from Red Hat Virtualization

Port	Protocol	Source	Destination	Purpose
443	TCP	OpenShift nodes	RHV Engine	RHV provider inventory Disk transfer authentication
443	TCP	OpenShift nodes	RHV hosts	Disk transfer authentication
54322	TCP	OpenShift nodes	RHV hosts	Disk transfer data copy

2.4. SOURCE VIRTUAL MACHINE PREREQUISITES

The following prerequisites apply to all migrations:

- ISO/CDROM disks must be unmounted.
- Each NIC must contain one IPv4 and/or one IPv6 address.
- The VM name must contain only lowercase letters (**a-z**), numbers (**0-9**), or hyphens (-), up to a maximum of 253 characters. The first and last characters must be alphanumeric. The name must not contain uppercase letters, spaces, periods (.), or special characters.
- The VM name must not duplicate the name of a VM in the OpenShift Virtualization environment.
- The VM operating system must be certified and supported for use as a [guest operating system with OpenShift Virtualization](#) and for [conversion to KVM with virt-v2v](#).

2.5. RED HAT VIRTUALIZATION PREREQUISITES

The following prerequisites apply to Red Hat Virtualization migrations:

- You must use a [compatible version](#) of Red Hat Virtualization.
- You must have the CA certificate of the Manager.
You can obtain the CA certificate by navigating to **https://<engine_host>/ovirt-engine/services/pki-resource?resource=ca-certificate&format=X509-PEM-CA** in a browser.

2.6. VMWARE PREREQUISITES

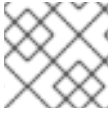
The following prerequisites apply to VMware migrations:

- You must use a [compatible version](#) of VMware vSphere.
- You must install [VMware Tools](#) on all source virtual machines (VMs).
- If you are running a warm migration, you must enable [changed block tracking \(CBT\)](#) on the VMs and on the VM disks.
- You must create a VMware Virtual Disk Development Kit (VDDK) image.
- You must obtain the SHA-1 fingerprint of the vCenter host.
- If you are migrating more than 10 VMs from an ESXi host in the same migration plan, you must increase the NFC service memory of the host.

2.6.1. Creating a VDDK image

The Migration Toolkit for Virtualization (MTV) uses the VMware Virtual Disk Development Kit (VDDK) SDK to transfer virtual disks from VMware vSphere.

You must download the VMware Virtual Disk Development Kit (VDDK), build a VDDK image, and push the VDDK image to your image registry. Later, you will add the VDDK image to the **HyperConverged** custom resource (CR).

**NOTE**

Storing the VDDK image in a public registry might violate the VMware license terms.

Prerequisites

- [OpenShift Container Platform image registry](#)
- **podman** installed.
- If you are using an external registry, OpenShift Virtualization must be able to access it.

Procedure

1. Create and navigate to a temporary directory:

```
$ mkdir /tmp/<dir_name> && cd /tmp/<dir_name>
```

2. In a browser, navigate to the [VMware VDDK download page](#).
3. Select the latest VDDK version and click **Download**.
4. Save the VDDK archive file in the temporary directory.
5. Extract the VDDK archive:

```
$ tar -xzf VMware-vix-disklib-<version>.x86_64.tar.gz
```

6. Create a **Dockerfile**:

```
$ cat > Dockerfile <<EOF
FROM registry.access.redhat.com/ubi8/ubi-minimal
COPY vmware-vix-disklib-distrib /vmware-vix-disklib-distrib
RUN mkdir -p /opt
ENTRYPOINT ["cp", "-r", "/vmware-vix-disklib-distrib", "/opt"]
EOF
```

7. Build the VDDK image:

```
$ podman build . -t <registry_route_or_server_path>/vddk:<tag>
```

8. Push the VDDK image to the registry:

```
$ podman push <registry_route_or_server_path>/vddk:<tag>
```

9. Ensure that the image is accessible to your OpenShift Virtualization environment.

2.6.2. Obtaining the SHA-1 fingerprint of a vCenter host

You must obtain the SHA-1 fingerprint of a vCenter host in order to create a **Secret** CR.

Procedure

- Run the following command:

```
$ openssl s_client \
  -connect <vcenter_host>:443 \ 1
  < /dev/null 2>/dev/null \
  | openssl x509 -fingerprint -noout -in /dev/stdin \
  | cut -d '=' -f 2
```

- 1 Specify the IP address or FQDN of the vCenter host.

Example output

```
01:23:45:67:89:AB:CD:EF:01:23:45:67:89:AB:CD:EF:01:23:45:67
```

2.6.3. Increasing the NFC service memory of an ESXi host

If you are migrating more than 10 VMs from an ESXi host in the same migration plan, you must increase the NFC service memory of the host. Otherwise, the migration will fail because the NFC service memory is limited to 10 parallel connections.

Procedure

1. Log in to the ESXi host as root.
2. Change the value of **maxMemory** to **1000000000** in **/etc/vmware/hostd/config.xml**:

```
...
  <nfcsvc>
    <path>libnfcsvc.so</path>
    <enabled>true</enabled>
    <maxMemory>1000000000</maxMemory>
    <maxStreamMemory>10485760</maxStreamMemory>
  </nfcsvc>
...
```

3. Restart **hostd**:

```
# /etc/init.d/hostd restart
```

You do not need to reboot the host.

2.7. SOFTWARE COMPATIBILITY GUIDELINES

You must install compatible software versions.

Table 2.4. Compatible software versions

Migration Toolkit for Virtualization	OpenShift Container Platform	OpenShift Virtualization	VMware vSphere	Red Hat Virtualization
2.2	4.9	4.9.1	6.5 or later	4.3 or later

CHAPTER 3. INSTALLING THE MTV OPERATOR

You can install the MTV Operator by using the OpenShift Container Platform web console or the command line interface (CLI).

3.1. INSTALLING THE MTV OPERATOR BY USING THE OPENSIFT CONTAINER PLATFORM WEB CONSOLE

You can install the MTV Operator by using the OpenShift Container Platform web console.

Prerequisites

- OpenShift Container Platform 4.9 installed.
- OpenShift Virtualization Operator installed.
- You must be logged in as a user with **cluster-admin** permissions.

Procedure

1. In the OpenShift Container Platform web console, click **Operators** → **OperatorHub**.
2. Use the **Filter by keyword** field to search for **mtv-operator**.
3. Click **Migration Toolkit for Virtualization Operator** and then click **Install**.
4. On the **Install Operator** page, click **Install**.
5. Click **Operators** → **Installed Operators** to verify that **Migration Toolkit for Virtualization Operator** appears in the **openshift-mtv** project with the status **Succeeded**.
6. Click **Migration Toolkit for Virtualization Operator**.
7. Under **Provided APIs**, locate the **ForkliftController**, and click **Create Instance**.
8. Click **Create**.
9. Click **Workloads** → **Pods** to verify that the MTV pods are running.

Obtaining the MTV web console URL

You can obtain the MTV web console URL by using the OpenShift Container Platform web console.

Prerequisites

- You must have the OpenShift Virtualization Operator installed.
- You must have the MTV Operator installed.
- You must be logged in as a user with **cluster-admin** privileges.

Procedure

1. Log in to the OpenShift Container Platform web console.
2. Click **Networking** → **Routes**.

3. Select the **openshift-mtv** project in the **Project:** list.
4. Click the URL for the **forklift-ui** service to open the login page for the MTV web console.

3.2. INSTALLING THE MTV OPERATOR FROM THE COMMAND LINE INTERFACE

You can install the MTV Operator from the command line interface (CLI).

Prerequisites

- OpenShift Container Platform 4.9 installed.
- OpenShift Virtualization Operator installed.
- You must be logged in as a user with **cluster-admin** permissions.

