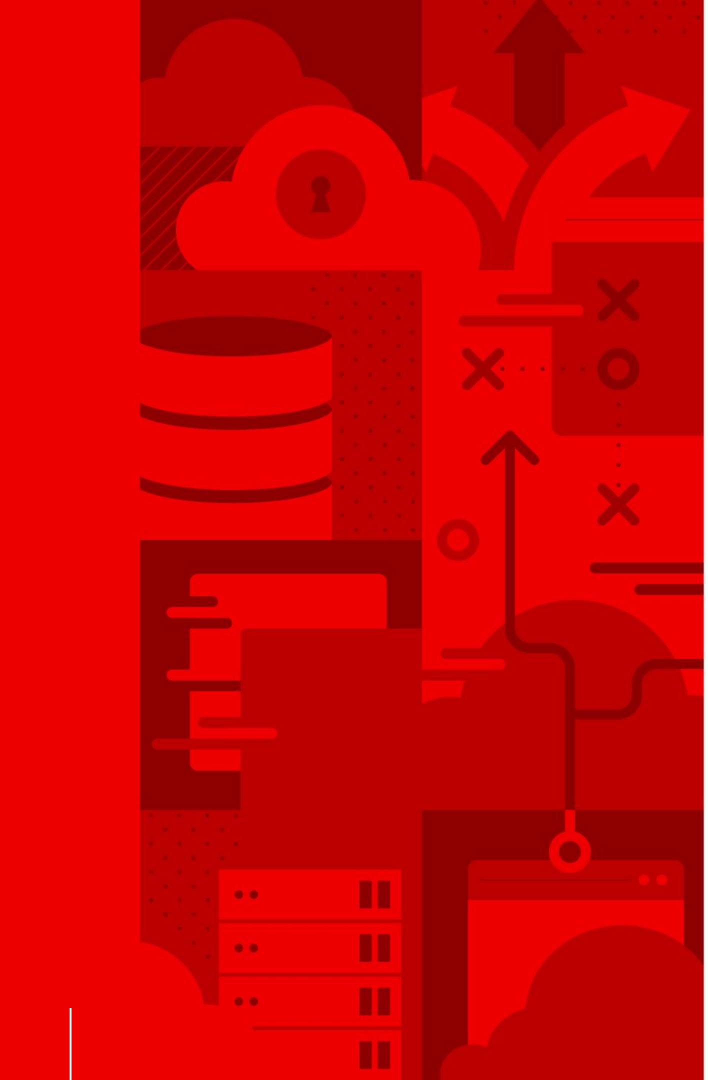
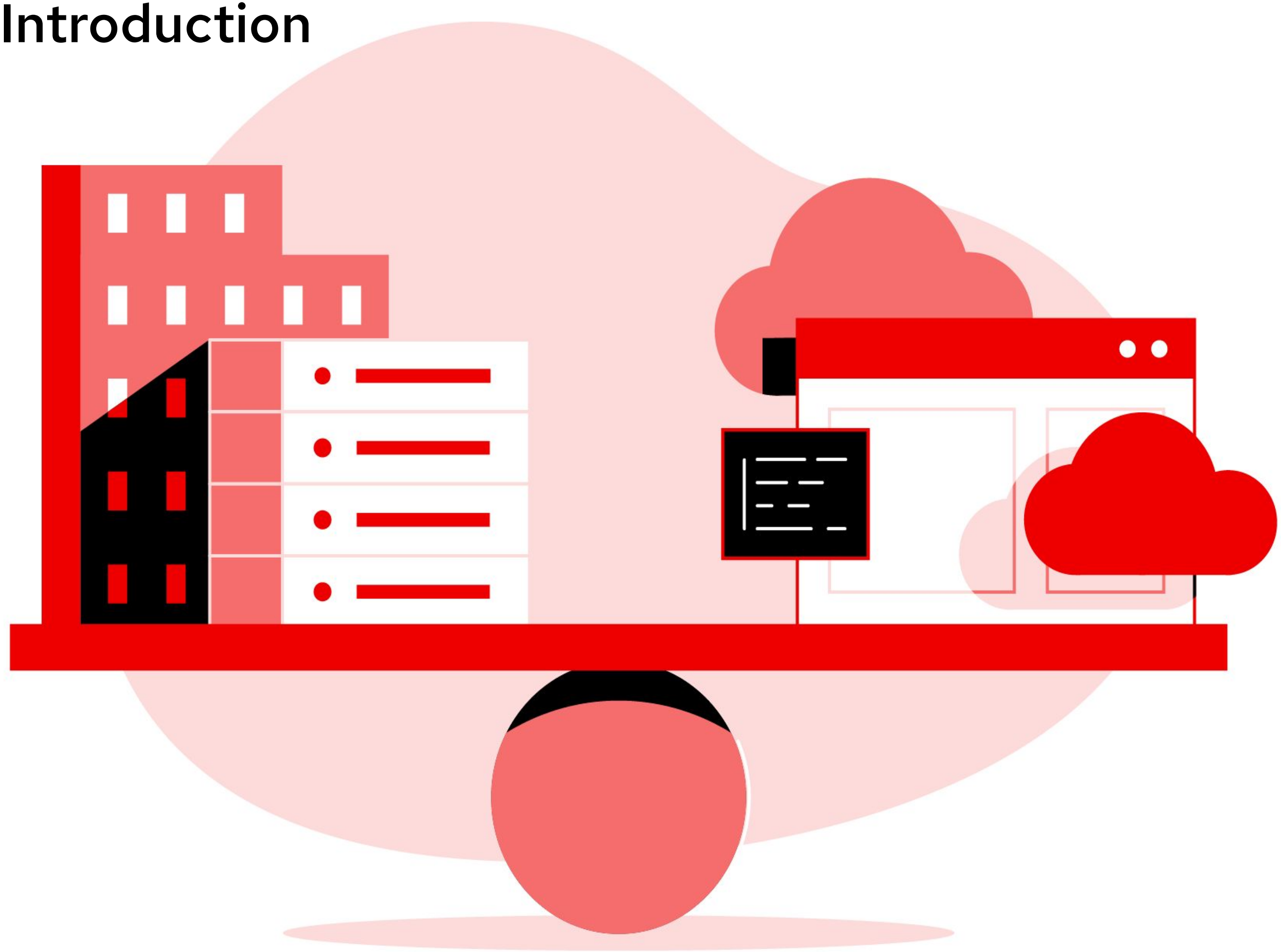


From Virtualization Platform to Hybrid Cloud Solution: A Hands-On Account

FOSDEM
February 3, 2024

Bella Khizgiyaev
bkhizgiy@redhat.com

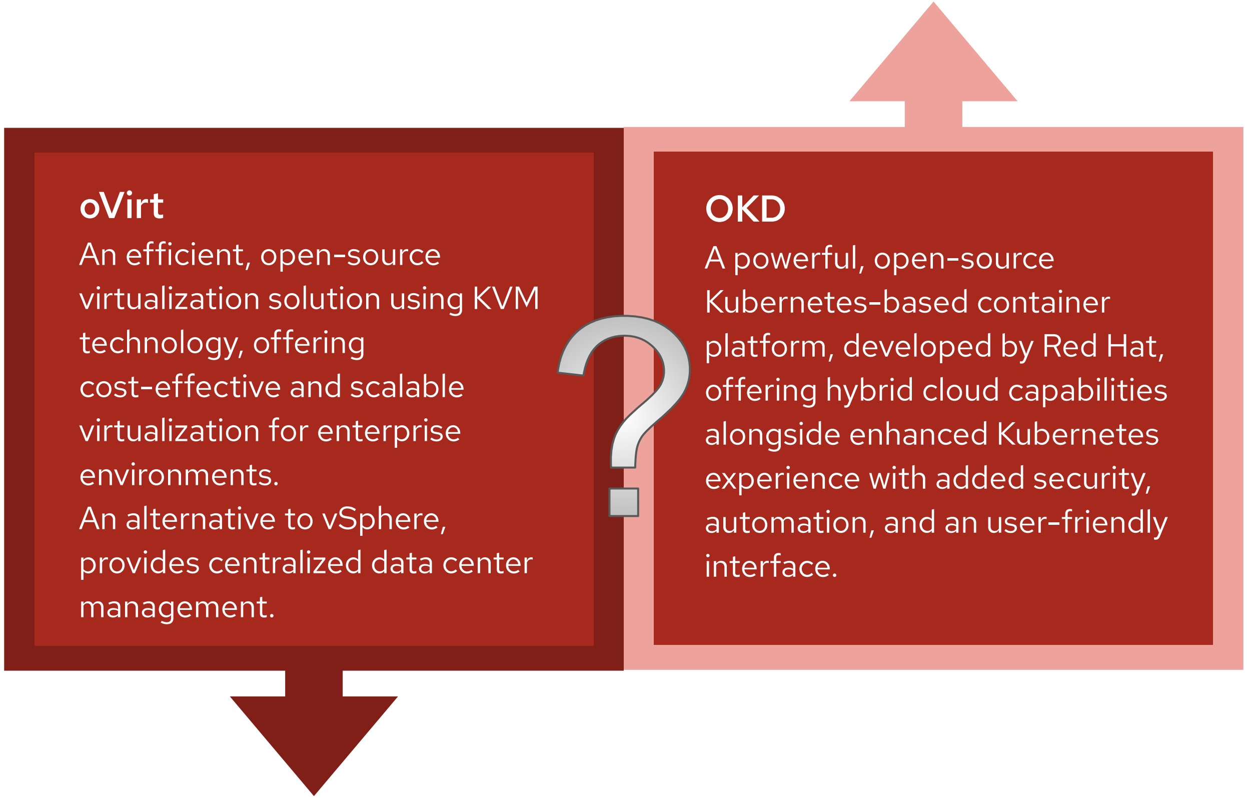
Introduction



Evolving IT Landscape

The ongoing shift from traditional virtualization platforms like oVirt to more dynamic, scalable hybrid cloud solutions, such as OKD.

Introduction



oVirt

An efficient, open-source virtualization solution using KVM technology, offering cost-effective and scalable virtualization for enterprise environments.

An alternative to vSphere, provides centralized data center management.

