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

FOSDEM
February 3, 2024

Bella Khizgiyaev
bkhizgiy@redhat.com
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

<table>
<thead>
<tr>
<th></th>
<th>Cost</th>
<th>Practicality</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Re-provisioning everything</td>
<td>![Neutral emoji]</td>
</tr>
<tr>
<td>2</td>
<td>Migrate workloads into containers using KONVEYOR</td>
<td>![Neutral emoji]</td>
</tr>
<tr>
<td>3</td>
<td>Migrate existing workloads using Forklift</td>
<td>![Happy emoji]</td>
</tr>
</tbody>
</table>
Forklift

- vSphere
- oVirt
- OpenStack
- VMware OVA
- KubeVirt

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.

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.

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.
Pre-Migration

Planning

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

The scripts [https://github.com/kubev2v/rhv-ocp-migration-scripts](https://github.com/kubev2v/rhv-ocp-migration-scripts)

---

**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.
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.

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.

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.
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.

<table>
<thead>
<tr>
<th>VM</th>
<th>Owners</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ba_shared_vm_1</code></td>
<td>Bob, Alice</td>
</tr>
<tr>
<td><code>ba_shared_vm_2</code></td>
<td>Bob, Alice</td>
</tr>
<tr>
<td><code>bob_vm</code></td>
<td>Bob</td>
</tr>
<tr>
<td><code>bm_shared_vm</code></td>
<td>Bob, Michael</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Project</th>
<th>VMs</th>
<th>Project Admins</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>bob-alice-ns</code></td>
<td><code>ba_shared_vm_1</code></td>
<td>Bob, Alice</td>
</tr>
<tr>
<td></td>
<td><code>ba_shared_vm_2</code></td>
<td></td>
</tr>
<tr>
<td><code>bob-ns</code></td>
<td><code>bob_vm</code></td>
<td>Bob</td>
</tr>
<tr>
<td><code>bob-michael-ns</code></td>
<td><code>bm_shared_vm</code></td>
<td>Bob, Michael</td>
</tr>
</tbody>
</table>
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.
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
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

Create Project

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

Learn more about working with projects →

Name

rhm-migration

Display name

rhm-migration

Description

This Project holds all the relevant RHV to CNV migration CRs.
Using the web console, create the source and target providers.

- Can be found under Migration → Providers for virtualization → Create 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.
Create Target 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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Source provider</th>
<th>Target provider</th>
<th>From</th>
<th>Status</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>network-map-rhv-cow</td>
<td>rhvm-provider</td>
<td>rbd host</td>
<td>7</td>
<td>Ready</td>
<td>default/blue</td>
</tr>
</tbody>
</table>

Source networks:
- OSP INTERNAL
- RDU/OSP_INTERNAL
- ovirtmgmt
- RDU/ovirtmgmt
- ovirtmgmt
- TIV2/ovirtmgmt
- vlan201
- TIV2/vlan201
- vlan202
- TIV2/vlan202
- vlan203
- TIV2/vlan203
- vlan400
- TIV2/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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Source provider</th>
<th>Target provider</th>
<th>Status</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>storage-map-rhv-m-cnw</td>
<td>rhv-provider</td>
<td>host</td>
<td>Ready</td>
<td>trident-nfs</td>
</tr>
</tbody>
</table>

- data-nfs-rdu
- data-nfs-tlv2
- export-nfs-rdu
- export-nfs-tlv2
- hosted_storage
- iso-nfs-rdu
- iso-nfs-tlv2
- rhv-infra-migration
- vm-keaas-tl2
- TLV2/vm-keaas-tlv2

Source 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.

Plans

<table>
<thead>
<tr>
<th>Name</th>
<th>Source provider</th>
<th>Target p...</th>
<th>VMs</th>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>plan-ibotnebe-rhvm</td>
<td>rhvm-provider</td>
<td>host</td>
<td>4</td>
<td>Ready</td>
<td>2 of 2 VMs migrated</td>
</tr>
<tr>
<td>plan-emarcus</td>
<td>rhv-migration-provision</td>
<td>host</td>
<td>2</td>
<td>Succeeded</td>
<td>3 of 3 VMs migrated</td>
</tr>
<tr>
<td>plan-amvuska</td>
<td>rhv-migration-provision</td>
<td>host</td>
<td>3</td>
<td>Succeeded</td>
<td>1 of 1 VMs migrated</td>
</tr>
<tr>
<td>plan-dpinhas</td>
<td>rhv-migration-provision</td>
<td>host</td>
<td>1</td>
<td>Succeeded</td>
<td>1 of 1 VMs migrated</td>
</tr>
<tr>
<td>plan-shadas</td>
<td>rhv-migration-provision</td>
<td>host</td>
<td>6</td>
<td>Succeeded</td>
<td>6 of 6 VMs migrated</td>
</tr>
<tr>
<td>plan-simalmus</td>
<td>rhv-migration-provision</td>
<td>host</td>
<td>4</td>
<td>Succeeded</td>
<td>4 of 4 VMs migrated</td>
</tr>
<tr>
<td>plan-heyde-sbonazzo</td>
<td>rhv-migration-provision</td>
<td>host</td>
<td>1</td>
<td>Succeeded</td>
<td>1 of 1 VMs migrated</td>
</tr>
<tr>
<td>plan-pbar-retty</td>
<td>rhv-migration-provision</td>
<td>host</td>
<td>1</td>
<td>Succeeded</td>
<td>1 of 1 VMs migrated</td>
</tr>
<tr>
<td>plan-lpinto</td>
<td>rhv-migration-provision</td>
<td>host</td>
<td>13</td>
<td>Succeeded</td>
<td>13 of 13 VMs migrated</td>
</tr>
<tr>
<td>plan-tgolembi</td>
<td>rhv-migration-provision</td>
<td>host</td>
<td>2</td>
<td>Succeeded</td>
<td>2 of 2 VMs migrated</td>
</tr>
</tbody>
</table>
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.
Network Bandwidth Limitations

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

Managing Parallel Migrations

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

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.

Communication & Coordination

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

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

**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.
Conclusions

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

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

**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.

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