Procedure

1. Create the openshift-mtv project:

```
$ cat << EOF | oc apply -f -
apiVersion: project.openshift.io/v1
kind: Project
metadata:
  name: openshift-mtv
EOF
```

2. Create an **OperatorGroup** CR called **migration**:

```
$ cat << EOF | oc apply -f -
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: migration
  namespace: openshift-mtv
spec:
  targetNamespaces:
    - openshift-mtv
EOF
```

3. Create a **Subscription** CR for the Operator:

```
$ cat << EOF | oc apply -f -
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: mtv-operator
  namespace: openshift-mtv
spec:
  channel: release-v2.2.0
  installPlanApproval: Automatic
  name: mtv-operator
  source: redhat-operators
```

```
sourceNamespace: openshift-marketplace
startingCSV: "mtv-operator.2.2.0"
EOF
```

4. Create a **ForkliftController** CR:

```
$ cat << EOF | oc apply -f -
apiVersion: forklift.konveyor.io/v1beta1
kind: ForkliftController
metadata:
  name: forklift-controller
  namespace: openshift-mtv
spec:
  olm_managed: true
EOF
```

5. Verify that the MTV pods are running:

```
$ oc get pods -n openshift-mtv
```

Example output

```
NAME                                READY STATUS RESTARTS AGE
forklift-controller-788bdb4c69-mw268 2/2 Running 0      2m
forklift-operator-6bf45b8d8-qps9v    1/1 Running 0      5m
forklift-ui-7cdf96d8f6-xnw5n        1/1 Running 0      2m
```

Obtaining the MTV web console URL

You can obtain the MTV web console URL from the command line.

Prerequisites

- You must have the OpenShift Virtualization Operator installed.
- You must have the MTV Operator installed.
- You must be logged in as a user with **cluster-admin** privileges.

Procedure

1. Obtain the MTV web console URL:

```
$ oc get route virt -n openshift-mtv \
-o custom-columns=:.spec.host
```

Example output

```
https://virt-openshift-mtv.apps.cluster.openshift.com.
```

2. Launch a browser and navigate to the MTV web console.

CHAPTER 4. MIGRATING VIRTUAL MACHINES BY USING THE MTV WEB CONSOLE

You can migrate virtual machines (VMs) to OpenShift Virtualization by using the MTV web console.



IMPORTANT

You must ensure that all [prerequisites](#) are met.

4.1. ADDING PROVIDERS

You can add providers by using the MTV web console.

4.1.1. Adding a VMware source provider

You can add a VMware source provider by using the MTV web console.

Prerequisites

- You must have VMware admin privileges.
- vCenter SHA-1 fingerprint.
- VMware Virtual Disk Development Kit (VDDK) image in a secure registry that is accessible to all clusters.

Procedure

1. Add the VDDK image to the **HyperConverged** CR:

```
$ cat << EOF | oc apply -f -
apiVersion: hco.kubevirt.io/v1beta1
kind: HyperConverged
metadata:
  name: kubevirt-hyperconverged
  namespace: openshift-cnv
spec:
  vddkInitImage: <registry_route_or_server_path>/vddk:<tag>
EOF
```

- 1 Specify the VDDK image that you created.

2. In the MTV web console, click **Providers**.
3. Click **Add provider**.
4. Select **VMware** from the **Type** list.
5. Fill in the following fields:
 - **Name:** Name to display in the list of providers
 - **Hostname or IP address:** vCenter host name or IP address

- **Username:** vCenter admin user, for example, **administrator@vsphere.local**
 - **Password:** vCenter admin password
 - **SHA-1 fingerprint:** vCenter SHA-1 fingerprint
6. Click **Add** to add and save the provider.
The source provider appears in the list of providers.

4.1.1.1. Selecting a migration network for a VMware source provider

You can select a migration network in the MTV web console for a VMware source provider to reduce risk to the source environment and to improve performance.

Using the default network for migration can result in poor performance because the network might not have sufficient bandwidth. This situation can have a negative effect on the source platform because the disk transfer operation might saturate the network.

Prerequisites

- The migration network must have sufficient throughput, minimum speed of 10 Gbps, for disk transfer.
- The migration network must be accessible to the OpenShift Virtualization nodes through the default gateway.



NOTE

The source virtual disks are copied by a pod that is connected to the pod network of the target namespace.

- The migration network must have jumbo frames enabled.

Procedure

1. In the MTV web console, click **Providers**
2. Click the **VMware** tab.
3. Click the host number in the **Hosts** column beside a provider to view a list of hosts.
4. Select one or more hosts and click **Select migration network**.
5. Select a **Network**.
You can clear the selection by selecting the default network.
6. If your source provider is VMware, complete the following fields:
 - **ESXi host admin username:** Specify the ESXi host admin user, for example, **root**.
 - **ESXi host admin password:** Specify the ESXi host admin password.
7. If your source provider is Red Hat Virtualization, complete the following fields:
 - **Username:** Specify the Manager user.

- **Password:** Specify the Manager password.
8. Click **Save**.
 9. Verify that the status of each host is **Ready**.
If a host status is not **Ready**, the host might be unreachable on the migration network or the credentials might be incorrect. You can modify the host configuration and save the changes.

4.1.2. Adding a Red Hat Virtualization source provider

You can add a Red Hat Virtualization source provider by using the MTV web console.

Prerequisites

- CA certificate of the Manager.

Procedure

1. In the MTV web console, click **Providers**.
2. Click **Add provider**.
3. Select **Red Hat Virtualization** from the **Type** list.
4. Fill in the following fields:
 - **Name:** Name to display in the list of providers
 - **Hostname or IP address:** Manager host name or IP address
 - **Username:** Manager user
 - **Password:** Manager password
 - **CA certificate:** CA certificate of the Manager
5. Click **Add** to add and save the provider.
The source provider appears in the list of providers.

4.1.3. Adding an OpenShift Virtualization provider

You can add an OpenShift Virtualization provider to the MTV web console in addition to the default OpenShift Virtualization provider, which is the provider where you installed MTV.

Prerequisites

- You must have an OpenShift Virtualization [service account token](#) with **cluster-admin** privileges.

Procedure

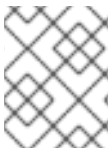
1. In the MTV web console, click **Providers**.
2. Click **Add provider**.
3. Select **OpenShift Virtualization** from the **Type** list.

4. Complete the following fields:
 - **Cluster name:** Specify the cluster name to display in the list of target providers.
 - **URL:** Specify the API endpoint of the cluster.
 - **Service account token:** Specify the **cluster-admin** service account token.
5. Click **Check connection** to verify the credentials.
6. Click **Add**.
The provider appears in the list of providers.

4.1.3.1. Selecting a migration network for an OpenShift Virtualization provider

You can select a default migration network for an OpenShift Virtualization provider in the MTV web console to improve performance. The default migration network is used to transfer disks to the namespaces in which it is configured.

If you do not select a migration network, the default migration network is the **pod** network, which might not be optimal for disk transfer.



NOTE

You can override the default migration network of the provider by selecting a different network when you create a migration plan.

Procedure

1. In the MTV web console, click **Providers**.
2. Click the **OpenShift Virtualization** tab.
3. Select a provider and click **Select migration network**.
4. Select a network from the list of available networks and click **Select**.
5. Click the network number in the **Networks** column beside the provider to verify that the selected network is the default migration network.

4.2. CREATING A NETWORK MAPPING

You can create one or more network mappings by using the MTV web console to map source networks to OpenShift Virtualization networks.

Prerequisites

- Source and target providers added to the web console.
- If you map more than one source and target network, each additional OpenShift Virtualization network requires its own [network attachment definition](#).

Procedure

1. Click **Mappings**.

2. Click the **Network** tab and then click **Create mapping**.
3. Complete the following fields:
 - **Name:** Enter a name to display in the network mappings list.
 - **Source provider:** Select a source provider.
 - **Target provider:** Select a target provider.
 - **Source networks:** Select a source network.
 - **Target namespaces/networks:** Select a target network.
4. Optional: Click **Add** to create additional network mappings or to map multiple source networks to a single target network.
5. If you create an additional network mapping, select the network attachment definition as the target network.
6. Click **Create**.
The network mapping is displayed on the **Network mappings** screen.

4.3. CREATING A STORAGE MAPPING

You can create a storage mapping by using the MTV web console to map source data stores to OpenShift Virtualization storage classes.