OKD

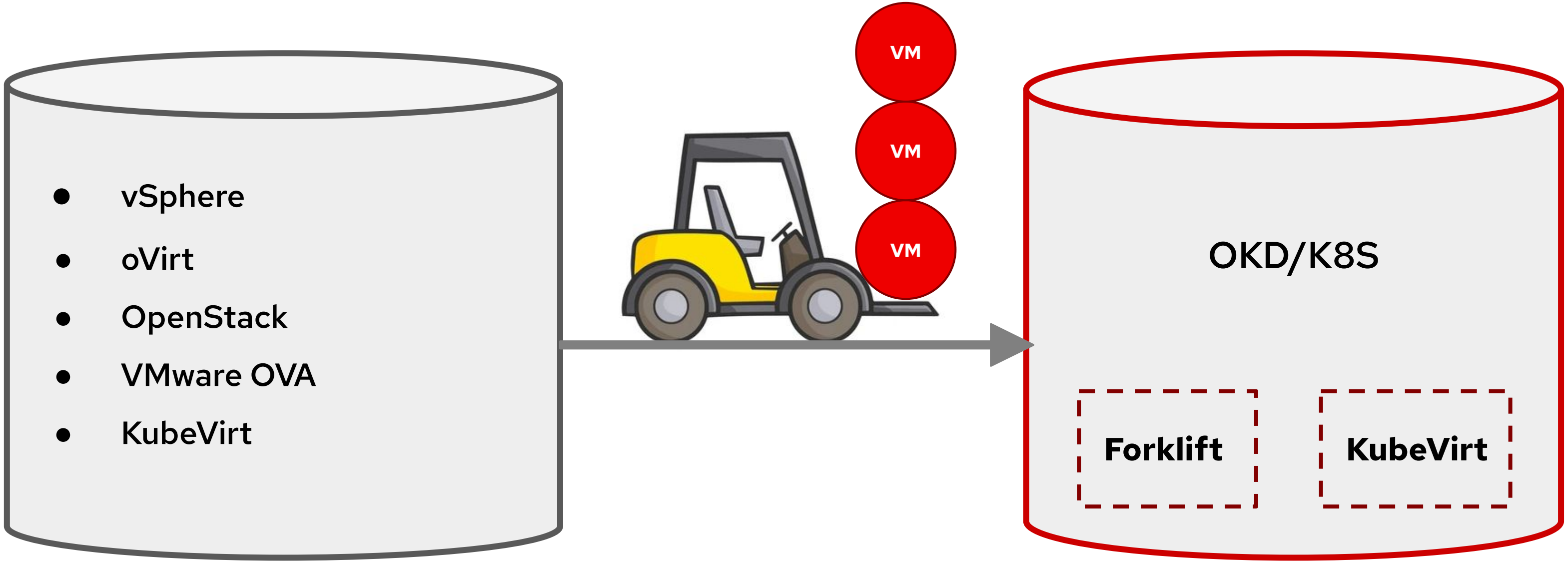
A powerful, open-source Kubernetes-based container platform, developed by Red Hat, offering hybrid cloud capabilities alongside enhanced Kubernetes experience with added security, automation, and an user-friendly interface.

Introduction

Different approaches for shifting virtual workloads

		Cost	Practicality
1	Re-provisioning everything	💰💰💰	😟
2	Migrate workloads into containers using *KONVEYOR*	💰💰	😟
3	Migrate existing workloads using *Forklift*	💰	😃

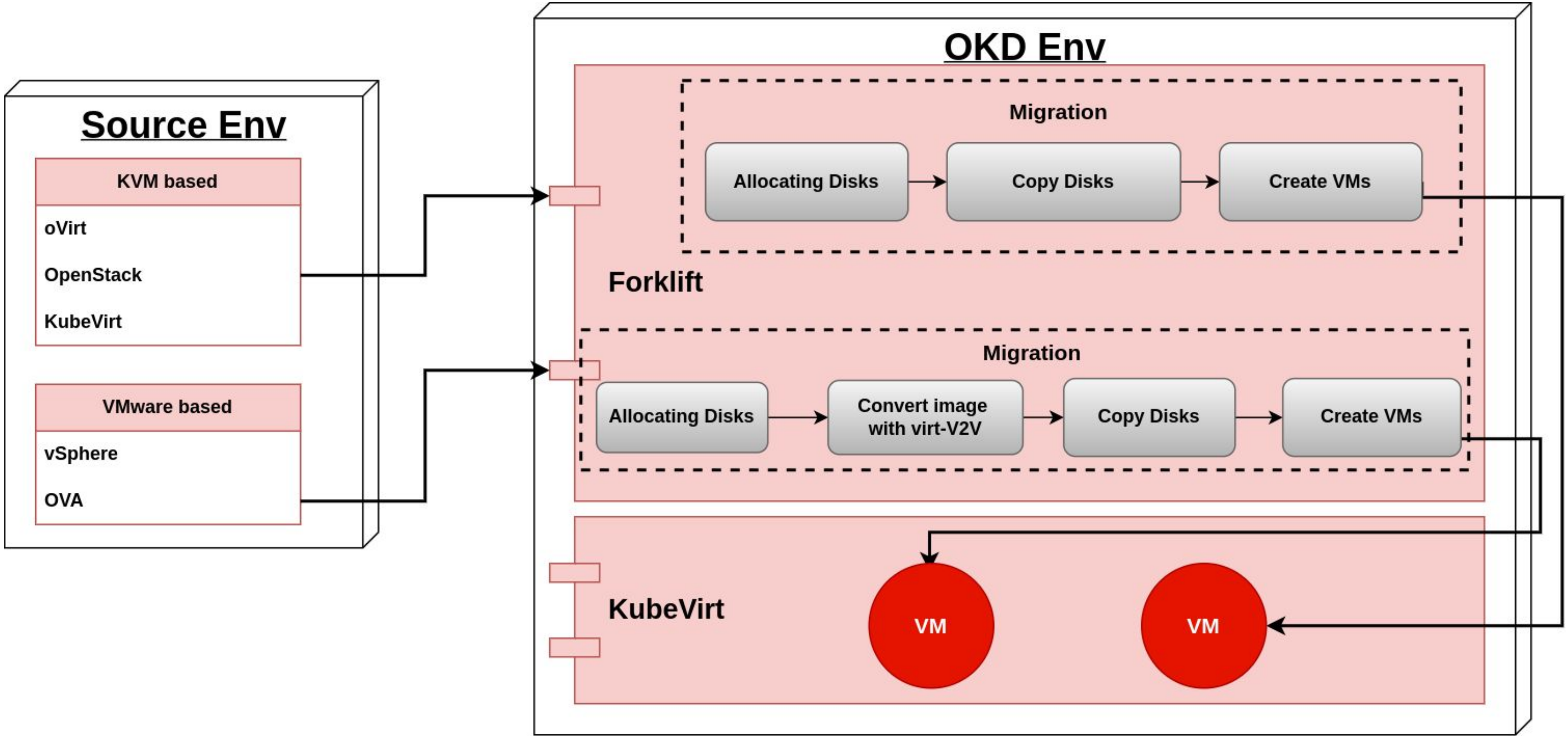
Forklift



- vSphere
- oVirt
- OpenStack
- VMware OVA
- KubeVirt

GitHub https://github.com/kubev2v/forklift

Forklift



GitHub https://github.com/kubev2v/forklift

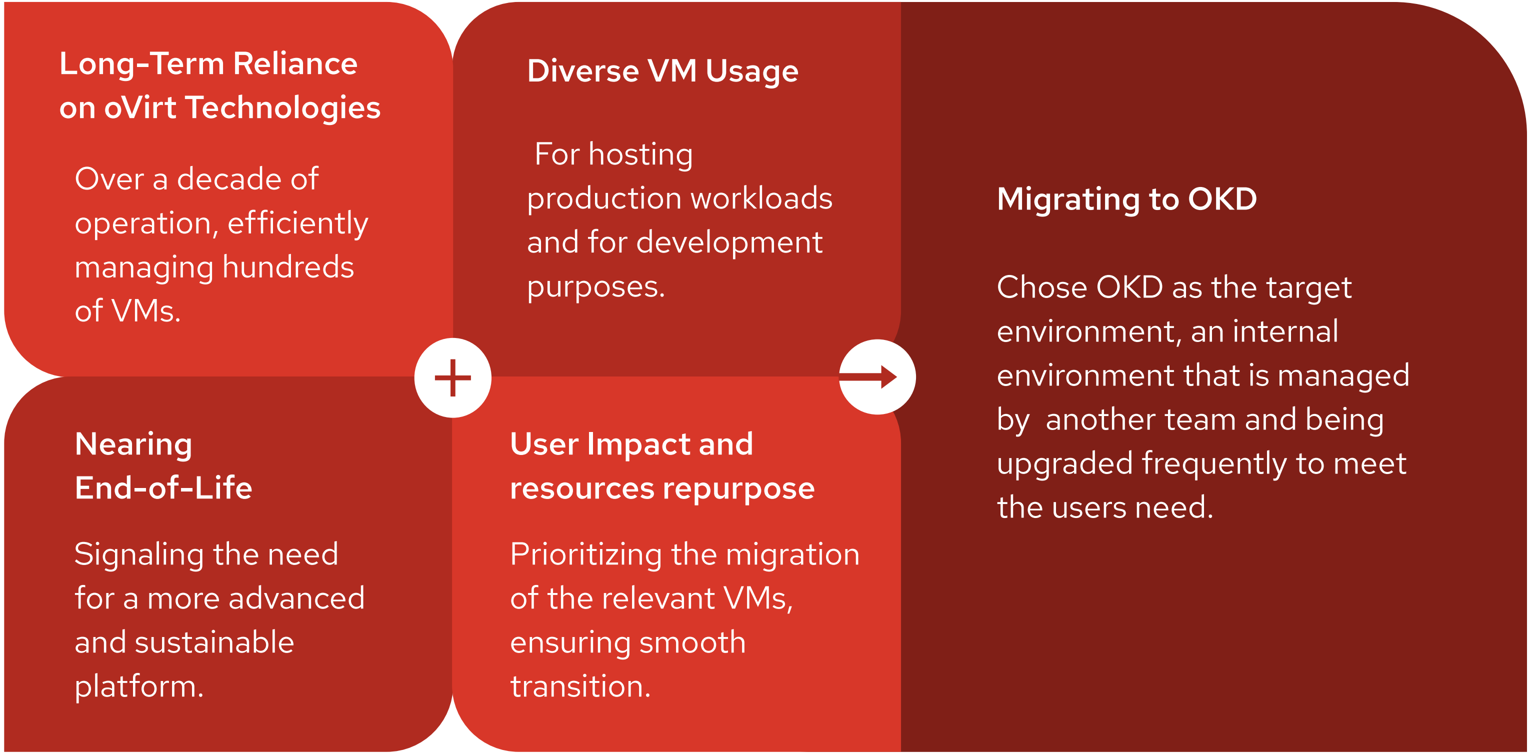
Our Transition Journey

Introducing the migration from oVirt to OKD

Utilizing ***Forklift*** for this Migration

Chosen as the preferred approach in our scenario, acting as a key tool in facilitating this transition.

Background



Long-Term Reliance on oVirt Technologies

Over a decade of operation, efficiently managing hundreds of VMs.

Diverse VM Usage

For hosting production workloads and for development purposes.

Nearing End-of-Life

Signaling the need for a more advanced and sustainable platform.

User Impact and resources repurpose

Prioritizing the migration of the relevant VMs, ensuring smooth transition.

Migrating to OKD

Chose OKD as the target environment, an internal environment that is managed by another team and being upgraded frequently to meet the users need.

Pre-Migration

Planing

Thorough Assessment

In-depth analysis of the existing oVirt environment to identify the scope and requirements for the migration.

Resource Evaluation

Ensuring the target OKD cluster has adequate resources (compute, storage, network) to accommodate the migrated workloads.

Timeline

Creating clear timeline and laying the steps for the transition.

Pre-Migration

Preparation & Resource Allocation

Migration Criteria and VM Selection

Determining the criteria for VM eligibility for migration.

- Identify active VMs based on their usage.
- Consult directly with VM owners.

Storage Planning

Assessing the storage needs by evaluating the capacity of the VMs planned for migration.

- Calculate the Disk size of the candidate VMs.

IP Address Allocation

Preparing for the allocation of enough IP addresses for the VMs in the target VLAN in OKD.

- Calculate the IPs assigned to the candidate VMs.

Permissions and ownership

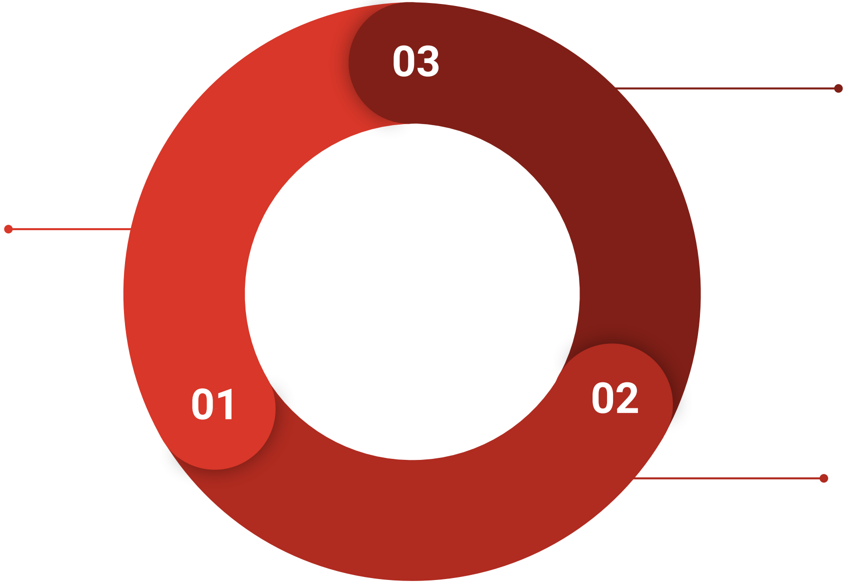
Our goal was to replicate the same ownership model within the new user-driven project framework in OKD.

- Get a list of the VM owners.
- Replicate the permissions to OKD.

A walk in the park?

Pre-Migration

Challenges



VM Selection

The difficulty in identifying the VMs for migration, especially determining which VMs were active.

Provisioning Model Differences

How to bridge the difference between oVirt's admin-assigned VM model to OKD's user-driven project and quota model.

Data Gathering Complexities

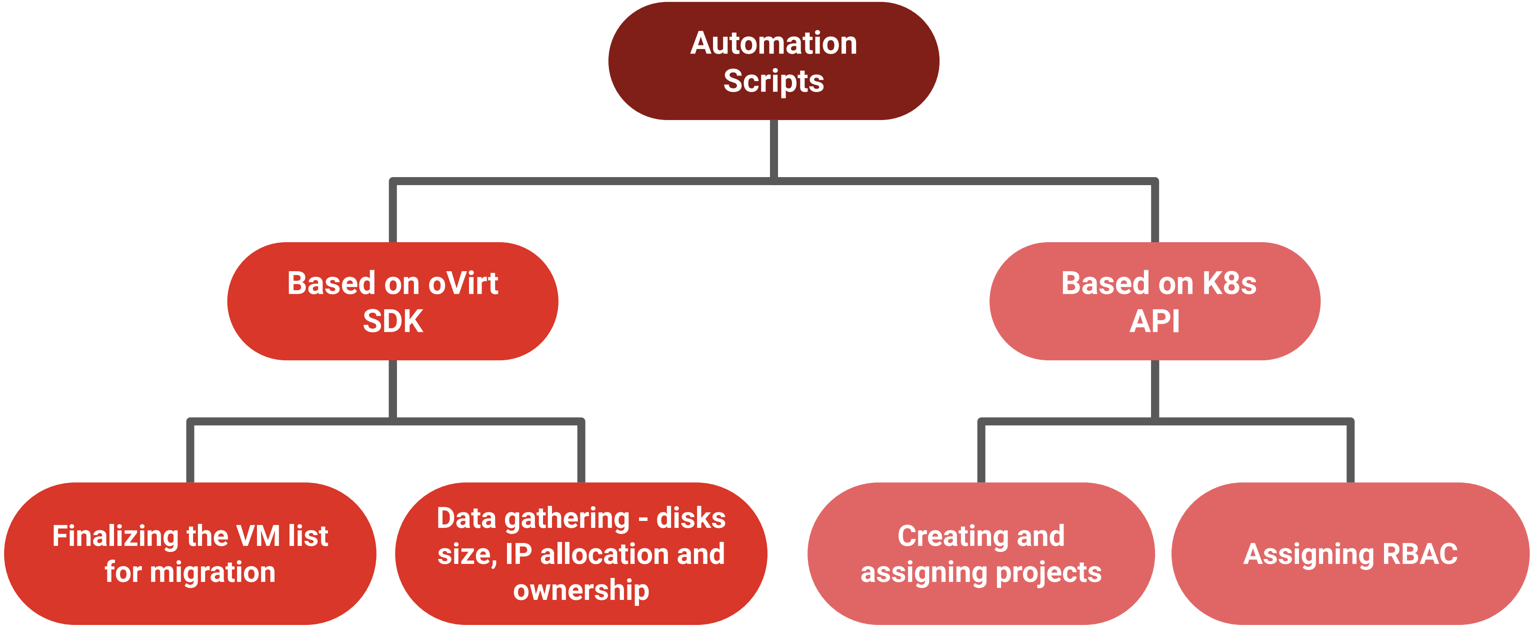
The challenges faced in gathering detailed information about the VMs, such as disk sizes and network information.

Red Hat

Pre-Migration

Scripts Implementation

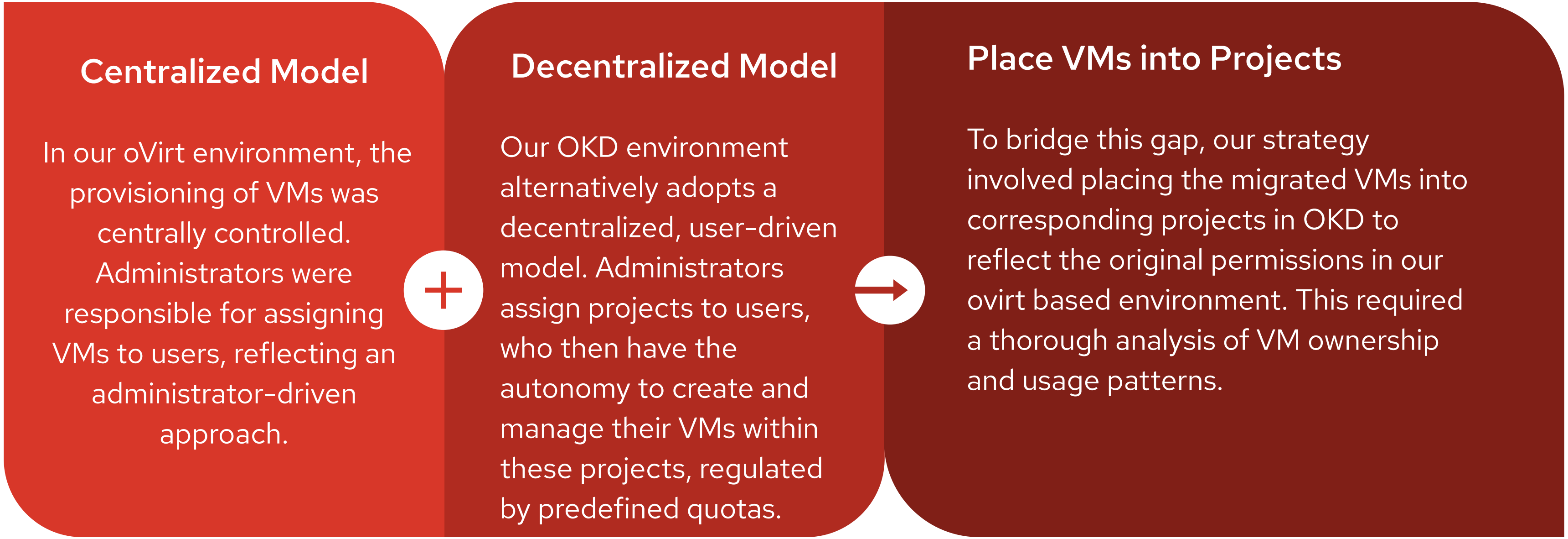
The Solution, developing Python scripts to automate data collection and project preparation, assist in manage the complexity and scale of the migration.


Automation Scripts
Based on oVirt SDK
Based on K8s API
Finalizing the VM list for migration
Data gathering - disks size, IP allocation and ownership
Creating and assigning projects
Assigning RBAC

The scripts https://github.com/kubev2v/rhv-ocp-migration-scripts

Pre-Migration

Bridging Between Provisioning Models



Centralized Model

In our oVirt environment, the provisioning of VMs was centrally controlled. Administrators were responsible for assigning VMs to users, reflecting an administrator-driven approach.

+

Decentralized Model

Our OKD environment alternatively adopts a decentralized, user-driven model. Administrators assign projects to users, who then have the autonomy to create and manage their VMs within these projects, regulated by predefined quotas.

→

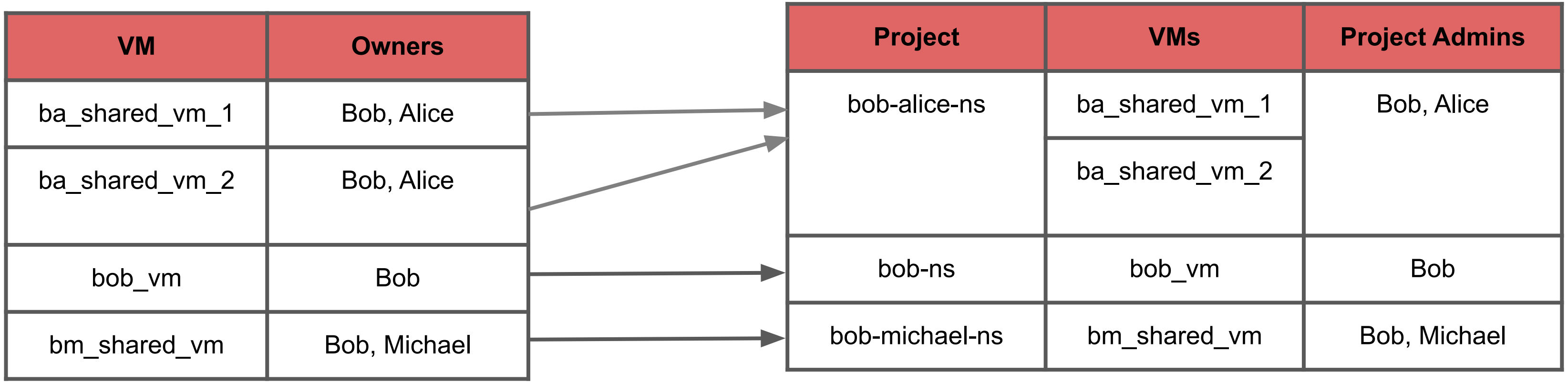
Place VMs into Projects

To bridge this gap, our strategy involved placing the migrated VMs into corresponding projects in OKD to reflect the original permissions in our ovirt based environment. This required a thorough analysis of VM ownership and usage patterns.