Prerequisites

- Source and target providers added to the web console.
- Local and shared persistent storage that support VM migration.

Procedure

1. Click **Mappings**.
2. Click the **Storage** tab and then click **Create mapping**.
3. Enter the **Name** of the storage mapping.
4. Select a **Source provider** and a **Target provider**.
5. If your source provider is VMware, select a **Source datastore** and a **Target storage class**.
6. If your source provider is Red Hat Virtualization, select a **Source storage domain** and a **Target storage class**.
7. Optional: Click **Add** to create additional storage mappings or to map multiple source data stores or storage domains to a single storage class.
8. Click **Create**.
The mapping is displayed on the **Storage mappings** page.

4.4. CREATING A MIGRATION PLAN

You can create a migration plan by using the MTV web console.

A migration plan allows you to group virtual machines to be migrated together or with the same migration parameters, for example, a percentage of the members of a cluster or a complete application.

You can configure a hook to run an Ansible playbook or custom container image during a specified stage of the migration plan.

Prerequisites

- If MTV is not installed on the target cluster, you must add a target provider on the **Providers** page of the web console.

Procedure


1. In the web console, click **Migration plans** and then click **Create migration plan**.
2. Complete the following fields:
 - **Plan name:** Enter a migration plan name to display in the migration plan list.
 - **Plan description:** Optional: Brief description of the migration plan.
 - **Source provider:** Select a source provider.
 - **Target provider:** Select a target provider.
 - **Target namespace:** You can type to search for an existing target namespace or create a new namespace.
 - You can change the migration transfer network for this plan by clicking **Select a different network**, selecting a network from the list, and clicking **Select**.
If you defined a migration transfer network for the OpenShift Virtualization provider and if the network is in the target namespace, that network is the default network for all migration plans. Otherwise, the **pod** network is used.
3. Click **Next**.
4. Select options to filter the list of source VMs and click **Next**.
5. Select the VMs to migrate and then click **Next**.
6. Select an existing network mapping or create a new network mapping.
To create a new network mapping:
 - Select a target network for each source network.
 - Optional: Select **Save mapping to use again** and enter a network mapping name.
7. Click **Next**.
8. Select an existing storage mapping or create a new storage mapping.
To create a new storage mapping:
 - Select a target storage class for each VMware data store or Red Hat Virtualization storage domain.

- Optional: Select **Save mapping to use again** and enter a storage mapping name.
9. Click **Next**.
 10. Select a migration type and click **Next**.
 - Cold migration: The source VMs are stopped while the data is copied.
 - Warm migration: The source VMs run while the data is copied incrementally. Later, you will run the cutover, which stops the VMs and copies the remaining VM data and metadata. Warm migration is not supported for Red Hat Virtualization.
 11. Optional: You can create a migration hook to run an Ansible playbook before or after migration:
 - a. Click **Add hook**
 - b. Select the step when the hook will run.
 - c. Select a hook definition:
 - **Ansible playbook**: Browse to the Ansible playbook or paste it into the field.
 - **Custom container image**: If you do not want to use the default **hook-runner** image, enter the image path: `<registry_path>/<image_name>:<tag>`.



NOTE

The registry must be accessible to your OpenShift Container Platform cluster.

12. Click **Next**.
13. Review your migration plan and click **Finish**.
The migration plan is saved in the migration plan list.
14. Click the Options menu  of the migration plan and select **View details** to verify the migration plan details.

4.5. RUNNING A MIGRATION PLAN

You can run a migration plan and view its progress in the MTV web console.

Prerequisites

- Valid migration plan.

Procedure

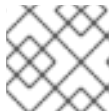
1. Click **Migration plans**.
The **Migration plans** list displays the source and target providers, the number of virtual machines (VMs) being migrated, and the status of the plan.
2. Click **Start** beside a migration plan to start the migration.
Warm migration only:

- The precopy stage starts.
 - Click **Cutover** to complete the migration.
3. Expand a migration plan to view the migration details.
The migration details screen displays the migration start and end time, the amount of data copied, and a progress pipeline for each VM being migrated.
 4. Expand a VM to view the migration steps, elapsed time of each step, and its state.

4.6. MIGRATION PLAN OPTIONS

On the **Migration plans** page of the MTV web console, you can click the Options menu  beside a migration plan to access the following options:

- **Edit:** Edit the details of a migration plan. You cannot edit a migration plan while it is running or after it has completed successfully.
- **Duplicate:** Create a new migration plan with the same virtual machines (VMs), parameters, mappings, and hooks as an existing plan. You can use this feature for the following tasks:
 - Migrate VMs to a different namespace.
 - Edit an archived migration plan.
 - Edit a migration plan with a different status, for example, failed, canceled, running, critical, or ready.
- **Archive:** Delete the logs, history, and metadata of a migration plan. The plan cannot be edited or restarted. It can only be viewed.



NOTE

The **Archive** option is irreversible. However, you can duplicate an archived plan.

- **Delete:** Permanently remove a migration plan. You cannot delete a running migration plan.



NOTE

The **Delete** option is irreversible.

Deleting a migration plan does not remove temporary resources such as **importer** pods, **conversion** pods, config maps, secrets, failed VMs, and data volumes. ([BZ#2018974](#)) You must archive a migration plan before deleting it in order to clean up the temporary resources.

- **View details:** Display the details of a migration plan.
- **Restart:** Restart a failed or canceled migration plan.
- **Cancel scheduled cutover:** Cancel a scheduled cutover migration for a warm migration plan.

4.7. CANCELING A MIGRATION

You can cancel the migration of some or all virtual machines (VMs) while a migration plan is in progress by using the MTV web console.

Procedure

1. Click **Migration Plans**.
2. Click the name of a running migration plan to view the migration details.
3. Select one or more VMs and click **Cancel**.
4. Click **Yes, cancel** to confirm the cancellation.
In the **Migration details by VM** list, the status of the canceled VMs is **Canceled**. The unmigrated and the migrated virtual machines are not affected.

You can restart a canceled migration by clicking **Restart** beside the migration plan on the **Migration plans** page.

CHAPTER 5. MIGRATING VIRTUAL MACHINES FROM THE COMMAND LINE

You can migrate virtual machines to OpenShift Virtualization from the command line.



IMPORTANT

You must ensure that all [prerequisites](#) are met.

5.1. MIGRATING VIRTUAL MACHINES

You migrate virtual machines (VMs) from the command line (CLI) by creating MTV custom resources (CRs).



IMPORTANT

You must specify a name for cluster-scoped CRs.

You must specify both a name and a namespace for namespace-scoped CRs.

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.
- VMware only: You must have a VMware Virtual Disk Development Kit (VDDK) image in a secure registry that is accessible to all clusters.

Procedure

1. VMware only: Add the VDDK image to the **HyperConverged** CR:

```
$ cat << EOF | oc apply -f -
apiVersion: hco.kubevirt.io/v1beta1
kind: HyperConverged
metadata:
  name: kubevirt-hyperconverged
  namespace: openshift-cnv
spec:
  vddkInitImage: <registry_route_or_server_path>/vddk:<tag> 1
EOF
```

- 1** Specify the VDDK image that you created.

2. Create a **Secret** manifest for the source provider credentials:

```
$ cat << EOF | oc apply -f -
apiVersion: v1
kind: Secret
metadata:
  name: <secret>
  namespace: openshift-mtv
type: Opaque
```

```
stringData:
  user: <user> 1
  password: <password> 2
  cacert: <engine_ca_certificate> 3
  thumbprint: <vcenter_fingerprint> 4
EOF
```

- 1 Specify the base64-encoded vCenter admin user or the RHV Manager user.
- 2 Specify the base64-encoded password.
- 3 RHV only: Specify the base64-encoded CA certificate of the Manager. You can retrieve it at https://<engine_host>/ovirt-engine/services/pki-resource?resource=ca-certificate&format=X509-PEM-CA.
- 4 VMware only: Specify the vCenter SHA-1 fingerprint.