Pre-Migration

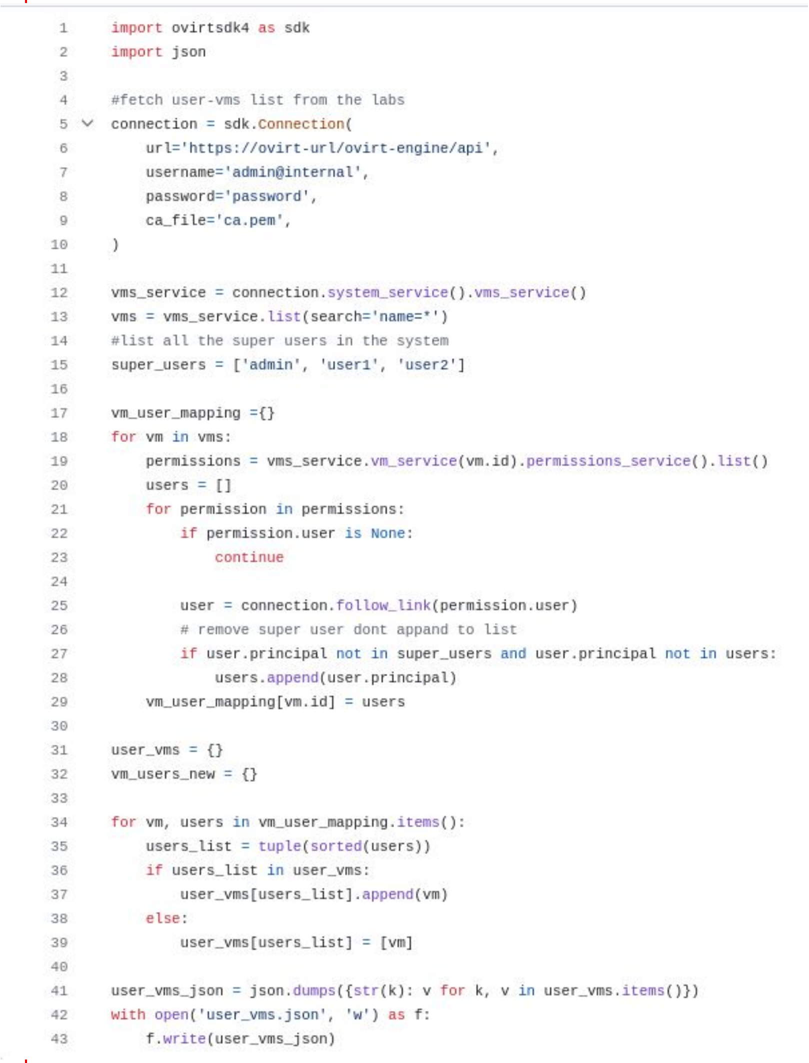
Example

For instance, a VM called ba_shared_vm_1 owned by two users, Alice and Bob, in RHV was migrated to a newly created project in OKD, named alice-bob-ns, where both users were assigned appropriate permissions.



VM	Owners
ba_shared_vm_1	Bob, Alice
ba_shared_vm_2	Bob, Alice
bob_vm	Bob
bm_shared_vm	Bob, Michael

Project	VMs	Project Admins
bob-alice-ns	ba_shared_vm_1	Bob, Alice
	ba_shared_vm_2	
bob-ns	bob_vm	Bob
bob-michael-ns	bm_shared_vm	Bob, Michael



```python
import ovirtsdk4 as sdk
import json

#fetch user-vms list from the labs
connection = sdk.Connection(
    url='https://ovirt-url/ovirt-engine/api',
    username='admin@internal',
    password='password',
    ca_file='ca.pem',
)

vms_service = connection.system_service().vms_service()
vms = vms_service.list(search='name=*')
#list all the super users in the system
super_users = ['admin', 'user1', 'user2']

vm_user_mapping ={}
for vm in vms:
    permissions = vms_service.vm_service(vm.id).permissions_service().list()
    users = []
    for permission in permissions:
        if permission.user is None:
            continue

        user = connection.follow_link(permission.user)
        # remove super user dont append to list
        if user.principal not in super_users and user.principal not in users:
            users.append(user.principal)
    vm_user_mapping[vm.id] = users

user_vms = {}
vm_users_new = {}

for vm, users in vm_user_mapping.items():
    users_list = tuple(sorted(users))
    if users_list in user_vms:
        user_vms[users_list].append(vm)
    else:
        user_vms[users_list] = [vm]

user_vms_json = json.dumps({str(k): v for k, v in user_vms.items()})
with open('user_vms.json', 'w') as f:
    f.write(user_vms_json)
```

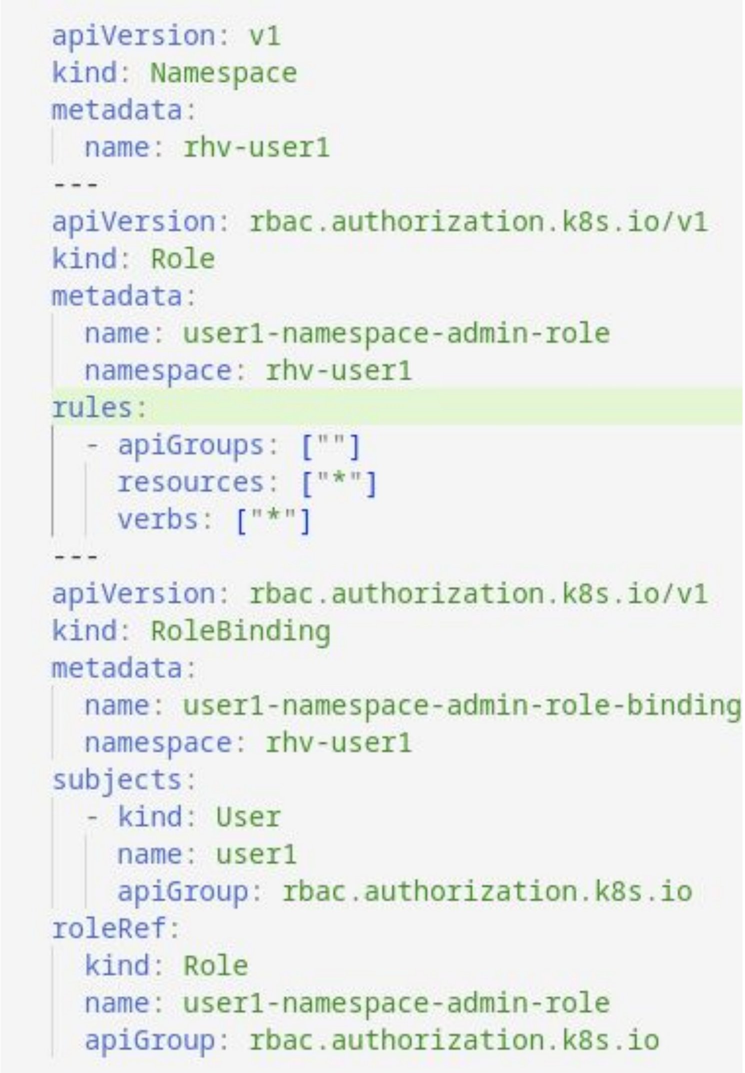
Pre-Migration

Bridging Provisioning Models

Script Functionality:

The scripts executed the following tasks.

1. Identified the VMs owners in RHV.
2. Removed admin/system users from that list.
3. Mapped between VM sets to users.



```yaml
apiVersion: v1
kind: Namespace
metadata:
  name: rhv-user1
---
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: user1-namespace-admin-role
  namespace: rhv-user1
rules:
  - apiGroups: [""]
    resources: ["*"]
    verbs: ["*"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: user1-namespace-admin-role-binding
  namespace: rhv-user1
subjects:
  - kind: User
    name: user1
    apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: user1-namespace-admin-role
  apiGroup: rbac.authorization.k8s.io
```

Pre-Migration

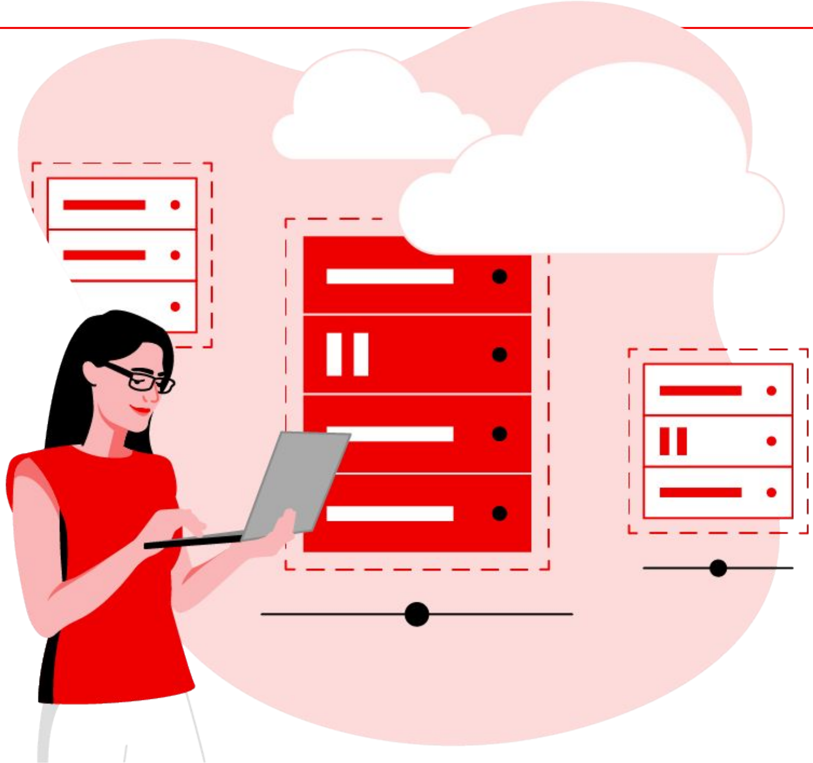
Bridging Provisioning Models

YAML Functionality:

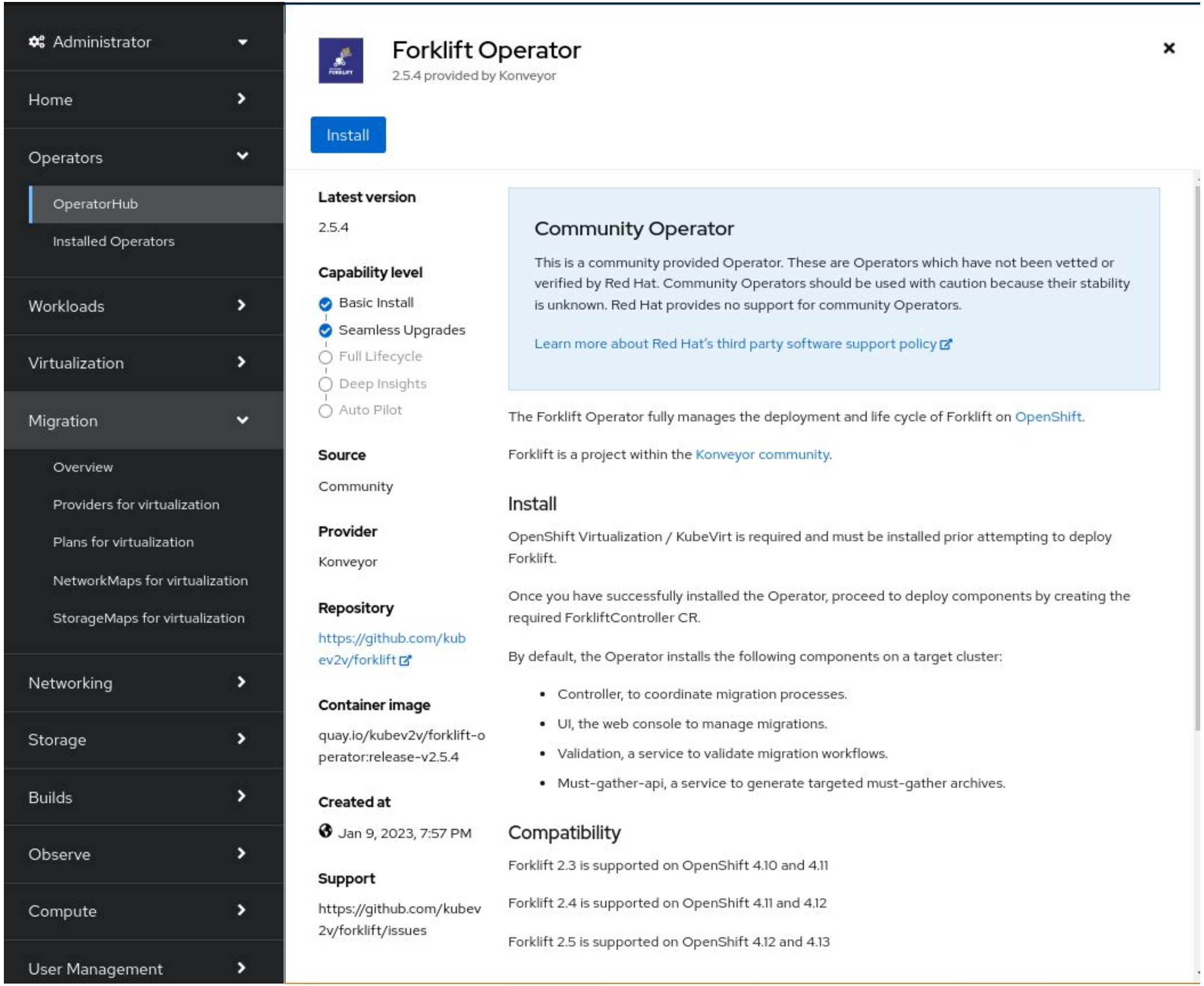
Executed the following tasks.

1. A YAML file containing projects configuration has been generated as the output of the script.
2. Created the corresponding Projects.
3. Assigned the User with the relevant RBAC to the project.

Migration Execution



Red Hat



Forklift Operator

2.5.4 provided by Konveyor

Latest version

2.5.4

Capability level

- ✓ Basic Install
- ✓ Seamless Upgrades
- ○ Full Lifecycle
- ○ Deep Insights
- ○ Auto Pilot

Source

Community

Provider

Konveyor

Repository

https://github.com/kubev2v/forklift

Container image

quay.io/kubev2v/forklift-operator:release-v2.5.4

Created at

Jan 9, 2023, 7:57 PM

Support

https://github.com/kubev2v/forklift/issues

Community Operator

This is a community provided Operator. These are Operators which have not been vetted or verified by Red Hat. Community Operators should be used with caution because their stability is unknown. Red Hat provides no support for community Operators.

Learn more about Red Hat's third party software support policy

The Forklift Operator fully manages the deployment and life cycle of Forklift on OpenShift.

Forklift is a project within the Konveyor community.

Install

OpenShift Virtualization / KubeVirt is required and must be installed prior attempting to deploy Forklift.

Once you have successfully installed the Operator, proceed to deploy components by creating the required ForkliftController CR.

By default, the Operator installs the following components on a target cluster:

- Controller, to coordinate migration processes.
- UI, the web console to manage migrations.
- Validation, a service to validate migration workflows.
- Must-gather-api, a service to generate targeted must-gather archives.

Compatibility

Forklift 2.3 is supported on OpenShift 4.10 and 4.11

Forklift 2.4 is supported on OpenShift 4.11 and 4.12

Forklift 2.5 is supported on OpenShift 4.12 and 4.13

Migration

Deploying Forklift

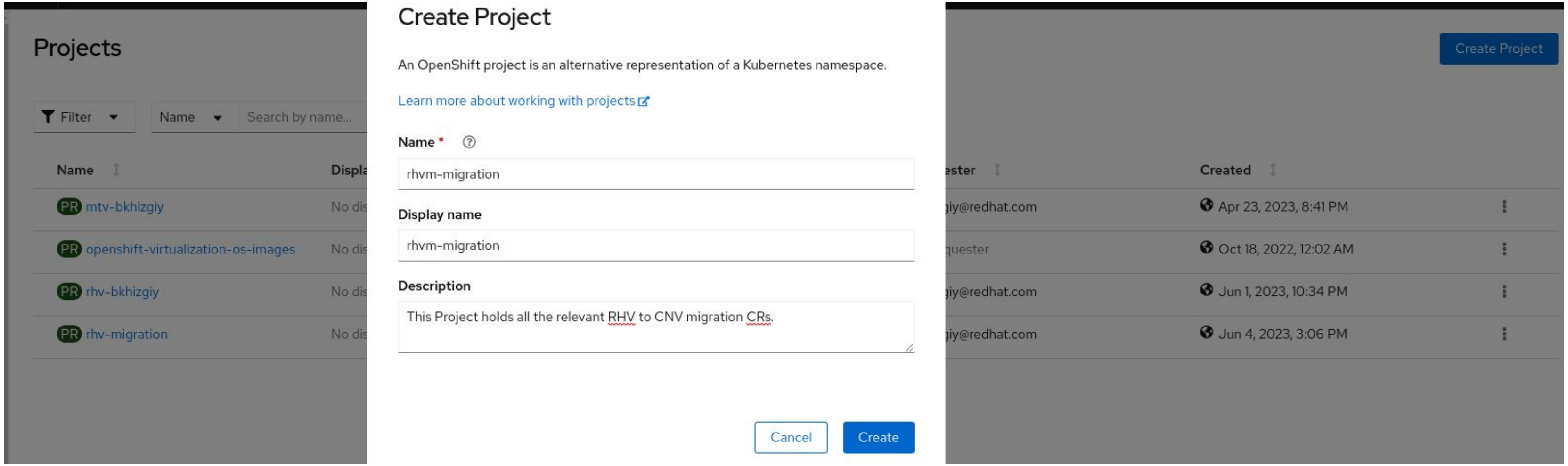
- Choosing the target OKD cluster to facilitate the migration process.
- Forklift can be installed from operator hub
- Managed by operator lifecycle manager (OLM).

Migration

Adding Project

Ensure the User used has adequate permissions for configuring the project to manage migration CRs.

- Source provider
- Target provider
- Network Mapping
- Storage Mapping
- Migration Plans



Projects

Name	Displa	ester	Created
PR mtv-bkhizgiy	No dis	giy@redhat.com	Apr 23, 2023, 8:41 PM
PR openshift-virtualization-os-images	No dis	quester	Oct 18, 2022, 12:02 AM
PR rhv-bkhizgiy	No dis	giy@redhat.com	Jun 1, 2023, 10:34 PM
PR rhv-migration	No dis	giy@redhat.com	Jun 4, 2023, 3:06 PM

Create Project

An OpenShift project is an alternative representation of a Kubernetes namespace.

Learn more about working with projects

Name * rhvm-migration

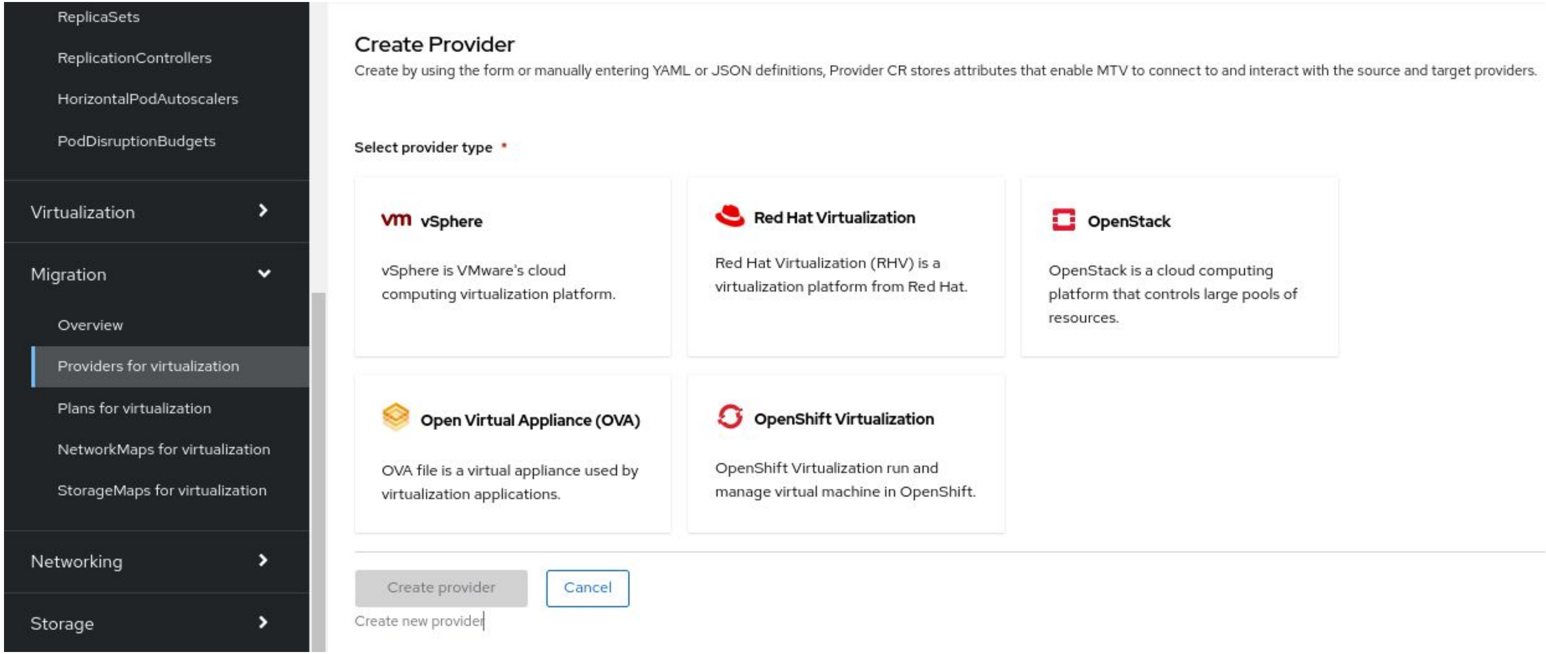
Display name rhvm-migration

Description This Project holds all the relevant RHV to CNV migration CRs.