3. Create a **Provider** manifest for the source provider:

```
$ cat << EOF | oc apply -f -
apiVersion: forklift.konveyor.io/v1beta1
kind: Provider
metadata:
  name: <provider>
  namespace: openshift-mtv
spec:
  type: <provider_type> 1
  url: <api_end_point> 2
  secret:
    name: <secret> 3
    namespace: openshift-mtv
EOF
```

- 1 Allowed values are **ovirt** and **vsphere**.
- 2 Specify the API end point URL, for example, https://<vCenter_host>/sdk for vSphere or https://<engine_host>/ovirt-engine/api/ for RHV.
- 3 Specify the name of provider **Secret** CR.

4. VMware only: Create a **Host** manifest:

```
$ cat << EOF | oc apply -f -
apiVersion: forklift.konveyor.io/v1beta1
kind: Host
metadata:
  name: <vmware_host>
  namespace: openshift-mtv
spec:
  provider:
    namespace: openshift-mtv
    name: <source_provider> 1
```

```
id: <source_host_mor> 2
ipAddress: <source_network_ip> 3
EOF
```

- 1 Specify the name of the VMware **Provider** CR.
- 2 Specify the managed object reference (MOR) of the VMware host.
- 3 Specify the IP address of the VMware migration network.

5. Create a **NetworkMap** manifest to map the source and destination networks:

```
$ cat << EOF | oc apply -f -
apiVersion: forklift.konveyor.io/v1beta1
kind: NetworkMap
metadata:
  name: <network_map>
  namespace: openshift-mtv
spec:
  map:
    - destination:
      name: <pod>
      namespace: openshift-mtv
      type: pod 1
      source: 2
        id: <source_network_id> 3
        name: <source_network_name>
    - destination:
      name: <network_attachment_definition> 4
      namespace: <network_attachment_definition_namespace> 5
      type: multus
      source:
        id: <source_network_id>
        name: <source_network_name>
  provider:
    source:
      name: <source_provider>
      namespace: openshift-mtv
    destination:
      name: <destination_cluster>
      namespace: openshift-mtv
EOF
```

- 1 Allowed values are **pod** and **multus**.
- 2 You can use either the **id** or the **name** parameter to specify the source network.
- 3 Specify the VMware network MOR or RHV network UUID.
- 4 Specify a network attachment definition for each additional OpenShift Virtualization network.
- 5 Specify the namespace of the OpenShift Virtualization network attachment definition.

6. Create a **StorageMap** manifest to map source and destination storage:

```
$ cat << EOF | oc apply -f -
apiVersion: forklift.konveyor.io/v1beta1
kind: StorageMap
metadata:
  name: <storage_map>
  namespace: openshift-mtv
spec:
  map:
    - destination:
      storageClass: <storage_class>
      accessMode: <access_mode> 1
      source:
        id: <source_datastore> 2
    - destination:
      storageClass: <storage_class>
      accessMode: <access_mode>
      source:
        id: <source_datastore>
  provider:
    source:
      name: <source_provider>
      namespace: openshift-mtv
    destination:
      name: <destination_cluster>
      namespace: openshift-mtv
EOF
```

- 1 Allowed values are **ReadWriteOnce** and **ReadWriteMany**.
- 2 Specify the VMware data storage MOR or RHV storage domain UUID, for example, **f2737930-b567-451a-9ceb-2887f6207009**.

7. Optional: Create a **Hook** manifest to run custom code on a VM during the phase specified in the **Plan** CR:

```
$ cat << EOF | oc apply -f -
apiVersion: forklift.konveyor.io/v1beta1
kind: Hook
metadata:
  name: <hook>
  namespace: openshift-mtv
spec:
  image: quay.io/konveyor/hook-runner 1
  playbook: | 2

LS0tCi0gYmFtZTogTWFFbGogIGhvc3RzOiBsb2NhbGhvc3QKICB0YXNrczoKICAtIG5hbWU6I
Exv

YWQgUGxhbgogICAgW5jbHVkZV92YXJzOgogICAgICBmaWxIOiAiL3RtcC9ob29rL3BsYW4u
eW1s

lgogICAgICBuYW11OiBwbGFuCiAgLSBuYW11OiBMb2FkIFdvcmtsbn2FkCiAgICBpbmNsdWRlX
```



```
3Zh
```

```
cnM6CiAgICAgIGZpbGU6IClvdG1wL2hvb2svd29ya2xvYWQueW1slgogICAgICBuYW1lOiB3b
3Jr
  bG9hZAoK
EOF
```

- 1 You can use the default **hook-runner** image or specify a custom image. If you specify a custom image, you do not have to specify a playbook.
- 2 Optional: Base64-encoded Ansible playbook. If you specify a playbook, the **image** must be **hook-runner**.

8. Create a **Plan** manifest for the migration:

```
$ cat << EOF | oc apply -f -
apiVersion: forklift.konveyor.io/v1beta1
kind: Plan
metadata:
  name: <plan> 1
  namespace: openshift-mtv
spec:
  warm: true 2
  provider:
    source:
      name: <source_provider>
      namespace: openshift-mtv
    destination:
      name: <destination_cluster>
      namespace: openshift-mtv
  map:
    network: 3
      name: <network_map> 4
      namespace: openshift-mtv
    storage:
      name: <storage_map> 5
      namespace: openshift-mtv
  targetNamespace: openshift-mtv
  vms: 6
    - id: <source_vm> 7
    - name: <source_vm>
      hooks: 8
        - hook:
            namespace: openshift-mtv
            name: <hook> 9
            step: <step> 10
EOF
```

- 1 Specify the name of the **Plan** CR.
- 2 VMware only: Specify whether the migration is warm or cold. If you specify a warm migration without specifying a value for the **cutover** parameter in the **Migration** manifest, only the precopy stage will run. Warm migration is not supported for RHV.

- 3 You can add multiple network mappings.
- 4 Specify the name of the **NetworkMap** CR.
- 5 Specify the name of the **StorageMap** CR.
- 6 You can use either the **id** or the **name** parameter to specify the source VMs.
- 7 Specify the VMware VM MOR or RHV VM UUID.
- 8 Optional: You can specify up to two hooks for a VM. Each hook must run during a separate migration step.
- 9 Specify the name of the **Hook** CR.
- 10 Allowed values are **PreHook**, before the migration plan starts, or **PostHook**, after the migration is complete.

9. Create a **Migration** manifest to run the **Plan** CR:

```
$ cat << EOF | oc apply -f -
apiVersion: forklift.konveyor.io/v1beta1
kind: Migration
metadata:
  name: <migration> 1
  namespace: openshift-mtv
spec:
  plan:
    name: <plan> 2
    namespace: openshift-mtv
    cutover: <cutover_time> 3
EOF
```

- 1 Specify the name of the **Migration** CR.
- 2 Specify the name of the **Plan** CR that you are running. The **Migration** CR creates a **VirtualMachine** CR for each VM that is migrated.
- 3 Optional: Specify a cutover time according to the ISO 8601 format with the UTC time offset, for example, **2021-04-04T01:23:45.678+09:00**.

You can associate multiple **Migration** CRs with a single **Plan** CR. If a migration does not complete, you can create a new **Migration** CR, without changing the **Plan** CR, to migrate the remaining VMs.

10. Retrieve the **Migration** CR to monitor the progress of the migration:

```
$ oc get migration/<migration> -n openshift-mtv -o yaml
```

5.2. CANCELING A MIGRATION

You can cancel an entire migration or individual virtual machines (VMs) while a migration is in progress from the command line interface (CLI).

Canceling an entire migration

- Delete the **Migration** CR:

```
$ oc delete migration <migration> -n openshift-mtv 1
```

- 1 Specify the name of the **Migration** CR.

Canceling the migration of individual VMs

1. Add the individual VMs to the **spec.cancel** block of the **Migration** manifest:

```
$ cat << EOF | oc apply -f -
apiVersion: forklift.konveyor.io/v1beta1
kind: Migration
metadata:
  name: <migration>
  namespace: openshift-mtv
...
spec:
  cancel:
    - id: vm-102 1
    - id: vm-203
    - name: rhel8-vm
EOF
```

- 1 You can specify a VM by using the **id** key or the **name** key.