Migration

Add Providers

- Using the web console, create the source and target providers.
- Can be found under Migration **->** Providers for virtualization **->** Create provider.



Create Provider

Create by using the form or manually entering YAML or JSON definitions, Provider CR stores attributes that enable MTV to connect to and interact with the source and target providers.

Select provider type *

vSphere
vSphere is VMware's cloud computing virtualization platform.

Red Hat Virtualization
Red Hat Virtualization (RHV) is a virtualization platform from Red Hat.

OpenStack
OpenStack is a cloud computing platform that controls large pools of resources.

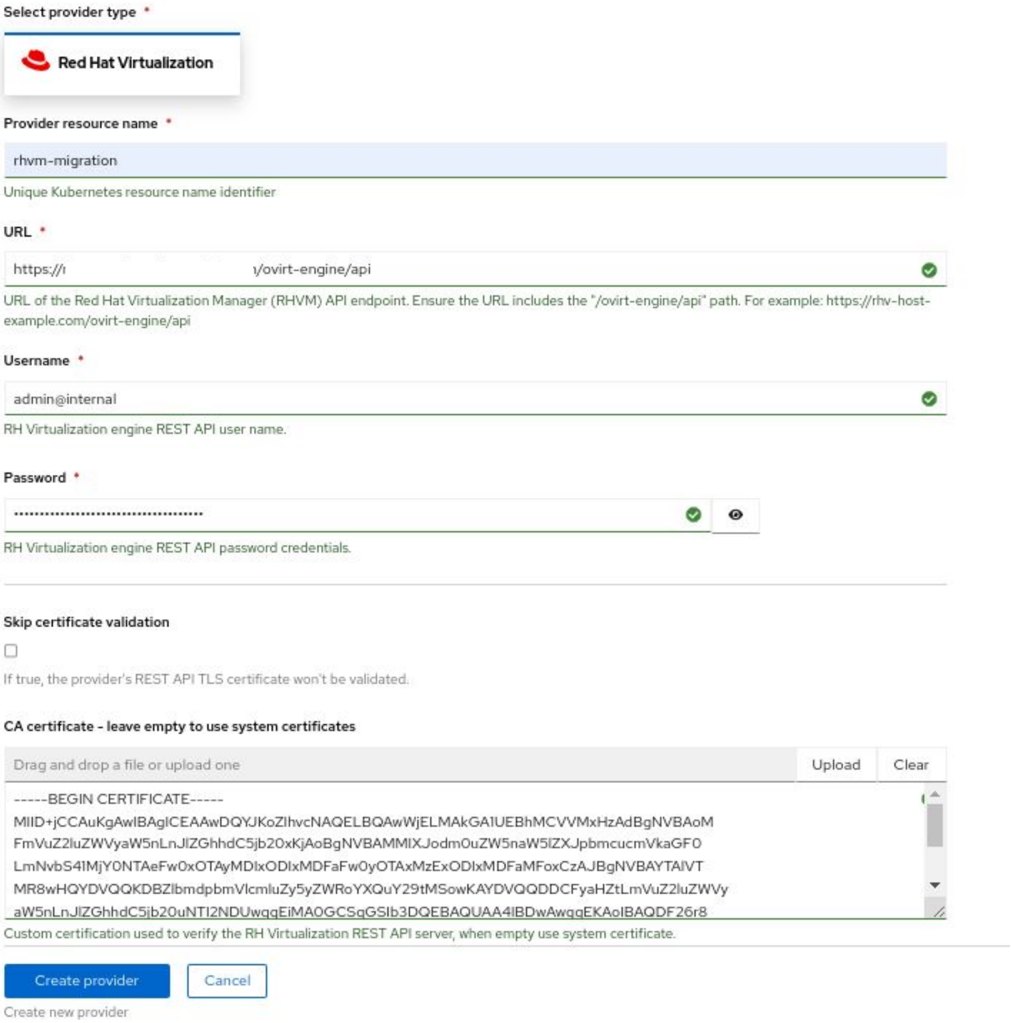
Open Virtual Appliance (OVA)
OVA file is a virtual appliance used by virtualization applications.

OpenShift Virtualization
OpenShift Virtualization run and manage virtual machine in OpenShift.

Create new provider

Migration

Create Source Provider



Select provider type *

Red Hat Virtualization

Provider resource name *

rhvm-migration

Unique Kubernetes resource name identifier

URL *

https:// /ovirt-engine/api

URL of the Red Hat Virtualization Manager (RHVM) API endpoint. Ensure the URL includes the "/ovirt-engine/api" path. For example: https://rhv-host-example.com/ovirt-engine/api

Username *

admin@internal

RH Virtualization engine REST API user name.

Password *

••••••••••••••••••••••••••••••••••••

RH Virtualization engine REST API password credentials.

Skip certificate validation

□

If true, the provider's REST API TLS certificate won't be validated.

CA certificate - leave empty to use system certificates

Drag and drop a file or upload one

```
-----BEGIN CERTIFICATE-----
MIID+jCCAuKgAwIBAgICEAAwDQYJKoZIhvcNAQELBQAwWjELMAkGA1UEBhMCVVMxHzAdBgNVBAoM
FmVuZ2luZWVyaW5nLnJlZGhhdC5jb20xKjAoBgNVBAMMIXJodm0uZW5naW5lZXJpbmcucmVkaGF0
LmNvbS41MjY0NTAeFw0xOTAyMDIxODIxMDFaFw0yOTAxMzExODIxMDFaMFoxCzAJBgNVBAYTAlVT
MR8wHQYDVQQKDBZlbmdpbmVlcmluZy5yZWRoYXQuY29tMSowKAYDVQQDDCFyaHZtLmVuZ2luZWVy
aW5nLnJlZGhhdC5jb20uNTI2NDUwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDF26r8
```

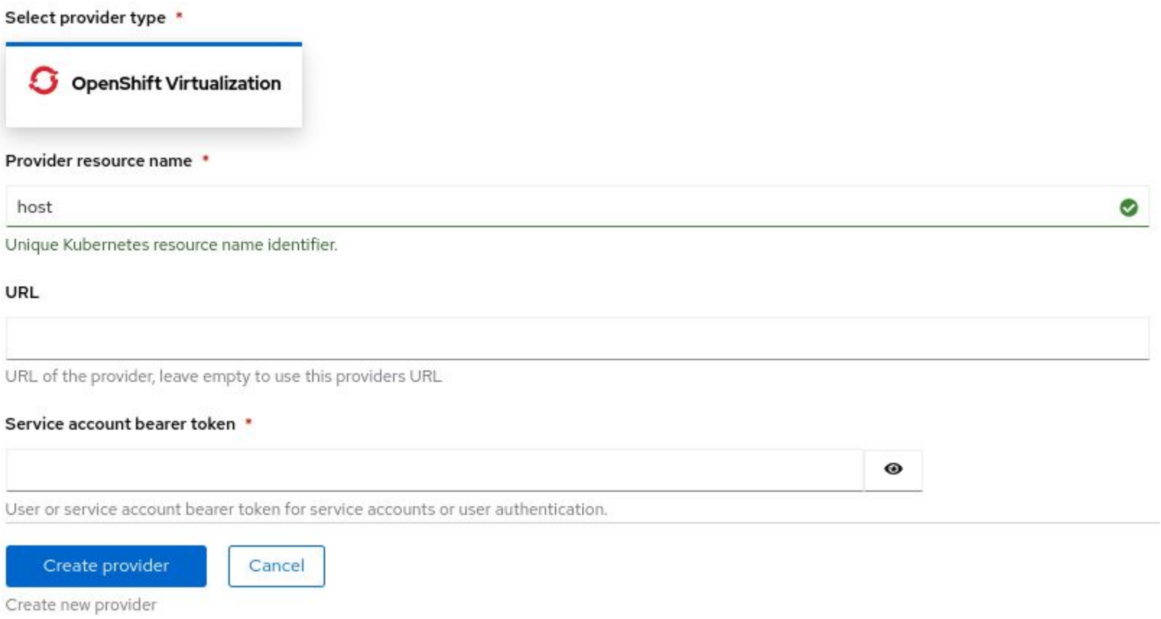
Custom certification used to verify the RH Virtualization REST API server, when empty use system certificate.

Create new provider

- Using the Forklift web console, set source provider for the migration.
- Provider Specification: Selected 'RHV' as the provider type.
- User Permissions: Ensure the user account configuring the provider has the necessary permissions on the VMs designated for migration.

Migration

Create Target Provider



Select provider type *

OpenShift Virtualization

Provider resource name *

host

Unique Kubernetes resource name identifier.

URL

URL of the provider, leave empty to use this providers URL

Service account bearer token *

User or service account bearer token for service accounts or user authentication.