The value of the **id** key is the *managed object reference*, for a VMware VM, or the *VM UUID*, for a RHV VM.

2. Retrieve the **Migration** CR to monitor the progress of the remaining VMs:

```
$ oc get migration/<migration> -n openshift-mtv -o yaml
```

CHAPTER 6. ADVANCED MIGRATION OPTIONS

6.1. CHANGING PRECOPY INTERVALS FOR WARM MIGRATION

You can change the snapshot interval by patching the **ForkliftController** custom resource (CR).

Procedure

- Patch the **ForkliftController** CR:

```
$ oc patch forkliftcontroller/<forklift-controller> -n openshift-mtv -p '{"spec":
{"controller_precopy_interval": <60>}}' --type=merge 1
```

- 1 Specify the precopy interval in minutes. The default value is **60**.

You do not need to restart the **forklift-controller** pod.

6.2. CREATING CUSTOM RULES FOR THE VALIDATION SERVICE

The **Validation** service uses Open Policy Agent (OPA) policy rules to check the suitability of each virtual machine (VM) for migration. The **Validation** service generates a list of *concerns* for each VM, which are stored in the **Provider Inventory** service as VM attributes. The web console displays the concerns for each VM in the provider inventory.

You can create custom rules to extend the default ruleset of the **Validation** service. For example, you can create a rule that checks whether a VM has multiple disks.

6.2.1. About Rego files

Validation rules are written in [Rego](#), the Open Policy Agent (OPA) native query language. The rules are stored as **.rego** files in the `/usr/share/opa/policies/io/konveyor/forklift/<provider>` directory of the **Validation** pod.

Each validation rule is defined in a separate **.rego** file and tests for a specific condition. If the condition evaluates as **true**, the rule adds a **{“category”, “label”, “assessment”}** hash to the **concerns**. The **concerns** content is added to the **concerns** key in the inventory record of the VM. The web console displays the content of the **concerns** key for each VM in the provider inventory.

The following **.rego** file example checks for distributed resource scheduling enabled in the cluster of a VMware VM:

drs_enabled.rego example

```
package io.konveyor.forklift.vmware 1

has_drs_enabled {
  input.host.cluster.drsEnabled 2
}

concerns[flag] {
  has_drs_enabled
  flag := {
```

```

    "category": "Information",
    "label": "VM running in a DRS-enabled cluster",
    "assessment": "Distributed resource scheduling is not currently supported by OpenShift
Virtualization. The VM can be migrated but it will not have this feature in the target environment."
  }
}

```

- 1 Each validation rule is defined within a package. The package namespaces are **io.konveyor.forklift.vmware** for VMware and **io.konveyor.forklift.ovirt** for Red Hat Virtualization.
- 2 Query parameters are based on the **input** key of the **Validation** service JSON.

6.2.2. Checking the default validation rules

Before you create a custom rule, you must check the default rules of the **Validation** service to ensure that you do not create a rule that redefines an existing default value.

Example: If a default rule contains the line **default valid_input = false** and you create a custom rule that contains the line **default valid_input = true**, the **Validation** service will not start.

Procedure

1. Connect to the terminal of the **Validation** pod:

```
$ oc rsh <validation_pod>
```

2. Go to the OPA policies directory for your provider:

```
$ cd /usr/share/opa/policies/io/konveyor/forklift/<provider> 1
```

- 1 Specify **vmware** or **ovirt**.

3. Search for the default policies:

```
$ grep -R "default" *
```

6.2.3. Retrieving the Inventory service JSON

You retrieve the **Inventory** service JSON by sending an **Inventory** service query to a virtual machine (VM). The output contains an **"input"** key, which contains the inventory attributes that are queried by the **Validation** service rules.

You can create a validation rule based on any attribute in the **"input"** key, for example, **input.snapshot.kind**.

Procedure

1. Retrieve the **Inventory** service route:

```
$ oc get route <inventory_service> -n openshift-mtv
```

- Retrieve the **UUID** of a provider:

```
$ GET https://<inventory_service_route>/providers/<provider> 1
```

- Allowed values for the provider are **vsphere** and **ovirt**.

- Retrieve the VMs of a provider:

```
$ GET https://<inventory_service_route>/providers/<provider>/<UUID>/vms
```

- Retrieve the details of a VM:

```
$ GET https://<inventory_service_route>/providers/<provider>/<UUID>/workloads/<vm>
```

Example output

```
{
  "input": {
    "selfLink": "providers/vsphere/c872d364-d62b-46f0-bd42-16799f40324e/workloads/vm-431",
    "id": "vm-431",
    "parent": {
      "kind": "Folder",
      "id": "group-v22"
    },
    "revision": 1,
    "name": "iscsi-target",
    "revisionValidated": 1,
    "isTemplate": false,
    "networks": [
      {
        "kind": "Network",
        "id": "network-31"
      },
      {
        "kind": "Network",
        "id": "network-33"
      }
    ],
    "disks": [
      {
        "key": 2000,
        "file": "[iSCSI_Datastore] iscsi-target/iscsi-target-000001.vmdk",
        "datastore": {
          "kind": "Datastore",
          "id": "datastore-63"
        },
        "capacity": 17179869184,
        "shared": false,
        "rdm": false
      },
      {
        "key": 2001,
        "file": "[iSCSI_Datastore] iscsi-target/iscsi-target_1-000001.vmdk",
```

```

    "datastore": {
      "kind": "Datastore",
      "id": "datastore-63"
    },
    "capacity": 10737418240,
    "shared": false,
    "rdm": false
  }
],
"concerns": [],
"policyVersion": 5,
"uuid": "42256329-8c3a-2a82-54fd-01d845a8bf49",
"firmware": "bios",
"powerState": "poweredOn",
"connectionState": "connected",
"snapshot": {
  "kind": "VirtualMachineSnapshot",
  "id": "snapshot-3034"
},
"changeTrackingEnabled": false,
"cpuAffinity": [
  0,
  2
],
"cpuHotAddEnabled": true,
"cpuHotRemoveEnabled": false,
"memoryHotAddEnabled": false,
"faultToleranceEnabled": false,
"cpuCount": 2,
"coresPerSocket": 1,
"memoryMB": 2048,
"guestName": "Red Hat Enterprise Linux 7 (64-bit)",
"balloonedMemory": 0,
"ipAddress": "10.19.2.96",
"storageUsed": 30436770129,
"numaNodeAffinity": [
  "0",
  "1"
],
"devices": [
  {
    "kind": "RealUSBController"
  }
],
"host": {
  "id": "host-29",
  "parent": {
    "kind": "Cluster",
    "id": "domain-c26"
  },
  "revision": 1,
  "name": "IP address or host name of the vCenter host or RHV Engine host",
  "selfLink": "providers/vsphere/c872d364-d62b-46f0-bd42-16799f40324e/hosts/host-
29",
  "status": "green",
  "inMaintenance": false,

```

```

"managementServerIp": "10.19.2.96",
"thumbprint": <thumbprint>,
"timezone": "UTC",
"cpuSockets": 2,
"cpuCores": 16,
"productName": "VMware ESXi",
"productVersion": "6.5.0",
"networking": {
  "pNICs": [
    {
      "key": "key-vim.host.PhysicalNic-vmnic0",
      "linkSpeed": 10000
    },
    {
      "key": "key-vim.host.PhysicalNic-vmnic1",
      "linkSpeed": 10000
    },
    {
      "key": "key-vim.host.PhysicalNic-vmnic2",
      "linkSpeed": 10000
    },
    {
      "key": "key-vim.host.PhysicalNic-vmnic3",
      "linkSpeed": 10000
    }
  ],
  "vNICs": [
    {
      "key": "key-vim.host.VirtualNic-vmk2",
      "portGroup": "VM_Migration",
      "dPortGroup": "",
      "ipAddress": "192.168.79.13",
      "subnetMask": "255.255.255.0",
      "mtu": 9000
    },
    {
      "key": "key-vim.host.VirtualNic-vmk0",
      "portGroup": "Management Network",
      "dPortGroup": "",
      "ipAddress": "10.19.2.13",
      "subnetMask": "255.255.255.128",
      "mtu": 1500
    },
    {
      "key": "key-vim.host.VirtualNic-vmk1",
      "portGroup": "Storage Network",
      "dPortGroup": "",
      "ipAddress": "172.31.2.13",
      "subnetMask": "255.255.0.0",
      "mtu": 1500
    },
    {
      "key": "key-vim.host.VirtualNic-vmk3",
      "portGroup": "",
      "dPortGroup": "dvportgroup-48",
      "ipAddress": "192.168.61.13",

```



```

        "subnetMask": "255.255.255.0",
        "mtu": 1500
    },
    {
        "key": "key-vim.host.VirtualNic-vmk4",
        "portGroup": "VM_DHCP_Network",
        "dPortGroup": "",
        "ipAddress": "10.19.2.231",
        "subnetMask": "255.255.255.128",
        "mtu": 1500
    }
],
"portGroups": [
    {
        "key": "key-vim.host.PortGroup-VM Network",
        "name": "VM Network",
        "vSwitch": "key-vim.host.VirtualSwitch-vSwitch0"
    },
    {
        "key": "key-vim.host.PortGroup-Management Network",
        "name": "Management Network",
        "vSwitch": "key-vim.host.VirtualSwitch-vSwitch0"
    },
    {
        "key": "key-vim.host.PortGroup-VM_10G_Network",
        "name": "VM_10G_Network",
        "vSwitch": "key-vim.host.VirtualSwitch-vSwitch1"
    },
    {
        "key": "key-vim.host.PortGroup-VM_Storage",
        "name": "VM_Storage",
        "vSwitch": "key-vim.host.VirtualSwitch-vSwitch1"
    },
    {
        "key": "key-vim.host.PortGroup-VM_DHCP_Network",
        "name": "VM_DHCP_Network",
        "vSwitch": "key-vim.host.VirtualSwitch-vSwitch1"
    },
    {
        "key": "key-vim.host.PortGroup-Storage Network",
        "name": "Storage Network",
        "vSwitch": "key-vim.host.VirtualSwitch-vSwitch1"
    },
    {
        "key": "key-vim.host.PortGroup-VM_Isolated_67",
        "name": "VM_Isolated_67",
        "vSwitch": "key-vim.host.VirtualSwitch-vSwitch2"
    },
    {
        "key": "key-vim.host.PortGroup-VM_Migration",
        "name": "VM_Migration",
        "vSwitch": "key-vim.host.VirtualSwitch-vSwitch2"
    }
],
"switches": [
    {

```

```

    "key": "key-vim.host.VirtualSwitch-vSwitch0",
    "name": "vSwitch0",
    "portGroups": [
      "key-vim.host.PortGroup-VM Network",
      "key-vim.host.PortGroup-Management Network"
    ],
    "pNICs": [
      "key-vim.host.PhysicalNic-vmnic4"
    ]
  },
  {
    "key": "key-vim.host.VirtualSwitch-vSwitch1",
    "name": "vSwitch1",
    "portGroups": [
      "key-vim.host.PortGroup-VM_10G_Network",
      "key-vim.host.PortGroup-VM_Storage",
      "key-vim.host.PortGroup-VM_DHCP_Network",
      "key-vim.host.PortGroup-Storage Network"
    ],
    "pNICs": [
      "key-vim.host.PhysicalNic-vmnic2",
      "key-vim.host.PhysicalNic-vmnic0"
    ]
  },
  {
    "key": "key-vim.host.VirtualSwitch-vSwitch2",
    "name": "vSwitch2",
    "portGroups": [
      "key-vim.host.PortGroup-VM_Isolated_67",
      "key-vim.host.PortGroup-VM_Migration"
    ],
    "pNICs": [
      "key-vim.host.PhysicalNic-vmnic3",
      "key-vim.host.PhysicalNic-vmnic1"
    ]
  }
]
},
"networks": [
  {
    "kind": "Network",
    "id": "network-31"
  },
  {
    "kind": "Network",
    "id": "network-34"
  },
  {
    "kind": "Network",
    "id": "network-57"
  },
  {
    "kind": "Network",
    "id": "network-33"
  }
]

```

```

        "kind": "Network",
        "id": "dvportgroup-47"
    }
],
"datastores": [
    {
        "kind": "Datastore",
        "id": "datastore-35"
    },
    {
        "kind": "Datastore",
        "id": "datastore-63"
    }
],
"vms": null,
"networkAdapters": [],
"cluster": {
    "id": "domain-c26",
    "parent": {
        "kind": "Folder",
        "id": "group-h23"
    },
    "revision": 1,
    "name": "mycluster",
    "selfLink": "providers/vsphere/c872d364-d62b-46f0-bd42-
16799f40324e/clusters/domain-c26",
    "folder": "group-h23",
    "networks": [
        {
            "kind": "Network",
            "id": "network-31"
        },
        {
            "kind": "Network",
            "id": "network-34"
        },
        {
            "kind": "Network",
            "id": "network-57"
        },
        {
            "kind": "Network",
            "id": "network-33"
        },
        {
            "kind": "Network",
            "id": "dvportgroup-47"
        }
    ]
},
"datastores": [
    {
        "kind": "Datastore",
        "id": "datastore-35"
    },
    {
        "kind": "Datastore",

```

```

        "id": "datastore-63"
      }
    ],
    "hosts": [
      {
        "kind": "Host",
        "id": "host-44"
      },
      {
        "kind": "Host",
        "id": "host-29"
      }
    ],
    "dasEnabled": false,
    "dasVms": [],
    "drsEnabled": true,
    "drsBehavior": "fullyAutomated",
    "drsVms": [],
    "datacenter": null
  }
}
}
}

```

6.2.4. Creating a validation rule

You create a validation rule by applying a config map custom resource (CR) containing the rule to the **Validation** service.



IMPORTANT

- If you create a rule with the same *name* as an existing rule, the **Validation** service performs an **OR** operation with the rules.
- If you create a rule that contradicts a default rule, the **Validation** service will not start.

Validation rule example

Validation rules are based on virtual machine (VM) attributes collected by the **Provider Inventory** service.

For example, the VMware API uses this path to check whether a VMware VM has NUMA node affinity configured: **MOR:VirtualMachine.config.extraConfig["numa.nodeAffinity"]**.

The **Provider Inventory** service simplifies this configuration and returns a testable attribute with a list value:

```

"numaNodeAffinity": [
  "0",
  "1"
],

```

You create a [Rego](#) query, based on this attribute, and add it to the **forklift-validation-config** config map:

```
`count(input.numaNodeAffinity) != 0`
```

Procedure

1. Create a config map CR according to the following example:

```
$ cat << EOF | oc apply -f -
apiVersion: v1
kind: ConfigMap
metadata:
  name: <forklift-validation-config>
  namespace: openshift-mtv
data:
  vmware_multiple_disks.rego: |-
    package <provider_package> 1

    has_multiple_disks { 2
      count(input.disks) > 1
    }

    concerns[flag] {
      has_multiple_disks 3
      flag := {
        "category": "<Information>", 4
        "label": "Multiple disks detected",
        "assessment": "Multiple disks detected on this VM."
      }
    }
  }
EOF
```

- 1** Specify the provider package name. Allowed values are **io.konveyor.forklift.vmware** for VMware and **io.konveyor.forklift.ovirt** for Red Hat Virtualization.
- 2** Specify the **concerns** name and Rego query.
- 3** Specify the **concerns** name and **flag** parameter values.
- 4** Allowed values are **Critical**, **Warning**, and **Information**.

2. Stop the **Validation** pod by scaling the **forklift-controller** deployment to **0**:

```
$ oc scale -n openshift-mtv --replicas=0 deployment/forklift-controller
```

3. Start the **Validation** pod by scaling the **forklift-controller** deployment to **1**:

```
$ oc scale -n openshift-mtv --replicas=1 deployment/forklift-controller
```

4. Check the **Validation** pod log to verify that the pod started:

```
$ oc logs -f <validation_pod>
```

If the custom rule conflicts with a default rule, the **Validation** pod will not start.