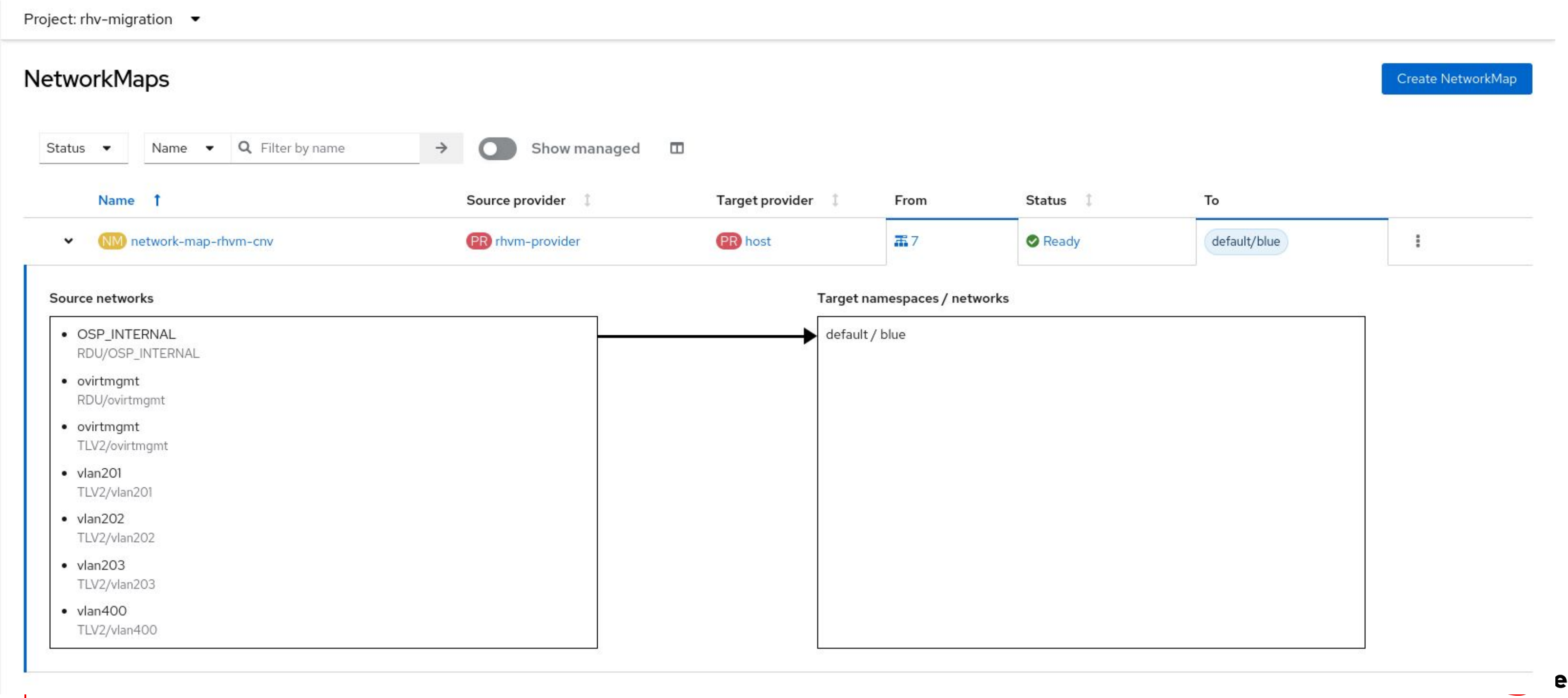
Create new provider

- Using the Forklift web console, set target provider for the migration.
- The provider type is OCP.
- When using the default Forklift namespace, the local cluster provider is automatically available.

Migration

Create Network Mapping

Using the web console, map all network resources from the oVirt source cluster to the corresponding resources in the new OKD cluster.



Project: rhv-migration

NetworkMaps

Name	Source provider	Target provider	From	Status	To
NM network-map-rhvm-cnv	PR rhvm-provider	PR host	7	✓ Ready	default/blue

Source networks

- OSP_INTERNAL
  RDU/OSP_INTERNAL
- ovirtmgmt
  RDU/ovirtmgmt
- ovirtmgmt
  TLV2/ovirtmgmt
- vlan201
  TLV2/vlan201
- vlan202
  TLV2/vlan202
- vlan203
  TLV2/vlan203
- vlan400
  TLV2/vlan400

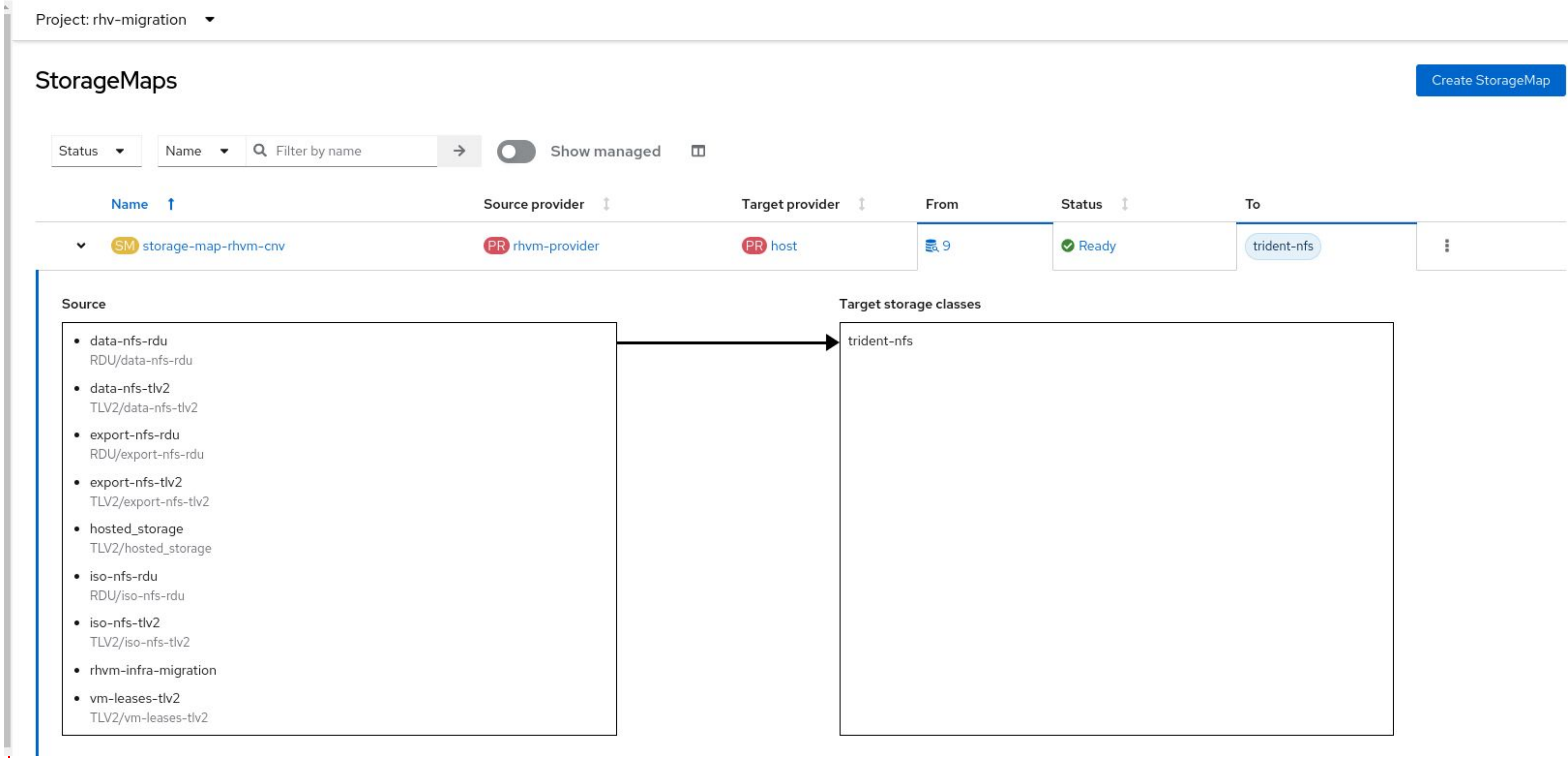
Target namespaces / networks

default / blue

Migration

Create Storage Mapping

Using the web console, map all storage resources from the ovirt source cluster to the corresponding resources in the new OKD cluster.



Project: rhv-migration

StorageMaps

Name	Source provider	Target provider	From	Status	To
SM storage-map-rhvm-cnv	PR rhvm-provider	PR host	9	Ready	trident-nfs

Source

- data-nfs-rdu
  RDU/data-nfs-rdu
- data-nfs-tlv2
  TLV2/data-nfs-tlv2
- export-nfs-rdu
  RDU/export-nfs-rdu
- export-nfs-tlv2
  TLV2/export-nfs-tlv2
- hosted_storage
  TLV2/hosted_storage
- iso-nfs-rdu
  RDU/iso-nfs-rdu
- iso-nfs-tlv2
  TLV2/iso-nfs-tlv2
- rhvm-infra-migration
- vm-leases-tlv2
  TLV2/vm-leases-tlv2