- Remove the source provider:

```
$ oc delete provider <provider> -n openshift-mtv
```

- Add the source provider to apply the new rule:

```
$ cat << EOF | oc apply -f -
apiVersion: forklift.konveyor.io/v1beta1
kind: Provider
metadata:
  name: <provider>
  namespace: openshift-mtv
spec:
  type: <provider_type> 1
  url: <api_end_point> 2
  secret:
    name: <secret> 3
    namespace: openshift-mtv
EOF
```

- Allowed values are **ovirt** and **vsphere**.
- Specify the API end point URL, for example, **https://<vCenter_host>/sdk** for vSphere or **https://<engine_host>/ovirt-engine/api/** for RHV.
- Specify the name of the provider **Secret** CR.

You must update the rules version after creating a custom rule so that the **Inventory** service detects the changes and validates the VMs.

6.2.5. Updating the inventory rules version

You must update the inventory rules version each time you update the rules so that the **Provider Inventory** service detects the changes and triggers the **Validation** service.

The rules version is recorded in a **rules_version.rego** file for each provider.

Procedure

- Retrieve the current rules version:

```
$ GET https://forklift-validation/v1/data/io/konveyor/forklift/<provider>/rules_version 1
```

Example output

```
{
  "result": {
    "rules_version": 5
  }
}
```

- Connect to the terminal of the **Validation** pod:

■

```
$ oc rsh <validation_pod>
```

3. Update the rules version in the `/usr/share/opa/policies/io/konveyor/forklift/<provider>/rules_version.rego` file.
4. Log out of the **Validation** pod terminal.
5. Verify the updated rules version:

```
$ GET https://forklift-validation/v1/data/io/konveyor/forklift/<provider>/rules_version 1
```

Example output

```
{  
  "result": {  
    "rules_version": 6  
  }  
}
```

CHAPTER 7. UPGRADING THE MIGRATION TOOLKIT FOR VIRTUALIZATION

You can upgrade the MTV Operator by using the OpenShift Container Platform web console to install the new version.



NOTE

You must upgrade to the next release without skipping a release, for example, from 2.0 to 2.1 or from 2.1 to 2.2.

Procedure

1. In the OCP web console, click **Operators** → **Installed Operators** → **Migration Toolkit for Virtualization Operator** → **Subscription**.
2. Change the update channel to **release-v2.2.0**.
See [Changing update channel](#) in the OpenShift Container Platform documentation.
3. Confirm that **Upgrade status** changes from **Up to date** to **Upgrade available**. If it does not, restart the **CatalogSource** pod:

- a. Note the catalog source, for example, **redhat-operators**.

- b. From the command line, retrieve the catalog source pod:

```
$ oc get pod -n openshift-marketplace | grep <catalog_source> 1
```

1 1 1 Specify the catalog source, for example, **redhat-operators**.

- c. Delete the pod:

```
$ oc delete pod -n openshift-marketplace <catalog_source_pod>
```

Upgrade status changes from **Up to date** to **Upgrade available**.

If you set **Update approval** on the **Subscriptions** tab to **Automatic**, the upgrade starts automatically.

4. If you set **Update approval** on the **Subscriptions** tab to **Manual**, approve the upgrade.
See [Manually approving a pending upgrade](#) in the OpenShift Container Platform documentation.
5. Verify that the **forklift-ui** pod is in a **Ready** state before you log in to the web console:

```
$ oc get pods -n openshift-mtv
```

Example output

```
NAME                                READY STATUS RESTARTS AGE
forklift-controller-788bdb4c69-mw268 2/2   Running 0     2m
forklift-operator-6bf45b8d8-qps9v    1/1   Running 0     5m
forklift-ui-7cdf96d8f6-xnw5n        1/1   Running 0     2m
```


-

CHAPTER 8. UNINSTALLING THE MIGRATION TOOLKIT FOR VIRTUALIZATION

You can uninstall the Migration Toolkit for Virtualization (MTV) by using the OpenShift Container Platform web console or the command line interface (CLI).



8.1. UNINSTALLING MTV BY USING THE OPENSIFT CONTAINER PLATFORM WEB CONSOLE

You can uninstall Migration Toolkit for Virtualization (MTV) by using the OpenShift Container Platform web console to delete the **openshift-mtv** project and custom resource definitions (CRDs).

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.

Procedure

1. Click **Home** → **Projects**.
2. Locate the **openshift-mtv** project.
3. On the right side of the project, select **Delete Project** from the Options menu .
4. In the **Delete Project** pane, enter the project name and click **Delete**.
5. Click **Administration** → **CustomResourceDefinitions**.
6. Enter **forklift** in the **Search** field to locate the CRDs in the **forklift.konveyor.io** group.
7. On the right side of each CRD, select **Delete CustomResourceDefinition** from the Options menu .

8.2. UNINSTALLING MTV FROM THE COMMAND LINE INTERFACE

You can uninstall Migration Toolkit for Virtualization (MTV) from the command line interface (CLI) by deleting the **openshift-mtv** project and the **forklift.konveyor.io** custom resource definitions (CRDs).

Prerequisites

- You must be logged in as a user with **cluster-admin** privileges.

Procedure

1. Delete the project:

```
$ oc delete project openshift-mtv
```

2. Delete the CRDs:

```
$ oc get crd -o name | grep 'forklift' | xargs oc delete
```

3. Delete the OAuthClient:

```
┆ $ oc delete oauthclient/forklift-ui
```

CHAPTER 9. TROUBLESHOOTING

This section provides information for troubleshooting common migration issues.

9.1. ARCHITECTURE

This section describes MTV custom resources, services, and workflows.

9.1.1. MTV custom resources and services

The Migration Toolkit for Virtualization (MTV) is provided as an OpenShift Container Platform Operator. It creates and manages the following custom resources (CRs) and services.

MTV custom resources

- **Provider** CR stores attributes that enable MTV to connect to and interact with the source and target providers.
- **NetworkMapping** CR maps the networks of the source and target providers.
- **StorageMapping** CR maps the storage of the source and target providers.
- **Provisioner** CR stores the configuration of the storage provisioners, such as supported volume and access modes.
- **Plan** CR contains a list of VMs with the same migration parameters and associated network and storage mappings.
- **Migration** CR runs a migration plan.
Only one **Migration** CR per migration plan can run at a given time. You can create multiple **Migration** CRs for a single **Plan** CR.

MTV services

- The **Inventory** service performs the following actions:
 - Connects to the source and target providers.
 - Maintains a local inventory for mappings and plans.
 - Stores VM configurations.
 - Runs the **Validation** service if a VM configuration change is detected.
- The **Validation** service checks the suitability of a VM for migration by applying rules.
- The **User Interface** service performs the following actions:
 - Enables you to create and configure MTV CRs.
 - Displays the status of the CRs and the progress of a migration.
- The **Migration Controller** service orchestrates migrations.
When you create a migration plan, the **Migration Controller** service validates the plan and adds a status label. If the plan fails validation, the plan status is **Not ready** and the plan cannot be used to perform a migration. If the plan passes validation, the plan status is **Ready** and it can be

used to perform a migration. After a successful migration, the **Migration Controller** service changes the plan status to **Completed**.

- The **Kubevirt Controller** and **Containerized Data Import (CDI) Controller** services handle most technical operations.

9.1.2. High-level migration workflow

The high-level workflow shows the migration process from the point of view of the user:

1. You create a source provider, a target provider, a network mapping, and a storage mapping.
2. You create a **Plan** custom resource (CR) that includes the following resources:
 - Source provider
 - Target provider, if MTV is not installed on the target cluster
 - Network mapping
 - Storage mapping
 - One or more virtual machines (VMs)
3. You run a migration plan by creating a **Migration** CR that references the **Plan** CR. If you cannot migrate all the VMs for any reason, you can create multiple **Migration** CRs for the same **Plan** CR until all VMs are migrated.
4. For each VM in the **Plan** CR, the **Migration Controller** service creates a **VirtualMachine** CR and records the VM migration progress in the **Migration** CR. When all VMs have been migrated, the **Migration Controller** service updates the status of the **Plan** CR to **Completed**. The power state of each source VM is maintained after migration.

9.1.3. Detailed migration workflow

You can use the detailed migration workflow to troubleshoot a failed migration.

The workflow describes the following steps:

1. When you create a **Migration** custom resource (CR) to run a migration plan, the **Migration Controller** service creates a **VirtualMachine** CR for each source virtual machine (VM) and a **DataVolume** CR for each source VM disk.
For each VM disk:
2. The **Containerized Data Importer (CDI) Controller** service creates a persistent volume claim (PVC) based on the parameters specified in the **DataVolume** CR.
3. If the **StorageClass** has a dynamic provisioner, the persistent volume (PV) is dynamically provisioned by the **StorageClass** provisioner.
4. The **CDI Controller** service creates an **importer** pod.
5. The **importer** pod streams the VM disk to the PV.
After the VM disks are transferred:
6. The **Migration Controller** service creates a **conversion** pod with the PVCs attached to it.

The **conversion** pod runs **virt-v2v**, which installs and configures device drivers on the PVCs of the target VM.

- When the target VM is powered on, the **KubeVirt Controller** service creates a **virt-launcher** pod and a **VirtualMachineInstance** CR.
The **virt-launcher** pod runs **QEMU-KVM** with the PVCs attached as VM disks.

9.2. LOGS AND CUSTOM RESOURCES

You can download logs and custom resource (CR) information for troubleshooting. For more information, see the [detailed migration workflow](#).

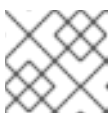
9.2.1. Collected logs and custom resource information

You can download logs and custom resource (CR) **yaml** files for the following targets by using the MTV web console or the command line interface (CLI):

- Migration plan: Web console or CLI.
- Virtual machine: Web console or CLI.
- Namespace: CLI only.

The **must-gather** tool collects the following logs and CR files in an archive file:

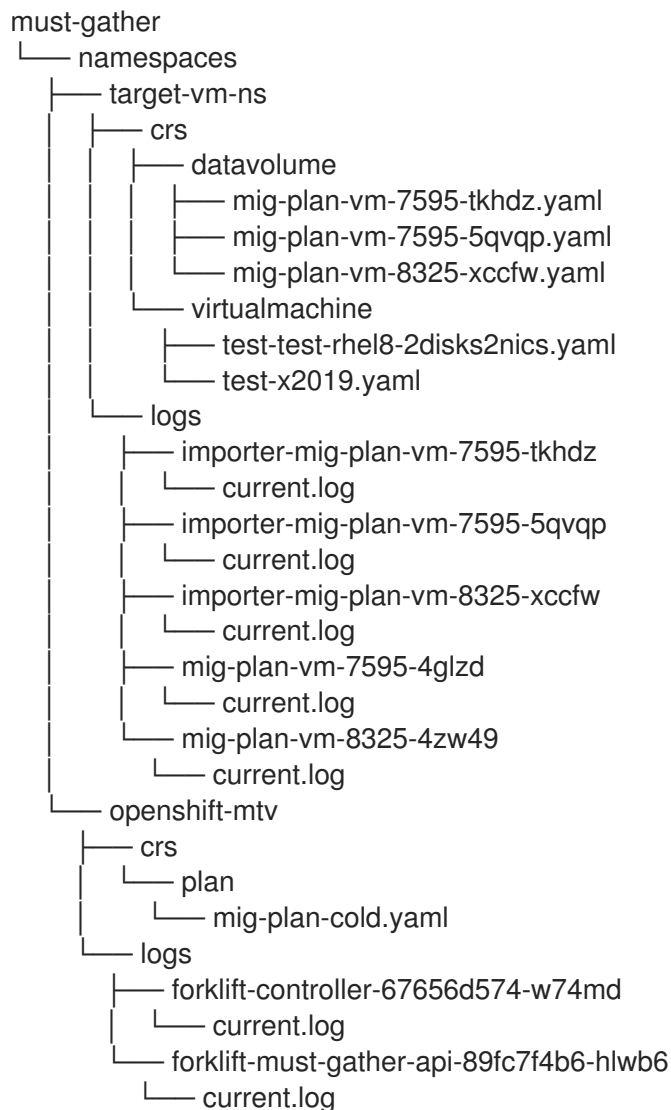
- CRs:
 - **DataVolume** CR: Represents a disk mounted on a migrated VM.
 - **VirtualMachine** CR: Represents a migrated VM.
 - **Plan** CR: Defines the VMs and storage and network mapping.
- Logs:
 - **importer** pod: Disk-to-data-volume conversion log. The **importer** pod naming convention is **importer-<migration_plan>-<vm_id><5_char_id>**, for example, **importer-mig-plan-ed90dfc6-9a17-4a8btnfh**, where **ed90dfc6-9a17-4a8** is a truncated RHV VM ID and **btnfh** is the generated 5-character ID.
 - **conversion** pod: VM conversion log. The **conversion** pod runs **virt-v2v**, which installs and configures device drivers on the PVCs of the VM. The **conversion** pod naming convention is **<migration_plan>-<vm_id><5_char_id>**.
 - **virt-launcher** pod: VM launcher log. When a migrated VM is powered on, the **virt-launcher** pod runs **QEMU-KVM** with the PVCs attached as VM disks.
 - **forklift-controller** pod: The log is filtered for the migration plan, virtual machine, or namespace specified by the **must-gather** command.
 - **forklift-must-gather-api** pod: The log is filtered for the migration plan, virtual machine, or namespace specified by the **must-gather** command.



NOTE

Empty or excluded log files are not included in the **must-gather** archive file.

Example must-gather archive structure for a VMware migration plan



9.2.2. Downloading logs and custom resource information from the web console

You can download logs and information about custom resources (CRs) for a completed, failed, or canceled migration plan or for migrated virtual machines (VMs) by using the MTV web console.

Procedure

1. In the web console, click **Migration plans**.
2. Click **Get logs** beside a migration plan name.
3. In the **Get logs** window, click **Get logs**.
The logs are collected. A **Log collection complete** message is displayed.
4. Click **Download logs** to download the archive file.
5. To download logs for a migrated VM, click a migration plan name and then click **Get logs** beside the VM.

9.2.3. Accessing logs and custom resource information from the command line interface

You can access logs and information about custom resources (CRs) from the command line interface by using the **must-gather** tool. You must attach a **must-gather** data file to all customer cases.

You can gather data for a specific namespace, a completed, failed, or canceled migration plan, or a migrated virtual machine (VM) by using the filtering options.



NOTE

If you specify a non-existent resource in the filtered **must-gather** command, no archive file is created.

Prerequisites

- You must be logged in to the OpenShift Virtualization cluster as a user with the **cluster-admin** role.
- You must have the [OpenShift Container Platform CLI \(oc\)](#) installed.

Procedure

1. Navigate to the directory where you want to store the **must-gather** data.
2. Run the **oc adm must-gather** command:

```
$ oc adm must-gather --image=registry.redhat.io/migration-toolkit-virtualization/mtv-must-gather-rhel8:2.2.0
```

The data is saved as **/must-gather/must-gather.tar.gz**. You can upload this file to a support case on the [Red Hat Customer Portal](#).

3. Optional: Run the **oc adm must-gather** command with the following options to gather filtered data:

- Namespace:

```
$ oc adm must-gather --image=registry.redhat.io/migration-toolkit-virtualization/mtv-must-gather-rhel8:2.2.0 \
  -- NS=<namespace> /usr/bin/targeted
```

- Migration plan:

```
$ oc adm must-gather --image=registry.redhat.io/migration-toolkit-virtualization/mtv-must-gather-rhel8:2.2.0 \
  -- PLAN=<migration_plan> /usr/bin/targeted
```

- Virtual machine:

```
$ oc adm must-gather --image=registry.redhat.io/migration-toolkit-virtualization/mtv-must-gather-rhel8:2.2.0 \
  -- VM=<vm_name> NS=<namespace> /usr/bin/targeted 1
```


- 1 You must specify the VM *name*, not the VM ID, as it appears in the **Plan** CR.