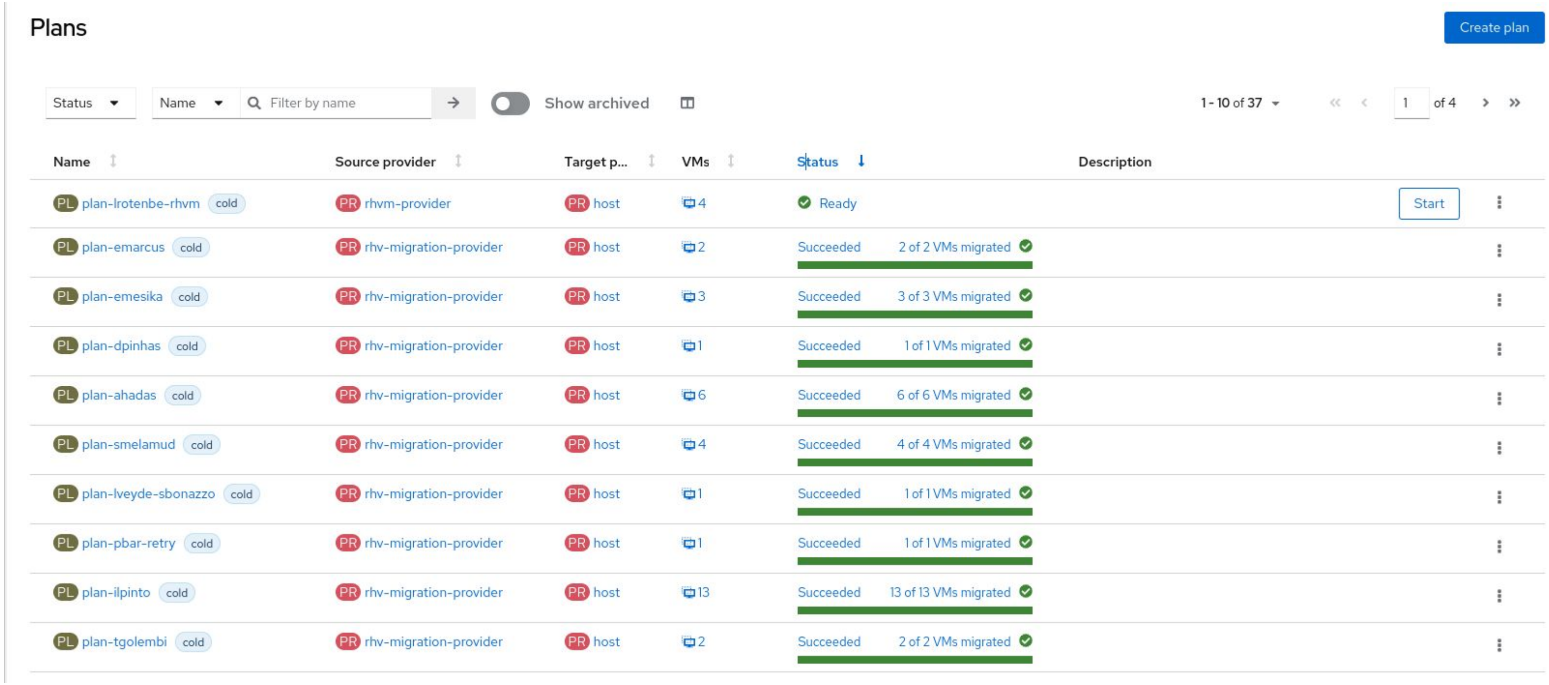
Target storage classes

trident-nfs

Migration

Create Plans

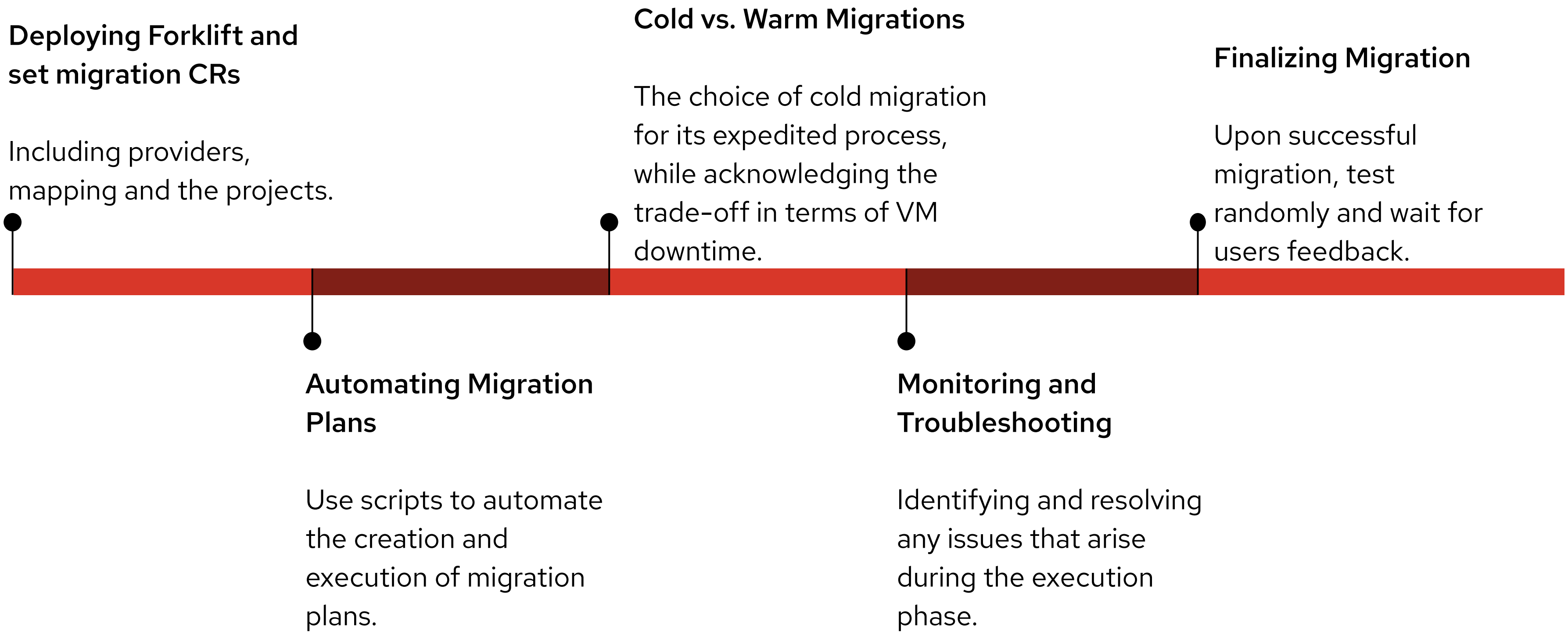
- Create all migration plans using scripts.
- The script assigns each migration plan with its target namespace.
- Using scripts, trigger all the desired migrations.



Name	Source provider	Target p...	VMs	Status	Description
PL plan-lrotenbe-rhvm cold	PR rhvm-provider	PR host	4	Ready	
PL plan-emarcus cold	PR rhv-migration-provider	PR host	2	Succeeded 2 of 2 VMs migrated	
PL plan-emesika cold	PR rhv-migration-provider	PR host	3	Succeeded 3 of 3 VMs migrated	
PL plan-dpinhas cold	PR rhv-migration-provider	PR host	1	Succeeded 1 of 1 VMs migrated	
PL plan-ahadas cold	PR rhv-migration-provider	PR host	6	Succeeded 6 of 6 VMs migrated	
PL plan-smelamud cold	PR rhv-migration-provider	PR host	4	Succeeded 4 of 4 VMs migrated	
PL plan-lveyde-sbonazzo cold	PR rhv-migration-provider	PR host	1	Succeeded 1 of 1 VMs migrated	
PL plan-pbar-retry cold	PR rhv-migration-provider	PR host	1	Succeeded 1 of 1 VMs migrated	
PL plan-ilpinto cold	PR rhv-migration-provider	PR host	13	Succeeded 13 of 13 VMs migrated	
PL plan-tgolembi cold	PR rhv-migration-provider	PR host	2	Succeeded 2 of 2 VMs migrated	

Migration

Strategy & Execution



Deploying Forklift and set migration CRs

Including providers, mapping and the projects.

Automating Migration Plans

Use scripts to automate the creation and execution of migration plans.

Cold vs. Warm Migrations

The choice of cold migration for its expedited process, while acknowledging the trade-off in terms of VM downtime.

Monitoring and Troubleshooting

Identifying and resolving any issues that arise during the execution phase.

Finalizing Migration

Upon successful migration, test randomly and wait for users feedback.

Migration

Challenges During the Migration

1 Network Bandwidth Limitations

Executing numerous migration plans simultaneously strained network resources, leading to slower migration speeds.

2 Managing Parallel Migrations

The challenge of handling multiple migrations concurrently and the need for effective coordination storage and network effects.

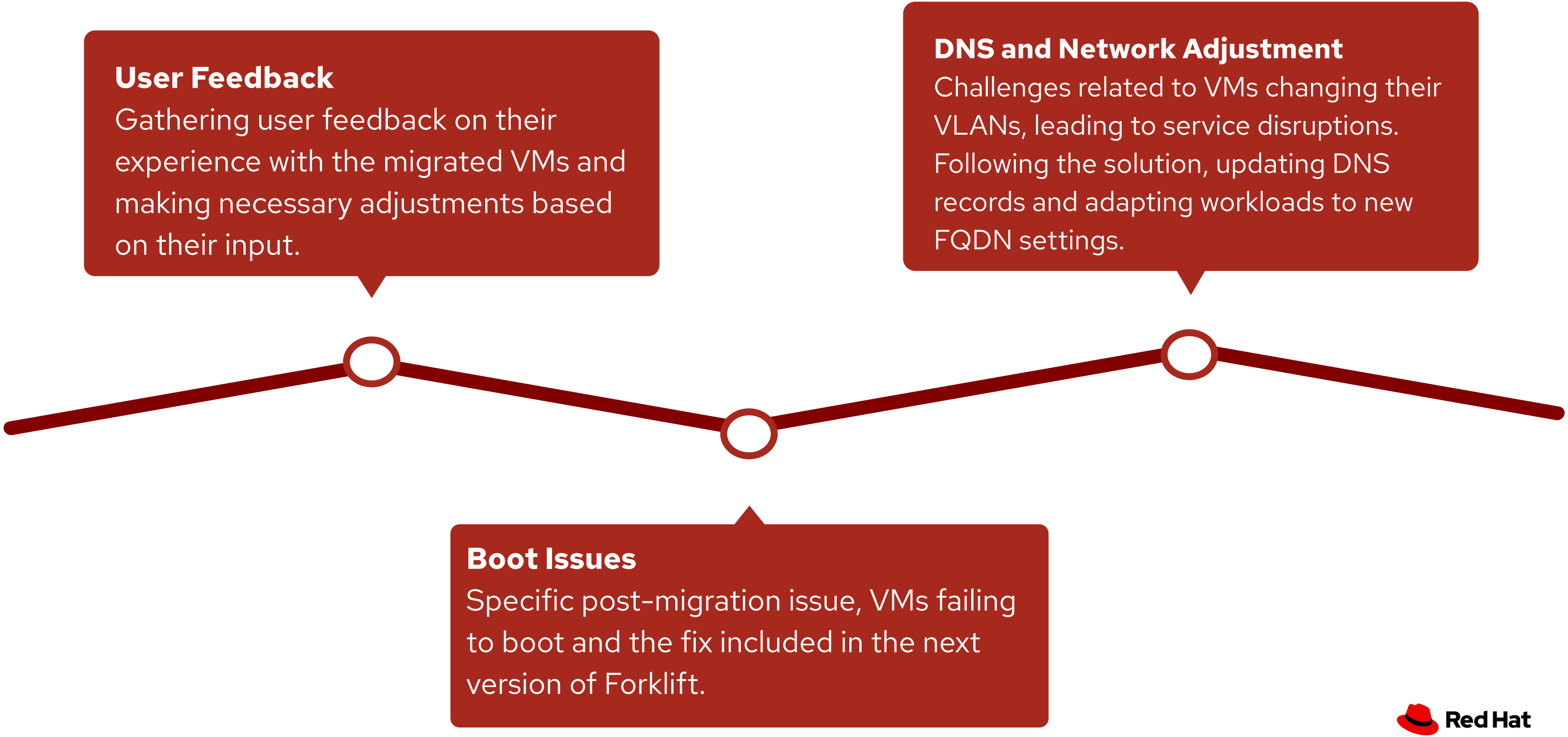
3 Encountering codebase issue

During the migration we found an issue in our codebase which had to be handled quickly to resume the transition. The fix was included in the next version.

4 Communication & Coordination

The importance of maintaining clear communication among team members and with end-users, especially when involving downtime.

Post-Migration



User Feedback
Gathering user feedback on their experience with the migrated VMs and making necessary adjustments based on their input.

DNS and Network Adjustment
Challenges related to VMs changing their VLANs, leading to service disruptions. Following the solution, updating DNS records and adapting workloads to new FQDN settings.

Boot Issues
Specific post-migration issue, VMs failing to boot and the fix included in the next version of Forklift.

Conclusions

Successful Transition
More than 100 vms and 12TB of data was migrated successfully.

Beyond the Forklift
Although the forklift tool was crucial, we've learned that it's not enough on its own. Additional steps and planning are essential for a successful migration

Importance of Planning
Thorough pre-migration planning was essential in identifying potential challenges and ensuring resource.

Different Environments
Although each migration experiences is different, there are some common ground.

Questions?

Blog Post